VIPA System 200V

IM | Manual

HB97E_IM | Rev. 12/33

August 2012

VIPA art of automation

For permission to reproduce or distribute, please contact:
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 (Hotline)
EMail: support@vipa.de

About this manual

This manual describes the System 200V IM modules that are available from VIPA. In addition to the product summary it contains detailed descriptions of the different modules. You are provided with information on the connection and the utilization of the System 200V IM modules. Every chapter is concluded with the technical data of the respective module.

Overview

Chapter 1: Basics

This introduction presents the VIPA System 200V as a centralized as well as decentralized automation system.

The chapter also contains general information about the System 200V, i.e. dimensions, installation and operating conditions.

Chapter 2: Assembly and installation guidelines

This chapter provides all the information required for the installation and the hook-up of a controller using the components of the System 200V.

Chapter 3: Profibus-DP

This chapter contains a description of Profibus applications for the System 200V. The text describes the configuration of the VIPA Profibus master and slave modules as well as a number of different communication examples.

Chapter 4: Interbus

This chapter contains all the information that is required to provide a connection between the System 200V peripherals and Interbus. It contains descriptions of the construction, commissioning and the configuration of the Interbus coupler.

Chapter 5: CAN bus CANopen

This chapter deals with the VIPA CANopen slave and related CAN bus applications. The structure of the program and the configuration of CAN slaves is explained by means of examples.

Chapter 6: DeviceNet

This chapter contains a description of the VIPA DeviceNet coupler. A description of the module is followed by an example of the configuration of the DeviceNet coupler and the configuration of the System 200V modules in the DeviceNet manager of Allen - Bradley. The chapter is concluded with an overview of diagnostic messages and Profibus interfacing options.

Chapter 7: SERCOS

Content of this chapter is the description of the SERCOS coupler from VIPA. Another part of this chapter is the project engineering of the SERCOS coupler and the parameterization of the System 200V modules.

Chapter 8: Ethernet coupler

Content of this chapter is the description of the Ethernet coupler IM 253NET from VIPA. It contains all information for installation and commissioning of the Ethernet coupler.

Chapter 9: Bus expansion modules IM 260 - IM 261

In this chapter follows the description of the bus expansion module IM 260 and IM 261 that is used to split a single System 200V row over up to 4 rows.

Contents

User considerations **1**
Safety information **2**
Chapter 1 Basics **1-1**
Safety information for Users 1-2
Overview 1-3
Components 1-4
General description System 200V 1-5
Chapter 2 Assembly and installation guidelines **2-1**
Overview 2-2
Assembly 2-5
Wiring 2-8
Assembly dimensions 2-10
Installation guidelines 2-12
Chapter 3 Profibus DP **3-1**
System overview 3-2
Basics 3-5
IM 208DP - Master - Structure 3-13
IM 208DP - Master - Deployment at CPU 21x 3-17
IM 208DP - Master - Project engineering 3-18
IM 208DP - Master - Slave operating mode 3-28
IM 208DP - Master - Overall reset 3-32
IM 208DP - Master - Firmware update 3-33
IM 253-1DPx0 - DP-V0 slave - Structure 3-35
IM 253-2DP20 - DP-V0 slave with DO 24xDC 24V - Structure 3-38
IM 253-2DP50 - DP-V0 slave (redundant) - Structure 3-42
IM 253-xDPx0 - DP-V0 slave - Block diagram 3-45
IM 253-xDPx0 - DP-V0 Slave - Project engineering 3-46
IM 253-xDPx0 - DP-V0 slave - Parameters 3-48
IM 253-xDPx0 - DP-V0 slave - Diagnostic functions 3-49
IM 253-xDPx1 - DP-V1 slave - Structure 3-56
IM 253-xDPx1 - DP-V1 slave - Block diagram 3-60
IM 253-xDPx1 - DP-V1 slave - Project engineering 3-61
IM 253-xDPx1 - DP-V1 slave - Parameters 3-63
IM 253-xDPx1 - DP-V1 slave - Diagnostic functions 3-68
IM 253-xDPx1 - DP-V1 slave - Firmware update 3-76
IM 253-xDPx1 - DP-V1 slave - I&M data 3-77
Installation guidelines 3-79
Commissioning 3-89
Using the diagnostic LEDs 3-90
Sample projects for Profibus communication 3-91
Technical data 3-99

Chapter 4 Interbus 4-1
System overview 4-2
Basics 4-3
IM 253IBS - Interbus coupler - Structure 4-7
Connection to Interbus 4-10
Deployment with Interbus 4-11
Commissioning 4-15
Technical data 4-18
Chapter 5 CANopen 5-1
System overview 5-2
Basics 5-4
IM 208CAN - CANopen master - Structure 5-6
IM 208CAN - CANopen master - Project engineering 5-8
IM 208CAN - CANopen master - Firmware update 5-15
IM 208CAN - CANopen master - Mode 5-16
IM 208CAN - CANopen master - Process image 5-17
IM 208CAN - CANopen master - Messages 5-19
IM 208CAN - CANopen master - Object directory 5-24
IM 253CAN - CANopen slave - Structure 5-38
IM 253CAN, DO 24xDC 24V - Structure 5-43
IM 253CAN - CANopen slave - Fast introduction 5-47
IM 253CAN - CANopen slave - Baudrate and module-ID 5-51
IM 253CAN - CANopen slave - Message structure 5-52
IM 253CAN - CANopen slave - PDO 5-54
IM 253CAN - CANopen slave - SDO 5-58
IM 253CAN - CANopen slave - Object directory 5-60
IM 253CAN - CANopen slave - Emergency Object 5-101
IM 253CAN - CANopen slave - NMT - network management 5-102
Technical data 5-104
Chapter 6 DeviceNet 6-1
System overview 6-2
Basics 6-3
IM 253DN - DeviceNet coupler - Structure 6-5
Configuration by means of the DeviceNet-Manager 6-8
Specifying baudrate and DeviceNet address 6-9
Test in conjunction with the DeviceNet 6-10
Module configuration in the DeviceNet-Manager 6-11
I/O addressing of the DeviceNet scanner 6-16
Diagnostics 6-17
Technical data 6-22
Chapter 7 SERCOS - Spare part 7-1
System overview 7-2
Basics 7-3
IM 253Sercos - SERCOS coupler - Structure 7-5
Basic parameterization via address adjuster 7-8
SERCOS Identifier 7-10
Example for the automatic ID assignment 7-13
Technical Data 7-22

Chapter 8 Ethernet coupler 8-1
System overview 8-2
Basics 8-3
Planning a network.. 8-7
IM 253NET - Ethernet coupler - Structure 8-9
Access to the Ethernet coupler.. 8-11
Principle of the automatic address allocation...................................... 8-14
Project engineering under WinNCS... 8-15
Diagnosis and test via Internet Browser 8-16
ModbusTCP 8-20
Modbus function codes 8-21
Siemens S5 Header Protocol 8-26
Principle of Alarm handling.. 8-28
Programming sample 8-32
Technical data... 8-33
Chapter 9 Bus expansion modules IM 260 - IM 261 9-1
Field of application 9-2
Wiring... 9-3
Status indication 9-4
Technical data.. 9-5
Appendix A-1
Index A-1

User considerations

Objective and contents

This manual describes the modules that are suitable for use in the System 200V. It contains a description of the construction, project implementation and 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 System 200V is constructed and produced for:

- all VIPA System 200V components
- 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

Overview

The focus of this chapter is on the introduction of the VIPA System 200V. Various options of configuring central and decentral systems are presented in a summary.

The chapter also contains the general specifications of the System 200V, i.e. dimensions, installation and environmental conditions.

Below follows a description of:

- Introduction of the System 200V
- General information, i.e. installation, operational safety and environmental conditions

Content

Topic **Page**

Chapter 1 Basics 1-1
Safety information for Users.. 1-2
Overview 1-3
Components.. 1-4
General description System 200V 1-5

Safety information for Users

Handling of electrostatically sensitive modules

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

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

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

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

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

Shipping of electrostatically sensitive modules

Modules have to be shipped in the original packing material.

Measurements and alterations on electrostatically sensitive modules

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

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

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

Attention!

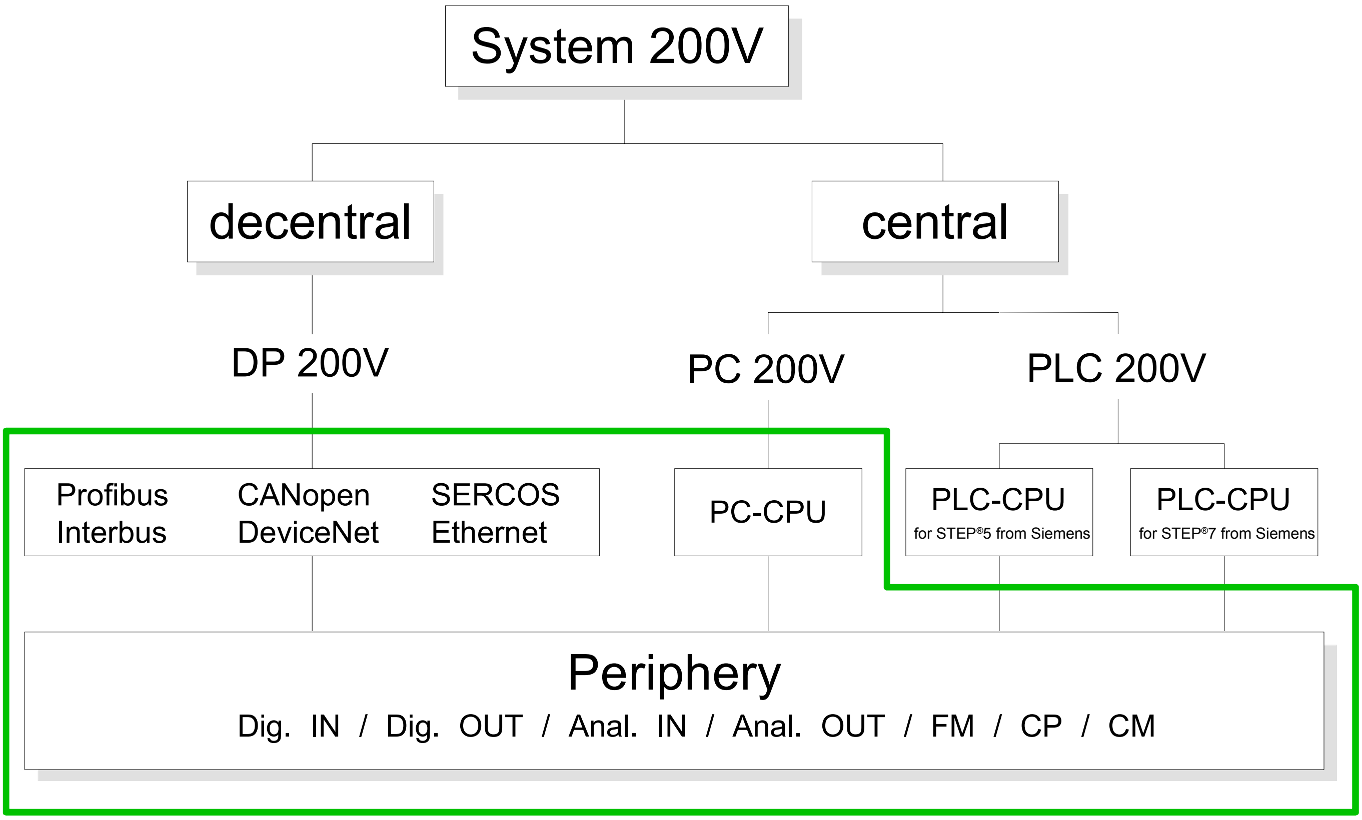
Personnel and instruments should be grounded when working on electrostatically sensitive modules.

Overview

The System 200V

The System 200V is a modular automation system for centralized and decentralized applications requiring low to medium performance specifications. The modules are installed directly on a 35mm DIN rail. Bus connectors inserted into the DIN rail provide the interconnecting bus.

The following figure illustrates the capabilities of the System 200V:



Components

Centralized system

The System 200V series consists of a number of PLC-CPUs. These are programmed in STEP®5 or STEP®7 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.

The PC based CPU 288 can be used to implement operating/monitoring tasks, control applications or other file processing applications.

The modules are programmed in C++ or Pascal.

The PC 288-CPU provides an active interface to the backplane bus and can therefore be employed as central controller for all peripheral and function modules of the VIPA System 200V.

With the appropriate expansion interface the System 200V can support up to 4 rows.

Decentralized 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 WinNCS VIPA configuration tool res. Siemens SIMATIC Manager.

Other fieldbus systems may be connected by means of slaves for Interbus, CANopen, DeviceNet, SERCOS and Ethernet.

Peripheral modules

A large number of peripheral modules are available from VIPA, for example digital as well as analog inputs/outputs, counter functions, displacement sensors, positioners and serial communication modules.

These peripheral modules can be used in centralized as well as decentralized mode.

Integration over GSD File

The functionality of all VIPA system components are available via different GSD-files.

For the Profibus interface is software standardized, we are able to guarantee the full functionality by including a GSD-file using the Siemens SIMATIC Manager.

For every system family there is an own GSD-file. Actual GSD files can be found at ftp.vipa.de/support.

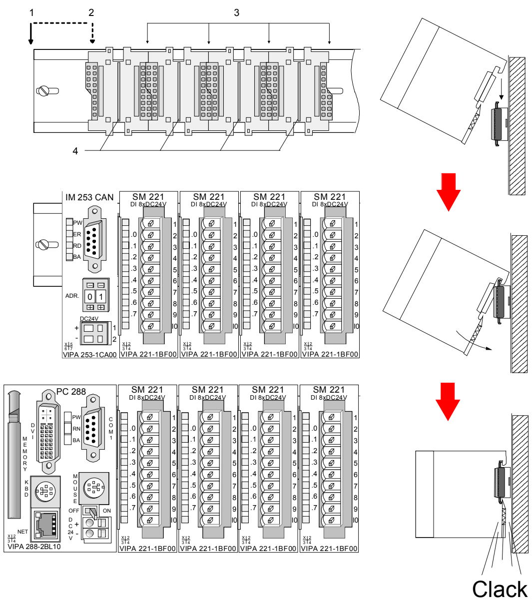
General description System 200V

Structure/ dimensions

- Standard 35mm DIN rail
- Peripheral modules with recessed labelling
- Dimensions of the basic enclosure:
  1tier width: (HxWxD) in mm: 76x25.4x74 in inches: 3x1x3
  2tier width: (HxWxD) in mm: 76x50.8x74 in inches: 3x2x3

Installation

Please note that you can only install header modules, like the CPU, the PC and couplers into plug-in location 1 or 1 and 2 (for double width modules).



[1] Header modules, like PC, CPU, bus couplers (double width)
[2] Header module (single width)
[3] Peripheral module
[4] Guide rails

Note

A maximum of 32 modules can be connected at the back plane bus. Take attention that here the **maximum sum current** of **3.5A** is not exeeded.

Please install modules with a high current consumption directly beside the header module.

Reliability

- Wiring by means of spring pressure connections (CageClamps) at the front-facing connector, core cross-section 0.08...2.5mm^2 or 1.5 mm^2 (18pole plug)
- Complete isolation of the wiring when modules are exchanged
- Every module is isolated from the backplane bus
- ESD/Burst acc. IEC 61000-4-2 / IEC 61000-4-4 (to level 3)
- Shock resistance acc. IEC 60068-2-6 / IEC 60068-2-27 (1G/12G)

Environmental conditions

- Operating temperature: 0 ... +60°C
- Storage temperature: -25 ... +70°C
- Relative humidity: 5 ... 95% without condensation
- Ventilation by means of a fan is not required

Chapter 2 Assembly and installation guidelines

Overview

This chapter contains the information required to assemble and wire a controller consisting of Systems 200V components.

Below follows a description of:

- a general summary of the components
- steps required for the assembly and for wiring
- EMC guidelines for assembling the System 200V

Content

Topic **Page**

Chapter 2 Assembly and installation guidelines........................... 2-1

Overview 2-2

Assembly... 2-5

Wiring... 2-8

Assembly dimensions.. 2-10

Installation guidelines 2-12

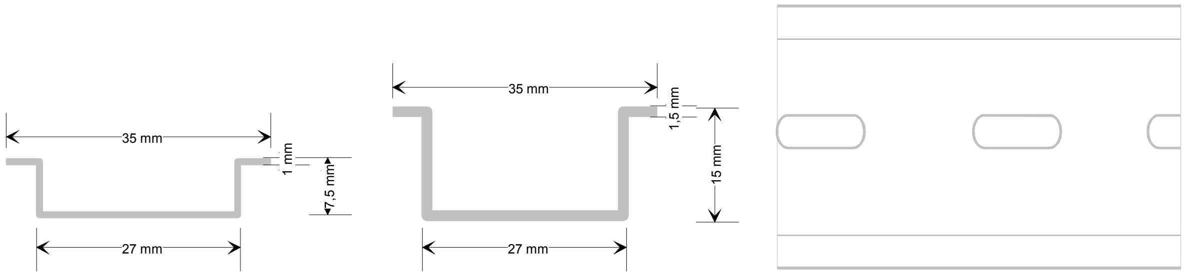
Overview

General

The modules are installed on a carrier rail. A bus connector provides interconnections between the modules. This bus connector links the modules via the backplane bus of the modules and it is placed into the profile rail that carries the modules.

Profile rail

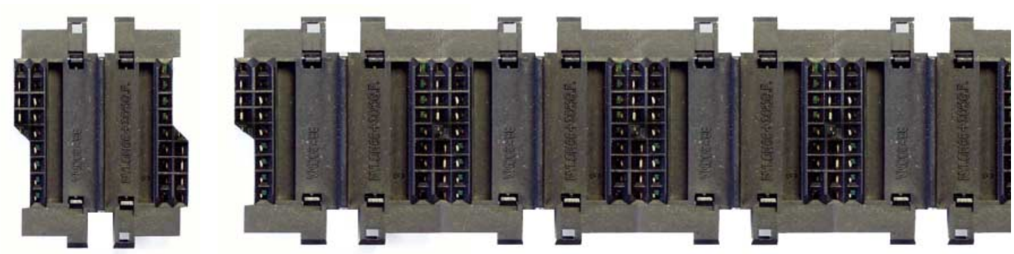
You may use the following standard 35mm profile rail to mount the System 200V modules:



Bus connector

System 200V modules communicate via a backplane bus connector. The backplane bus connector is isolated and available from VIPA in of 1-, 2-, 4- or 8tier width.

The following figure shows a 1tier connector and a 4tier connector bus:

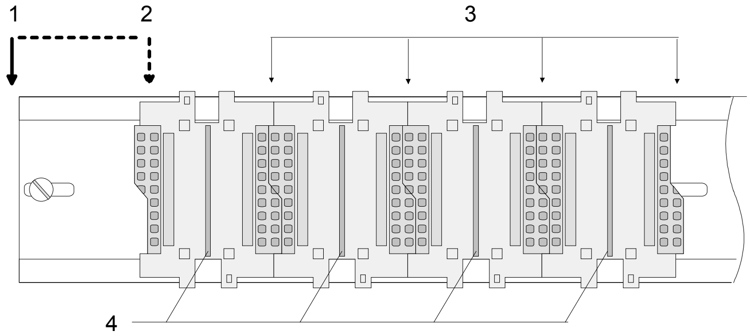


The bus connector is isolated and has to be inserted into the profile rail until it clips in its place and the bus connections protrude from the rail.

Profile rail installation

The following figure shows the installation of a 4tier width bus connector in a profile rail and the plug-in locations for the modules.

The different plug-in locations are defined by guide rails.

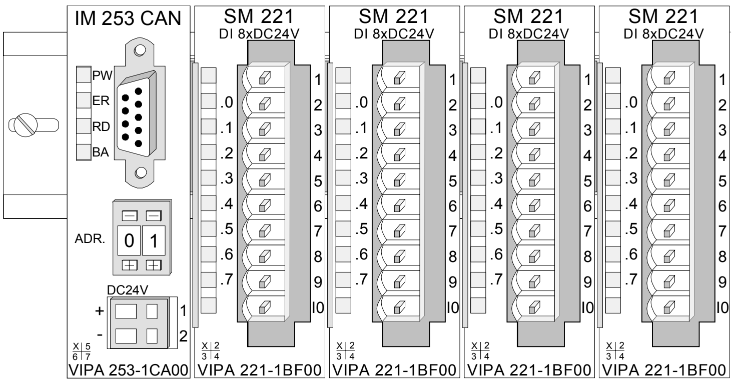


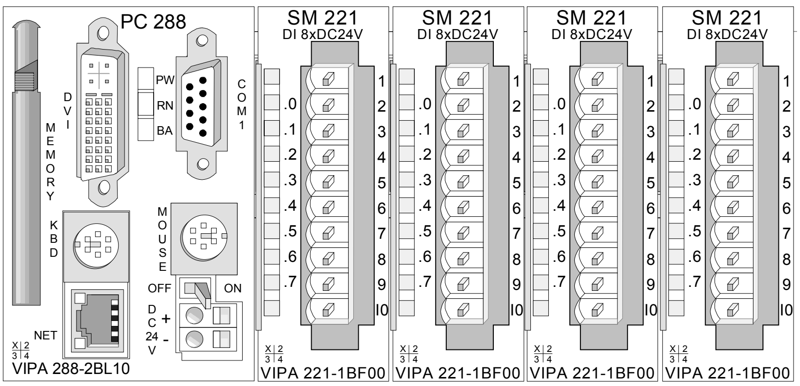
[1] Header module, like PC, CPU, bus coupler, if double width
[2] Header module (single width)
[3] Peripheral module
[4] Guide rails

Note

A maximum of 32 modules can be connected at the back plane bus.

Take attention that here the **maximum sum current** of **3.5A** is not exceeded.





Assembly regarding the current consumption

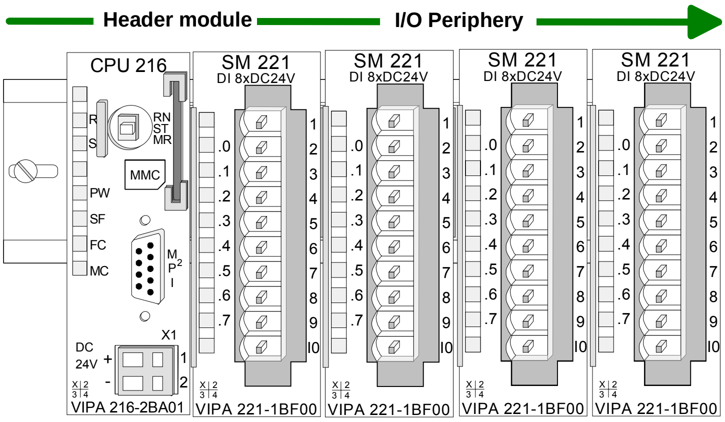
- Use bus connectors as long as possible.
- Sort the modules with a high current consumption right beside the header module. At ftp.vipa.de/manuals/system200v a list of current consumption of every System 200V module can be found.

Assembly horizontal respectively vertical

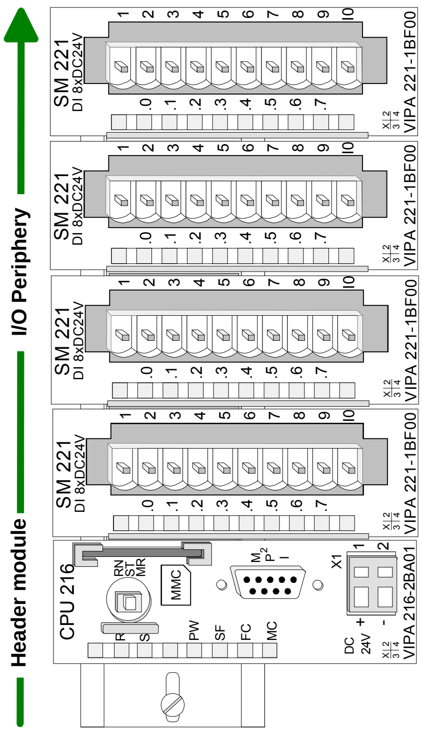
You may install the System 200V as well horizontal as vertical. Please regard the allowed environment temperatures:

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

The horizontal structure always starts at the left side with a header module (CPU, bus coupler, PC), then you plug-in the peripheral modules beside to the right. You may plug-in maximum 32 peripheral modules.



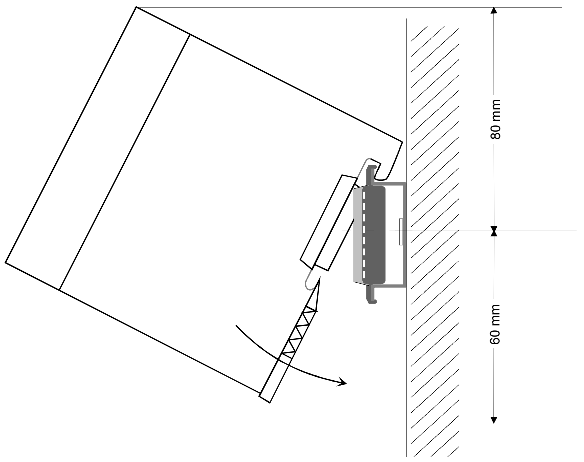
The vertical structure is turned for 90° against the clock.



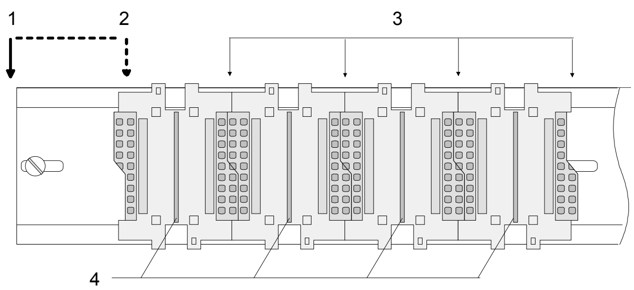
Assembly

Please follow these rules during the assembly!

- Turn off the power supply before you insert or remove any modules!
- Make sure that a clearance of at least 60mm exists above and 80mm below the middle of the bus rail.



- Every row must be completed from left to right and it has to start with a header module (PC, CPU, and bus coupler).



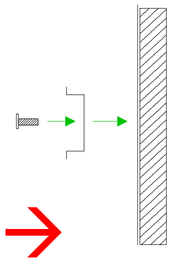
[1] Header module, like PC, CPU, bus coupler, if double width
[2] Header module (single width)
[3] Peripheral module
[4] Guide rails

- Modules are to install adjacent to each other. Gaps are not permitted between the modules since this would interrupt the backplane bus.
- A module is only installed properly and connected electrically when it has clicked into place with an audible click.
- Plug-in locations after the last module may remain unoccupied.

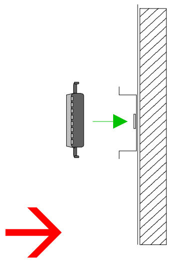
Note!
A maximum of 32 modules can be connected at the back plane bus. Take attention that here the maximum **sum current** of **3.5A** is not exceeded.

Assembly procedure

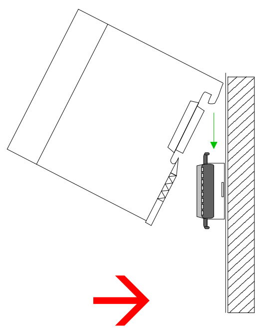
The following sequence represents the assembly procedure as viewed from the side.



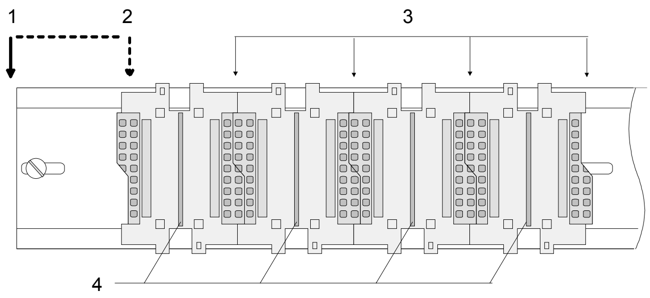
- Install the profile rail. Make sure that a clearance of at least 60mm exists above and 80mm below the middle of the bus rail.



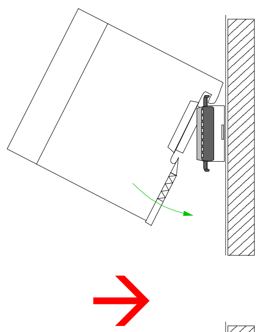
- Press the bus connector into the rail until it clips securely into place and the bus-connectors protrude from the profile rail. This provides the basis for the installation of your modules.



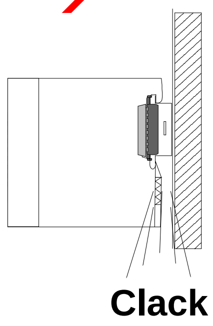
- Start at the outer left location with the installation of your header module like CPU, PC or bus coupler and install the peripheral modules to the right of this.



[1] Header module like PC, CPU, bus coupler
[2] Header module when this is a double width or a peripheral module
[3] Peripheral module
[4] Guide rails



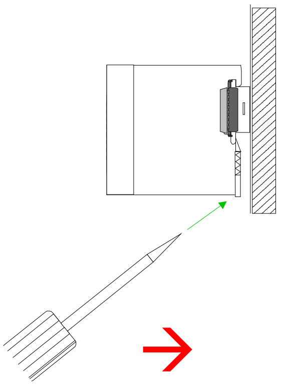
- Insert the module that you are installing into the profile rail at an angle of 45 degrees from the top and rotate the module into place until it clicks into the profile rail with an audible click. The proper connection to the backplane bus can only be guaranteed when the module has properly clicked into place.



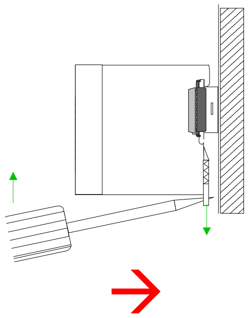
Attention!
Power must be turned off before modules are installed or removed!

Removal procedure

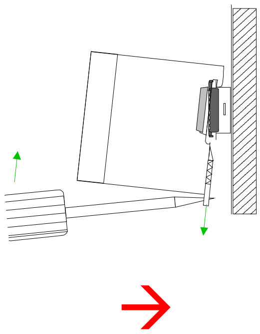
The following sequence shows the steps required for the removal of modules in a side view.



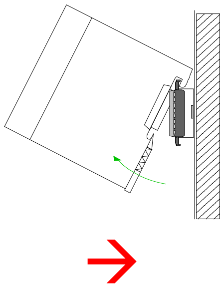
- The enclosure of the module has a spring-loaded clip at the bottom by which the module can be removed from the rail.
- Insert a screwdriver into the slot as shown.

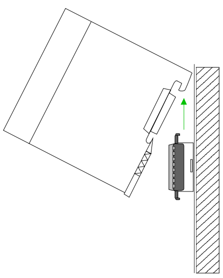


- The clip is unlocked by pressing the screwdriver in an upward direction.



- Withdraw the module with a slight rotation to the top.





Attention!

Power must be turned off before modules are installed or removed!

Please remember that the backplane bus is interrupted at the point where the module was removed!

Wiring

Outline

Most peripheral modules are equipped with a 10pole or an 18pole connector. This connector provides the electrical interface for the signaling and supply lines of the modules.

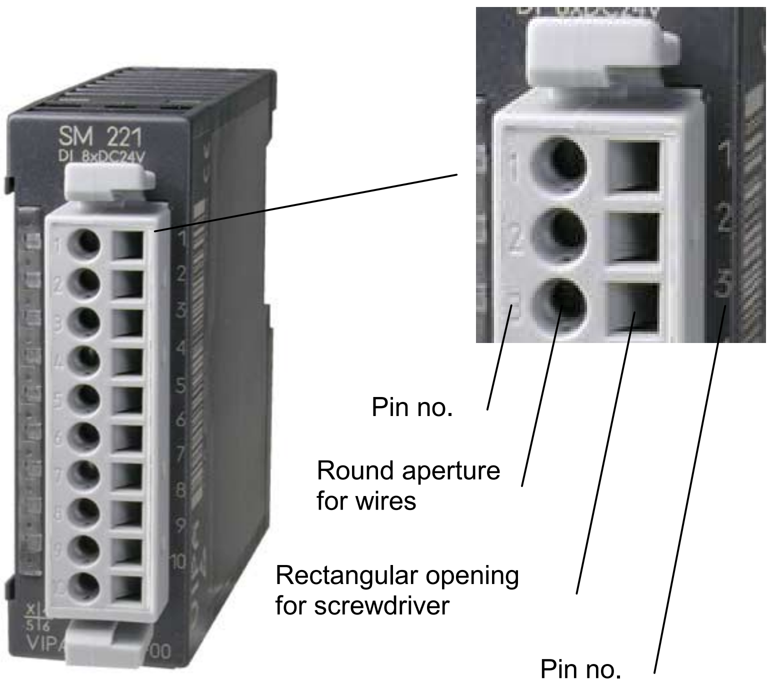
The modules carry spring-clip connectors for the interconnections and wiring.

The spring-clip connector technology simplifies the wiring requirements for signaling and power cables.

In contrast to screw terminal connections, spring-clip wiring is vibration proof. The assignment of the terminals is contained in the description of the respective modules.

You may connect conductors with a diameter from 0.08mm^2 up to 2.5mm^2 (max. 1.5mm^2 for 18pole connectors).

The following figure shows a module with a 10pole connector.

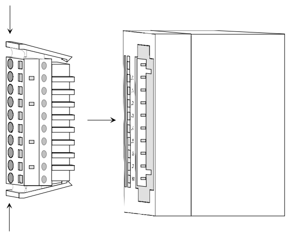


Note!

The spring-clip is destroyed if you insert the screwdriver into the opening for the hook-up wire!

Make sure that you only insert the screwdriver into the square hole of the connector!

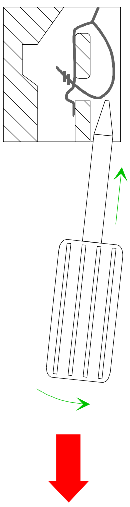
Wiring procedure



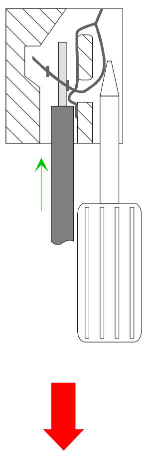
- Install the connector on the module until it locks with an audible click. For this purpose you press the two clips together as shown.

  The connector is now in a permanent position and can easily be wired.

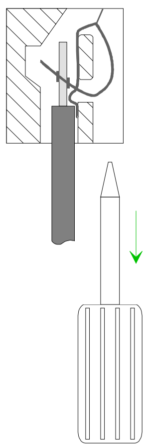
The following section shows the wiring procedure from above.



- Insert a screwdriver at an angel into the square opening as shown.
- Press and hold the screwdriver in the opposite direction to open the contact spring.



- Insert the stripped end of the hook-up wire into the round opening. You can use wires with a diameter of $0.08mm^2$ to $2.5mm^2$ ($1.5mm^2$ for 18pole connectors).



- When you remove the screwdriver, the wire is clipped securely.

Wire the power supply connections first
followed by the signal cables (inputs and outputs).

Assembly dimensions

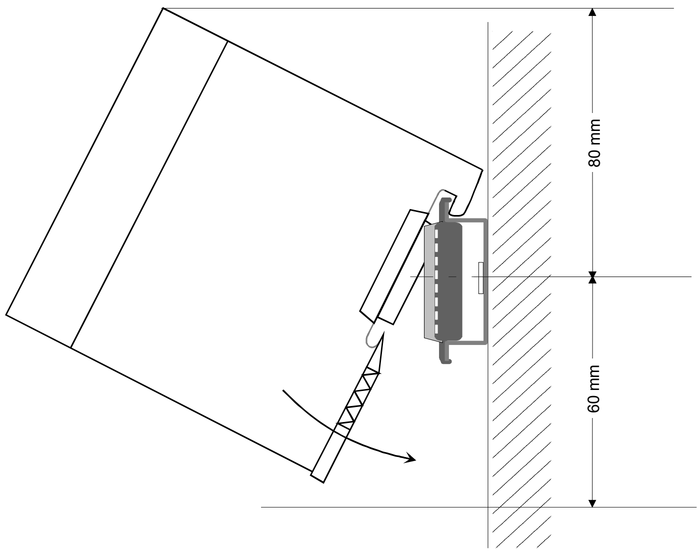
Overview

Here follow all the important dimensions of the System 200V.

Dimensions Basic enclosure

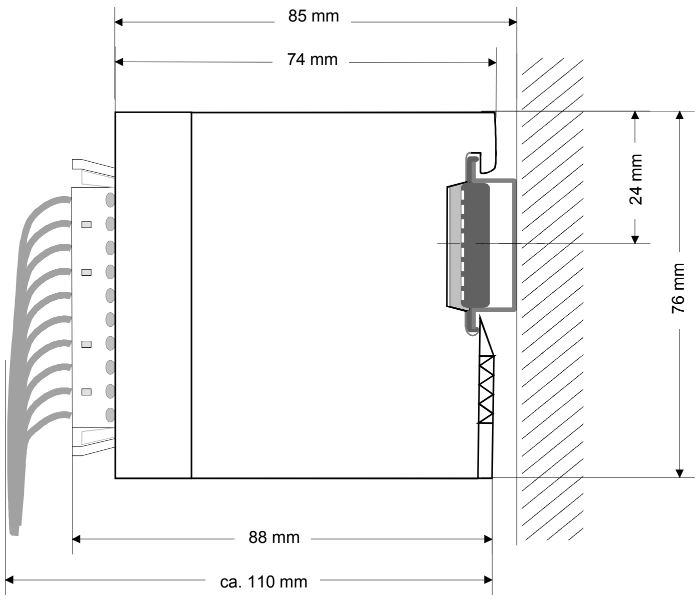
1tier width (HxWxD) in mm: 76 x 25.4 x 74
2tier width (HxWxD) in mm: 76 x 50.8 x 74

Installation dimensions

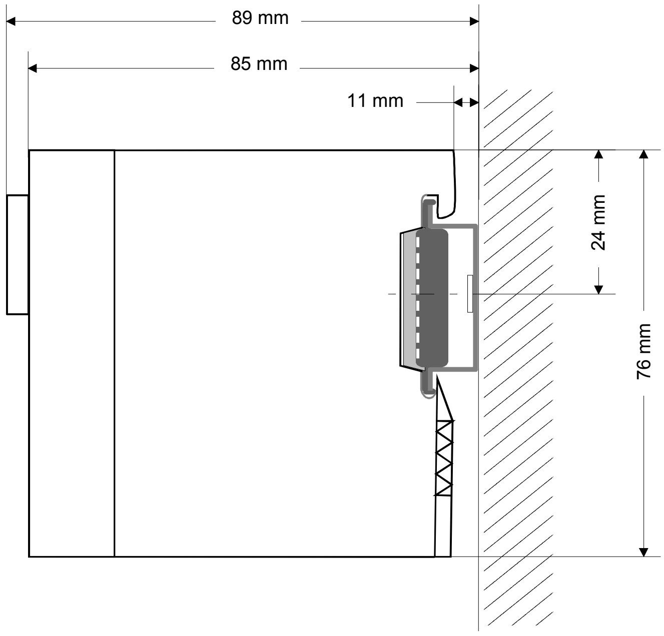


Installed and wired dimensions

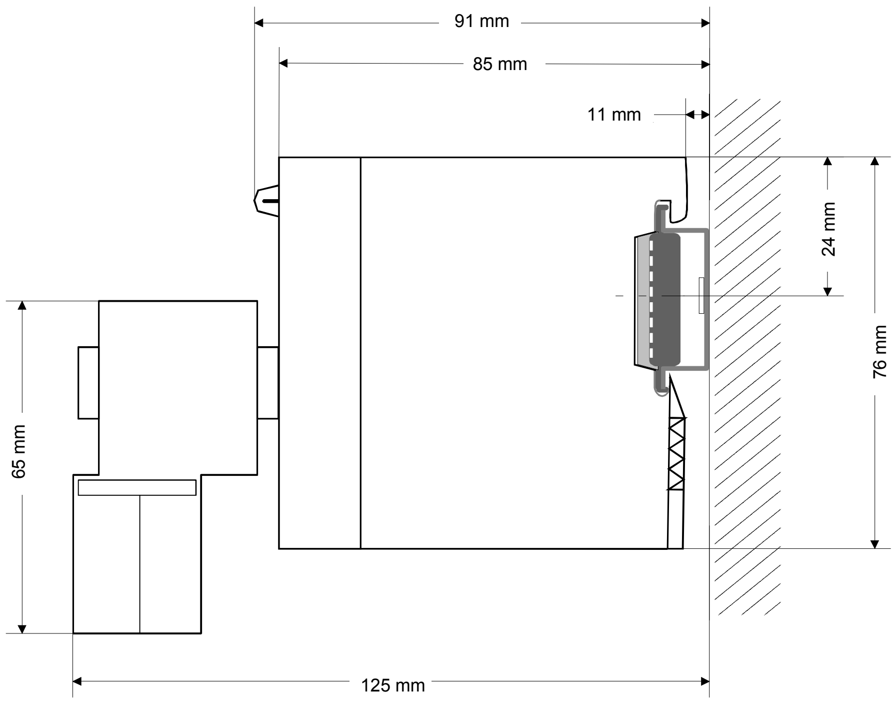
In- / Output modules



Function modules



CPUs here with EasyConn from VIPA



Installation guidelines

General

The installation guidelines contain information on the proper assembly of System 200V. Here we describe possible ways of interference that may disturb the controlling system and how you have to approach shielding and screening issues to ensure the electromagnetic compatibility (EMC).

What is EMC?

The term "electromagnetic compatibility" (EMC) refers to the ability of an electrical device to operate properly in an electromagnetic environment without interference from the environment or without the device causing illegal interference to the environment.

All System 200V components were developed for applications in harsh industrial environments and they comply with EMC requirements to a large degree. In spite of this you should implement an EMC strategy before installing any components which should include any possible source of interference.

Possible sources for disturbances

Electromagnetic interference can enter your system in many different ways:

- Fields
- I/O signal lines
- Bus system
- Power supply
- Protective conductor

Interference is coupled into your system in different ways, depending in the propagation medium (conducted or not) and the distance to the source of the interference.
We differentiate between:

- galvanic coupling
- capacitive coupling
- inductive coupling
- radiated power coupling

The most important rules for ensuring EMC

In many cases, adherence to a set of very elementary rules is sufficient to ensure EMC. For this reason we wish to advise you to heed the following rules when you are installing your controllers.

- During the installation of your components you have to ensure that any inactive metal components are grounded via a proper large-surface earth.
  - Install a central connection between the chassis ground and the earthing/protection system.
  - Interconnect any inactive metal components via low-impedance conductors with a large cross-sectional area.
  - Avoid aluminum components. Aluminum oxidizes easily and is therefore not suitable for grounding purposes.
- Ensure that wiring is routed properly during installation.
  - Divide the cabling into different types of cable. (Heavy current, power supply, signal and data lines).
  - Install heavy current lines and signal or data lines in separate channeling or cabling trusses.
  - Install signaling and data lines as close as possible to any metallic ground surfaces (e.g. frames, metal rails, sheet metal).
- Ensure that the screening of lines is grounded properly.
  - Data lines must be screened.
  - Analog lines must be screened. Where low-amplitude signals are transferred, it may be advisable to connect the screen on one side of the cable only.
  - Attach the screening of cables to the ground rail by means of large surface connectors located as close as possible to the point of entry. Clamp cables mechanically by means of cable clamps.
  - Ensure that the ground rail has a low-impedance connection to the cabinet/cubicle.
  - Use only metallic or metallized covers for the plugs of screened data lines.
- In critical cases you should implement special EMC measures.
  - Connect snubber networks to all inductive loads that are controlled by System 200V modules.
  - Use incandescent lamps for illumination purposes inside cabinets or cubicles, do not use fluorescent lamps.
- Create a single reference potential and ensure that all electrical equipment is grounded wherever possible.
  - Ensure that earthing measures are implemented effectively. The controllers are earthed to provide protection and for functional reasons.
  - Provide a star-shaped connection between the plant, cabinets/cubicles of the System 200V and the earthing/protection system. In this way you avoid ground loops.
  - Where potential differences exist you must install sufficiently large equipotential bonding conductors between the different parts of the plant.

Screening of cables

The screening of cables reduces the influence of electrical, magnetic or electromagnetic fields; we talk of attenuation.

The earthing rail that is connected conductively to the cabinet diverts interfering currents from screen conductors to ground. It is essential that the connection to the protective conductor is of low-impedance as the interfering currents could otherwise become a source of trouble in themselves.

The following should be noted when cables are screened:

- Use cables with braided screens wherever possible.
- The coverage of the screen should exceed 80%.
- Screens should always be grounded at both ends of cables. High frequency interference can only be suppressed by grounding cables on both ends.

  Grounding at one end may become necessary under exceptional circumstances. However, this only provides attenuation to low frequency interference. One-sided earthing may be of advantage where:
  - it is not possible to install equipotential bonding conductors.
  - analog signals (in the mV or µA range) are transferred.
  - foil-type shields (static shields) are used.
- Always use metallic or metallized covers for the plugs on data lines for serial links. Connect the screen of the data line to the cover. Do **not** connect the screen to PIN 1 of the plug!
- In a stationary environment it is recommended that the insulation is stripped from the screened cable interruption-free and to attach the screen to the screening/protective ground rail.
- Connect screening braids by means of metallic cable clamps. These clamps need a good electrical and large surface contact with the screen.
- Attach the screen of a cable to the grounding rail directly where the cable enters the cabinet/cubicle. Continue the screen right up to the System 200V module but do **not** connect the screen to ground at this point!

Please heed the following when you assemble the system!

Where potential differences exist between earthing connections it is possible that an equalizing current could be established where the screen of a cable is connected at both ends.

Remedy: install equipotential bonding conductors

Chapter 3 Profibus DP

Overview

This chapter contains a description of Profibus applications of the System 200V. A short introduction and presentation of the system is followed by the project design and configuration of the Profibus master and slave modules that are available from VIPA. The chapter concludes with a number of communication examples and the technical data.

The following text describes:

- System overview of the Profibus modules from VIPA
- The principles of Profibus DP with DP-V0 / DP-V1
- Structure and project engineering of the Profibus masters IM 208DP and Profibus slaves IM 253DP
- Sample projects and technical data

Content

Topic **Page**

Chapter 3 Profibus DP ... 3-1
System overview ... 3-2
Basics ... 3-5
IM 208DP - Master - Structure ... 3-13
IM 208DP - Master - Deployment at CPU 21x ... 3-17
IM 208DP - Master - Project engineering ... 3-18
IM 208DP - Master - Slave operating mode ... 3-28
IM 208DP - Master - Overall reset ... 3-32
IM 208DP - Master - Firmware update ... 3-33
IM 253-1DPx0 - DP-V0 slave - Structure ... 3-35
IM 253-2DP20 - DP-V0 slave with DO 24xDC 24V - Structure ... 3-38
IM 253-2DP50 - DP-V0 slave (redundant) - Structure ... 3-42
IM 253-xDPx0 - DP-V0 slave - Block diagram ... 3-45
IM 253-xDPx0 - DP-V0 Slave - Project engineering ... 3-46
IM 253-xDPx0 - DP-V0 slave - Parameters ... 3-48
IM 253-xDPx0 - DP-V0 slave - Diagnostic functions ... 3-49
IM 253-xDPx1 - DP-V1 slave - Structure ... 3-56
IM 253-xDPx1 - DP-V1 slave - Block diagram ... 3-60
IM 253-xDPx1 - DP-V1 slave - Project engineering ... 3-61
IM 253-xDPx1 - DP-V1 slave - Parameters ... 3-63
IM 253-xDPx1 - DP-V1 slave - Diagnostic functions ... 3-68
IM 253-xDPx1 - DP-V1 slave - Firmware update ... 3-76
IM 253-xDPx1 - DP-V1 slave - I&M data ... 3-77
Installation guidelines ... 3-79
Commissioning ... 3-89
Using the diagnostic LEDs ... 3-90
Sample projects for Profibus communication ... 3-91
Technical data ... 3-99

System overview

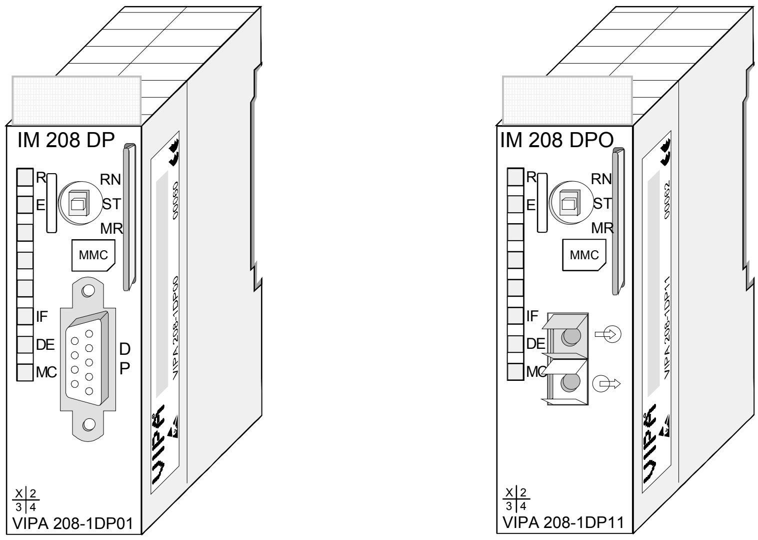
System 200V Profibus DP modules

Most System 200V Profibus modules from VIPA are available with RS485 as well as a FO connector. The following groups of Profibus modules are available at present:

- Profibus DP master
- Profibus DP slave with DP-V0 / DP-V1
- Profibus DP slave combination modules
- CPU 21xDP - CPU 21x for S7 from Siemens with integrated Profibus DP slave (refer to manual HB97_CPU)
- CPU 24xDP - CPU 24x for S5 from Siemens with integrated Profibus DP slave (refer to manual HB99)

Profibus DP master

- Profibus DP master, class 1
- Project design using WinNCS from VIPA or Siemens SIMATIC Manager
- Project-related data is saved in the internal Flash-ROM or stored on a MMC.

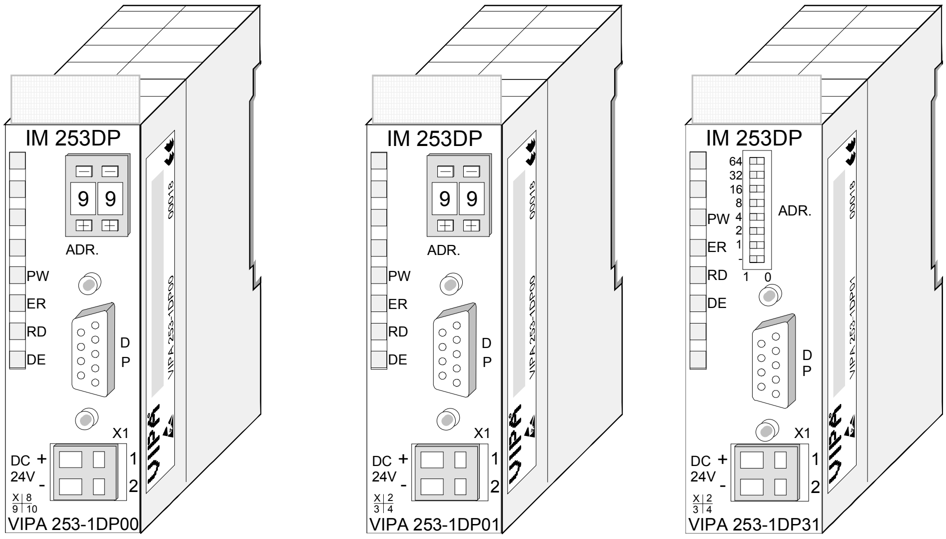


Order data DP master

Type	Order number	Description	Page
IM 208DP	VIPA 208-1DP01	Profibus DP master with RS485	3-13
IM 208DPO	VIPA 208-1DP11	Profibus DP master with FO connector	

Profibus DP slaves

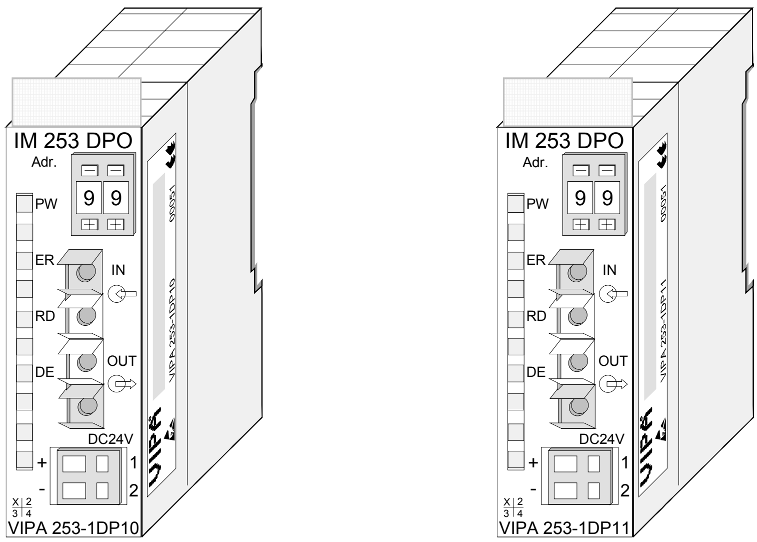
- Version with RS485 interface or fiber optic connectors
- Version with DP-V1 interface
- Online diagnostic protocol



Order data

Type	Order number	Description	Page
IM 253DP	VIPA 253-1DP00	Profibus DP-V0 slave	3-35
IM 253DP	VIPA 253-1DP01	Profibus DP-V0/V1 slave	3-56
IM 253DP	VIPA 253-1DP31	Profibus DP-V0/V1 slave - ECO	3-56

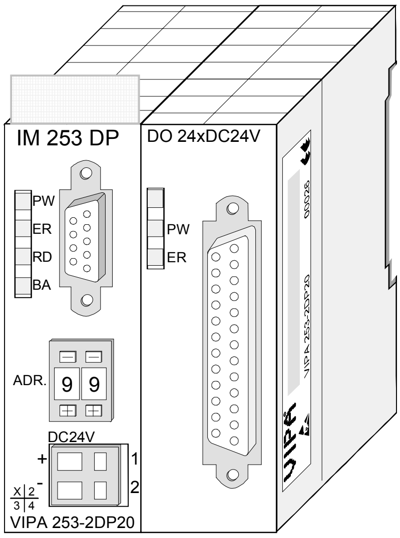
Profibus-DP slaves with FO connector



Order data

Type	Order number	Description	Page
IM 253DPO	VIPA 253-1DP10	Profibus DP-V0 slave with FO connector	3-35
IM 253DPO	VIPA 253-1DP11	Profibus DP-V0/V1 slave with FO connector	3-56

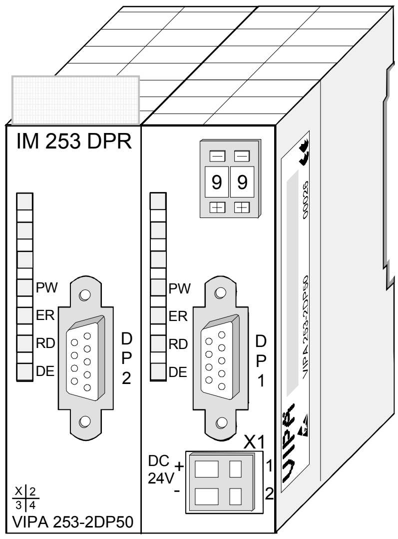
Profibus DP slave (combi modules)



Order data

Type	Order number	Description	Page
IM 253DP DO 24xDC 24V	VIPA 253-2DP20	Profibus DP-V0 slave with 24 DO	3-38

Profibus DPR slave (redundant)



Order data

Type	Order number	Description	Page
IM 253DPR	VIPA 253-2DP50	Profibus DP-V0 slave 2 channel redundant	3-42

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-specific 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 cyclical data communication between bus master and slave systems.

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 alarm 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 alarms. Examples for different types of alarms are status alarm, update alarm and a manufacturer-specific alarm.

Please note in operating the DP V1 functionality that your DP master supports DP-V1 as well. For this you find details in the documentation to 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 (IM 208DP or IM 208DPO) 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 (IM 253DP, IM 253DPO and the CPU 24xDP, CPU 21xDP) are modular slave devices that transfer data between the System 200V 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. VIPA Profibus DP master guarantees (from Firmware version V3.00) that the consistency will cater for the required length.

Restrictions

- Max. 125 DP slaves at one DP master - max. 32 slaves/segment
- Max. 16 DPO slaves at one DPO master at 1.5MBaud
- 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 max. distance for FO cables between two stations is 300m (at HCS-FO) and 50m (at POF-FO).
- The maximum baud rate is 12MBaud.
- The Profibus address of operational modules must never be changed.

Diagnostic

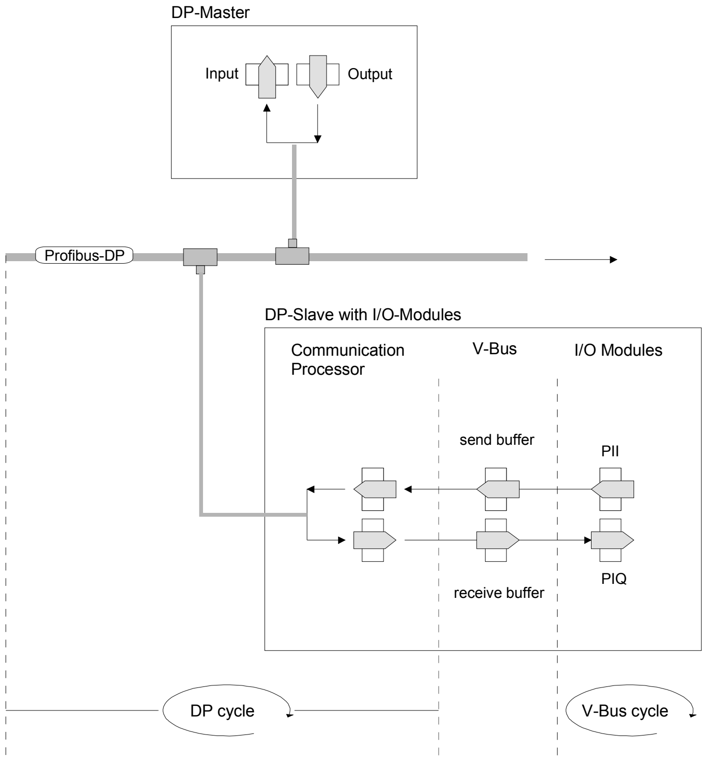
Profibus DP provides an extensive set of diagnostic functions for fast error localization. Diagnostic messages are transferred via the bus and collected by the master.

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

V-bus cycle

A V-bus cycle (V-Bus = VIPA backplane bus) 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 ”buffer send” and the contents of the ”buffer receive” 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 ”buffer receive” and the data in the ”buffer send” 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 V-bus cycle.

V-bus cycle ≤ DP cycle

To ensure that the data transfer is synchronized the V-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 (VIPA_0550.gsd).

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

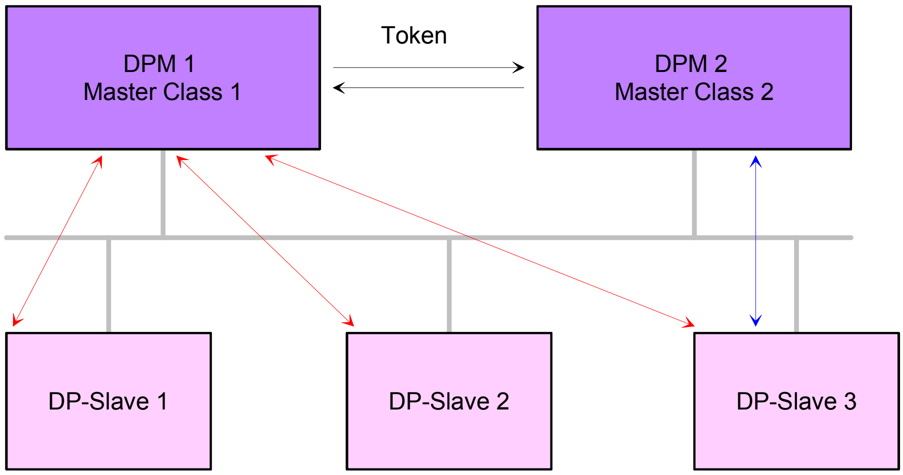
Note!

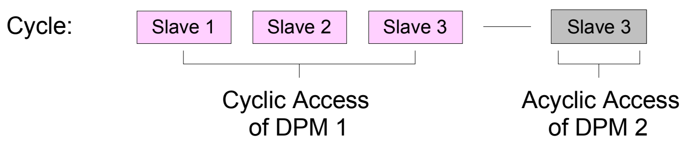
Starting with release version 6, the RUN-LED of a DP-V0 slave extinguishes as soon as the V-Bus cycle lasts longer than the DP cycle. This function is de-activated at the employment of a DP-V1 slave as DP-V0.

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 alarm 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).

Services Acyclic data communication

Additional available services are shown in following table.

More detailed information to the services and the DP-V0/1 communication - principles is to find in the Profibus norm IEC 61158.

DPM 1 (MSAC-C1)

Services for Acyclic data communication between the DPM 1 and Slaves	
Read	The master reads a data block from the slave.
Write	The master writes a data block to the slave.
Alarm	An alarm is transmitted from the slave to the master, which explicitly acknowledges receipt. The slave can only send a new alarm message after it has received this acknowledgement; this prevents any alarms being overwritten.
Alarm_Acknowledge	The master acknowledges receipt of an alarm 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 (MSAC-C2)

Services for Acyclic data communication between the 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 available in the Slave.	

Data transfer medium

Profibus employs screened twisted pair cable on the basis of the RS485 interfaces or a duplex fiber optic link (FO). The data transfer rate of both systems is limited to a max. of 12MBaud.

For details please refer to the "Assembly and installation guidelines".

Electrical system based on RS485

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 VIPA 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.

Optical system using fiber optic data links

The fiber optic system employs pulses of monochromatic light. The optical waveguide is not susceptible to external electrical interference. Fiber optic systems have a linear structure. Each device requires two lines, a transmit and a receive line. It is not necessary to provide a terminator at the last device.

Due to the linear structure of the FO data link, it is not possible to install or remove stations without interruption to data communication.

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 126. The address of the VIPA Profibus coupler is set by the addressing switch located on the front of the module.

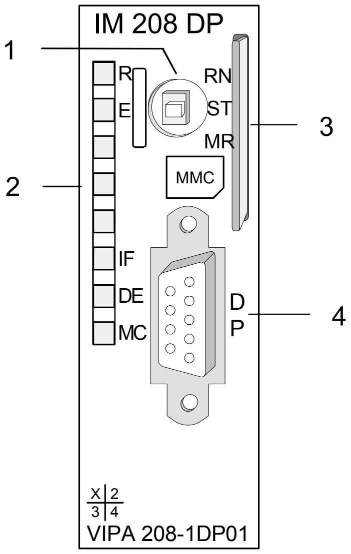
You assign the address to the VIPA Profibus master during the configuration phase.

IM 208DP - Master - Structure

Properties

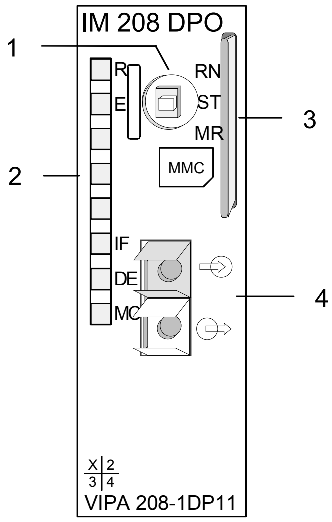
- Class 1 Profibus DP master
- 125 DP slaves (16 at DPO) connectable to one DP master
- Inserts the data areas of the slaves located on the V-bus into the addressing area of the CPU
- Project engineering by means of VIPAs WinNCS or Siemens SIMATIC Manager or ComProfibus
- Diagnostic facilities

Front view IM 208DP



[1] Operating mode switch RUN/STOP
[2] LED status indicators
[3] Slot for memory card
[4] RS485 interface

Front view IM 208DPO



[1] Operating mode switch RUN/STOP
[2] LED status indicators
[3] Slot for memory card
[4] FO interface

Components

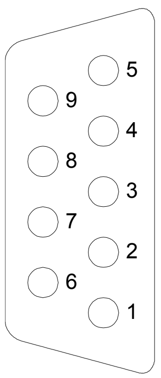
LEDs

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
R	green	If R is the only LED that is on, then the master status is RUN. The slaves are being accessed and the outputs are 0 ("Clear" state). If both R+DE are on the status of the Master is "operate". It is communicating with the slaves. Blinks 3 times: Transfer from MMC to Flash-ROM without error.
E	red	On at slave failure (ERROR). Blinks 3 times: Transfer from MMC to Flash-ROM without error.
IF	red	Initialization error for bad parameterization
DE	green	DE (Data exchange) indicates Profibus communication activity. At the DP master with the order no. 208-1DP01 this LED is yellow.
MC	yellow	Blinks at reading the parameters from MMC. Is on at wrong parameterization.

RS485 interface (at IM 208DP)

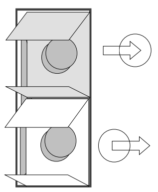
The VIPA Profibus master is connected to your Profibus network via the 9pin socket. The following figure shows the assignment of the individual pins:



Pin	Assignment
1	shield
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.

FOL interface (at IM 208DPO)

The IM 208DPO is connected to Profibus by a FOL (**f**iber **o**ptic **l**ink) interface. The layout of this interface is shown below:



Connect to receive line (from slave)

Connect to send line (to slave)

Power supply

The Profibus master receives power via the backplane bus.

Operating mode selector

The operating mode selector is used to select the operating modes STOP (ST), RUN (RN) and MEMORY (MR).

The master will change to RUN mode if the operating mode selector is set to RN and parameters are acceptable.

When the operating mode switch is set to ST, the master will change to STOP mode. In this mode all communication is terminated, the outputs of the allocated slaves will be set to 0 and the master issues an alarm to the controlling system.

This chapter contains under ”Operating modes” a detailed explanation of the change between RUN and STOP mode.

In position MR you may activate:

- the data transfer from MMC into Flash-ROM
- a serial mode for deploying the VIPA Green Cable
- Overall reset of the DP master

More details to theses possibilities can be found below.

MMC as external storage medium

The VIPA MMC (**mem**ory **c**ard) is employed as an external storage medium. The MMC is available from VIPA with the order no.: VIPA 953-0KX00. You can get a external MMC reading device from VIPA (Order no: VIPA 950-0AD00) for your PC. Hereby you can read and write MMC by using your PC.

You initiate the transfer of project data from the MMC into the master by setting the operating mode selector into position MR. For details, please refer to the section on "Transferring a project" below.

Operating modes

Power ON

The DP master is powered on. The master will change automatically to RUN mode when the operating mode lever is in position RUN and the parameters are valid.

STOP

In STOP mode the outputs of the allocated slaves will be set to 0 if the parameters are valid. Although no communication will take place, the master will remain active on the bus using current bus parameters and occupying the allocated bus address. To release the address the Profibus plug must be removed from the DP master.

STOP → RUN

In the RN position the master will re-boot. Here an existing hardware configuration is not deleted.

At a STOP → RUN transition the communication link to the slaves is established. At this time only the R-LED is on. Once communication has been established DP master changes to RUN mode. The DP master interface shows this status by means of the LEDs R and DE.

With incorrect parameters the DP master remains in STOP state and shows an error in parameterization by means of the IF-LED. The DP master will then be active on the bus with the following default bus parameters:

Default bus parameters: Address: 2, Communication rate: 1.5MBaud.

Note!

With DP master firmware versions older than V 5.0.0 with a STOP-RUN transition of the DP master a just existing hardware configuration is deleted and a probably in the flash ROM stored project is used.

To retransmit the hardware configuration a power cycle of the CPU is necessary.

RUN

In RUN mode the R- and DE-LEDs are on. In this condition data transfer can take place. If an error occurs, e.g. slave failure, the DP master will indicate the event by means of the E-LED and it will issue an alarm to the system on the next higher level.

RUN → STOP

The master is placed in STOP mode. It terminates communication and all outputs are set to 0. An alarm is issued to the system on the next higher level.

IM 208DP - Master - Deployment at CPU 21x

Communication

Via the IM 208 master modules you may connect up to 125 Profibus DP slaves (up to 16 at DPO) to one System 200V CPU.

The master communicates with the slaves and transfers the data areas via the backplane bus into the address area of the CPU. There may occur a maximum of 1024Byte input and 1024Byte output data.

With firmware versions < V3.0.0 there are only 256Byte available for input and output data.

With every boot procedure of the CPU, this fetches the I/O mapping data from all masters.

Alarm processing

The alarm processing is activated, i.e. a IM 208 error message may initialize the following alarms, causing the CPU to call the according OBs:

- Process alarm: OB40
- Diagnostic alarm: OB82
- Slave failure: OB86

As soon as the BASP signal (i.e. "**B**efehls**a**usgabe**sp**erre" = command output lock) comes from the CPU, the IM 208 sets the outputs of the connected periphery to zero.

Note!

After a slave failure, the process image of the inputs is in the same state than before the failure.

Preconditions

At deployment of the IM 208 Profibus DP master, please make sure that this has a firmware version V3.0.0 or higher; otherwise it is not deployable with a CPU 21x with firmware version V3.0.0 or higher.

The according firmware version is to find on the label at the backside of the module.

Having questions to the firmware update, please call the VIPA support (support@vipa.de).

More detailed descriptions to the inclusion into your CPU are to find in the documentation of your CPU.

IM 208DP - Master - Project engineering

General

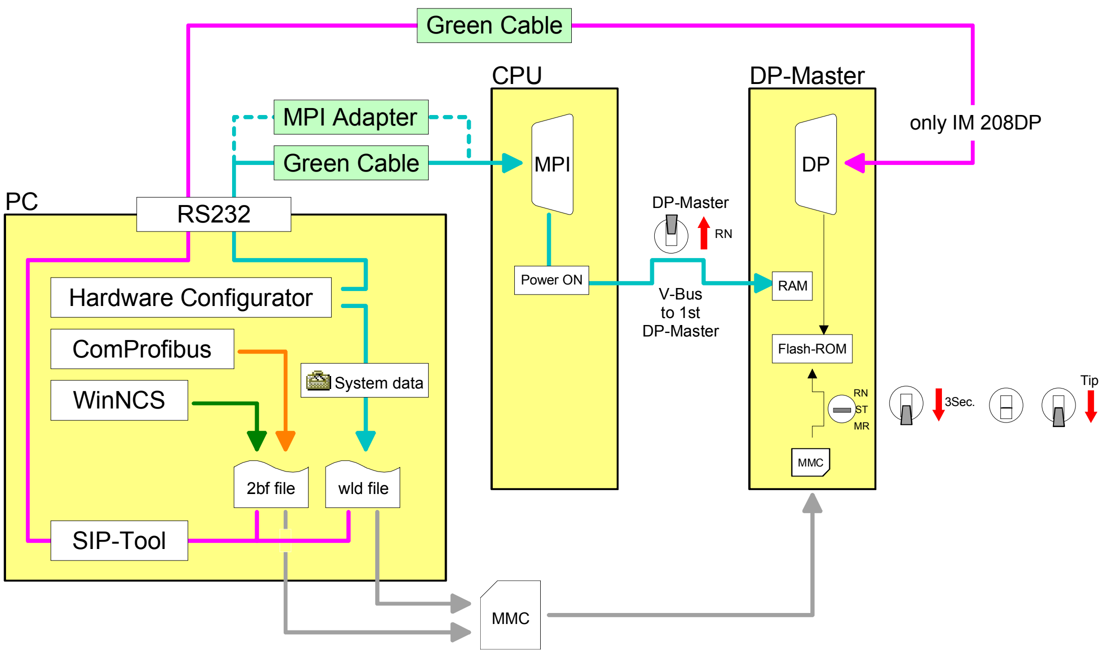
There are the following possibilities for project engineering:

- Project engineering of the 1. DP master in the System (CPU 21xDPM, IM 208)

  Project engineering in the hardware configurator from Siemens and transfer via the system data into the CPU. At CPU start-up the DP master is configured by the CPU.
- Project engineering of further DP master in the system (only IM 208)

  Project engineering in the hardware configurator from Siemens and export as wld-file. The file is transferred by MMC respectively SIP-Tool and Green Cable to the DP master. With a overall reset sequence the project is transferred to the Flash ROM of the DP master.
- Project engineering with WinNCS respectively ComProfibus

  Project engineering with VIPA WinNCS respectively ComProfibus from Siemens and export as 2bf-file. The file is transferred by MMC respectively SIP-Tool and Green Cable to the DP master. With a overall reset sequence the project is transferred to the Flash ROM of the DP master.



Required firmware versions

DP master and CPU should have a firmware version V3.0.0 or higher, otherwise the DP master may not be deployed at the CPU 21x.

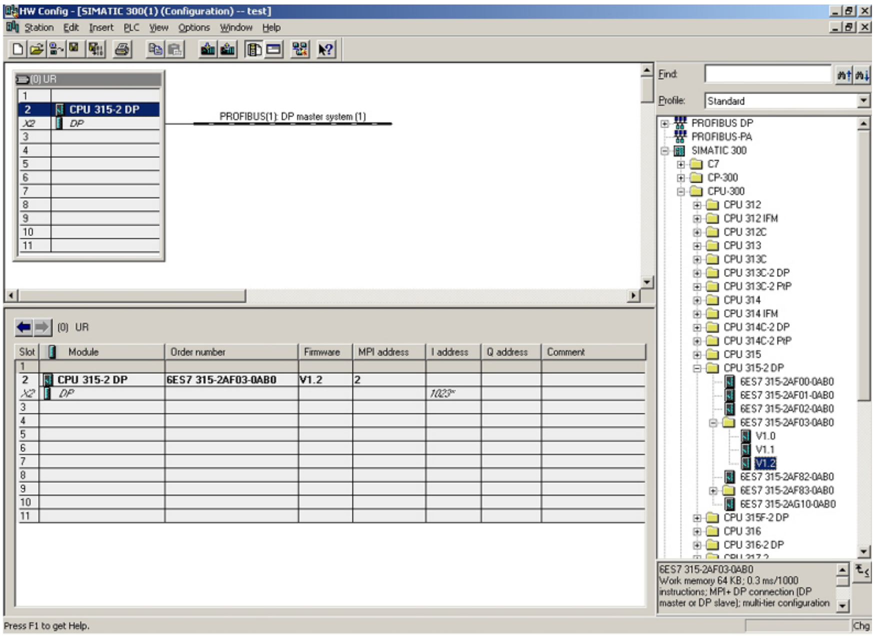
The according firmware version is to find on the label at the backside of each module.

Firmware version

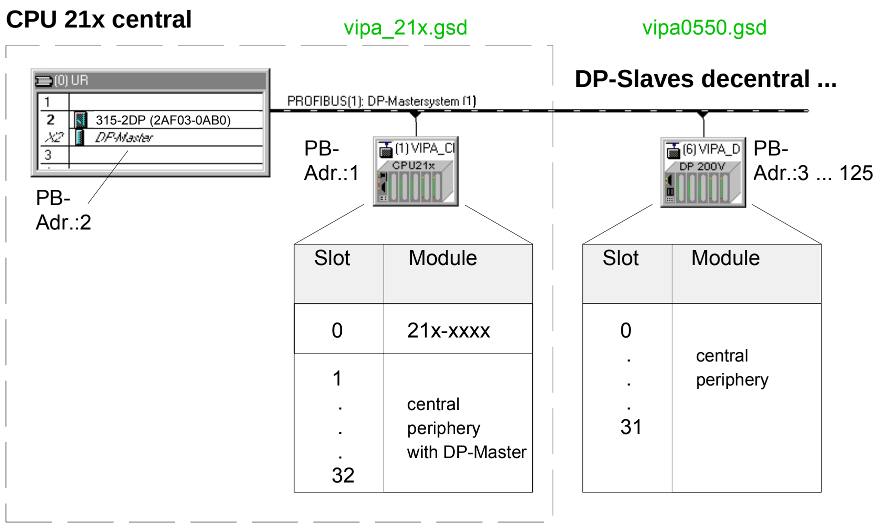
DP master	CPU	Properties
V3.0.0	V3.0.0	1024Byte in- and output data
V3.0.4	V3.0.0	Project engineering via wld-file
V3.0.6	V3.3.0	Project engineering as HW configuration via MPI
V3.0.6	---------	Overall reset of the DP master

Project engineering of the 1. DP master in the system

In the hardware configurator from Siemens your PLC system is projected together with the DP master. This "hardware configuration" is to be transferred via MPI into the CPU. At PowerON, the configuration data is transferred to the DP master.

1.	Create a new project System 300 and add a profile rail from the hardware catalog.
2.	Insert the CPU 315-2DP. This may be found with Profibus master in the hardware catalog at: *Simatic300>CPU-300>CPU315-2DP* **(6ES7 315-2AF03-0AB0 V1.2)**
3.	Assign a Profibus address 2 or higher to your master.
4.	Click at DP, select the operating mode "DP master" at *Object properties* and confirm your entry with OK.
5.	With a right-click on "DP", a context menu opens. Choose "Insert master system". Create a new Profibus subnet via NEW. The following picture shows the new master system: 

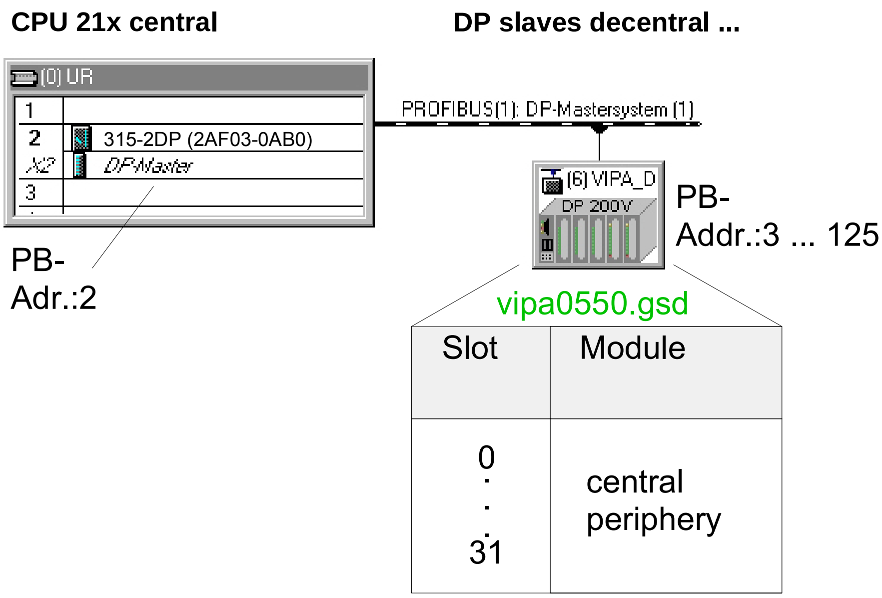
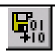
Note!
At DP master firmware older than V 5.0.0 the operating mode switch of the DP master should be in RN position. Otherwise, a STOP-RUN switch causes the master to reboot and the project is deleted.

<table>
<tr><td>6.</td><td>To be compatible to the Siemens SIMATIC Manager, the System 200V CPU has to be included explicitly.
For this, you add a System "VIPA_CPU21x" to the subnet. This system is to find in the hardware catalog at:
PROFIBUS DP > Additional field devices > IO >
VIPA_System_200V > VIPA_CPU21x.
Assign the Profibus address 1 to this slave.
Set the according CPU 21x from VIPA at slot 0 by choosing it in the hardware catalog at VIPA_CPU21x.
Slot 0 is mandatory!</td></tr>
<tr><td>7.</td><td>For including the modules connected to the VIPA-Bus, you drag and drop the according System 200V modules from the hardware catalog at VIPA_CPU21x to the slots following the CPU. Start with slot 1. The same is to do for the DP master (place holder).</td></tr>
<tr><td>8.</td><td>For projecting DP slaves connected to the DP master select the VIPA_DP200V_2 system. Select the according Profibus system in the hardware catalog and drag it to the DP master subnet.
Assign an address > 2 to the slave.
Set the according modules at slot 0 by choosing it in the hardware catalog at VIPA_DP200V_2.

</td></tr>
<tr><td>9.</td><td>Click on (save and translate).</td></tr>
</table>

How the project is transferred via MPI to the CPU is shown at the following pages at "Transfer variants"

Project engineering of further DP master in the system

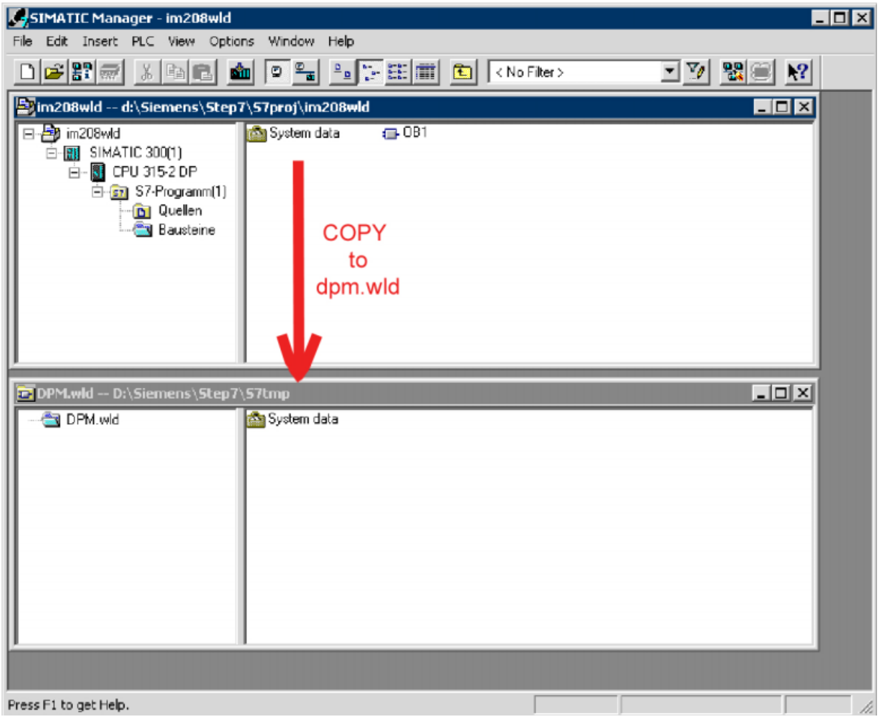
If there are further more IM 208 DP master in one system, for each a project is to be projected. This project may be transferred to the corresponding DP master either by MMC or by Green Cable. The Project is permanently stored in the Flash ROM of the DP master by means of an overall reset sequence.

1.	Create a new project System 300 for the corresponding DP master and add a profile rail from the hardware catalog.
2.	Insert the CPU 315-2DP. This may be found with Profibus master in the hardware catalog at: *Simatic300>CPU-300>CPU315-2DP* **(6ES7 315-2AF03-0AB0 V1.2)**
3.	Assign a Profibus address 2 or higher to your master.
4.	Click at DP, select the operating mode "DP master" at *Object properties* and confirm your entry with OK.
5.	With a right-click on "DP", a context menu opens. Choose "Insert master system". Create a new Profibus subnet via NEW.
6.	To configure DP salves, which are connected to the DP master, select the VIPA_DP200V_2 system. Select the according Profibus system in the hardware catalog and drag it to the DP master subnet. Assign an address > 2 to the slave. Set the according modules at slot 0 by choosing it in the hardware catalog at VIPA_DP200V_2. 
7.	Click on  (save and translate).

Export dpm.wld

Export your project to a MMC by creating a wld-file. The MMC is then plugged in the according DP master. The Project is permanently stored in the Flash ROM of the DP master by means of an overall reset sequence.

After the transfer, you may release the MMC again. This allows you to configure several masters at the same backplane bus with one MMC.

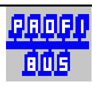
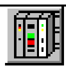
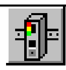
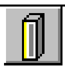
8.	Create with **File** > *Memory Card File* > *New ...* a new wld-file. This need to have the file name **dpm.wld** to be recognized from the Profibus master. → This file is additionally shown to the configuration window.
9.	Go into your project into the directory *modules* and copy the directory " System data" into the created dpm.wld-file.  If an already existing "System data" directory shall be overwritten, you first have to delete that.

How the dpm.wld-file is transferred to the corresponding DP master is shown at the following pages at "Transfer variants".

Configuration with WinNCS respectively ComProfibus

The Profibus master may be easily configured by means of the VIPA WinNCS configuration tool. You may export your project as 2bf-file on a MMC res. transfer it via SIP-Tool into the DP master (only at IM 208DP possible).

The WinNCS configuration procedure is outlined below. For more detailed information see the manual HB91 for WinNCS.

1.	Start WinNCS and create a new project file for the ”Profibus” function by clicking on **File** > *create/open*.
2.	If you have not yet done so, use  to insert a **Profibus function group** into the network window and click [Accept] in the parameter box.
3.	Use  to insert a **Profibus host/master** into the network window and specify the Profibus address of your master in the parameter window.
4.	Insert a **Profibus slave** into the network window by means of . Enter the Profibus address, the family "I/O" and the station type "*VIPA_DP200V_2*" into the parameter window and click [Accept].
5.	Use  to define the configuration of every peripheral module that is connected to the corresponding slave via the backplane bus. You can select automatic addressing for the periphery by clicking [Auto] and display allocated addresses by means of [MAP]. **Please take care that the automatic address allocation does not cause conflicts with the local periphery!** For intelligent modules like the CP 240 the configurable parameters will be displayed.
6.	When you have configured all the slaves with the respective periphery, the bus parameters for Profibus must be calculated. Select the Profibus function group in the network window. In the parameter window click on the "Bus parameter" tab. Select the required baud rate and click [calculate]. The bus parameters will be calculated - [Accept] these values. The bus parameters must be re-calculated with every change to the set of modules!
7.	Activate the master level in the network window and export your project into the file dpm.2bf.
8.	Transfer the dpm.2bf-file into your IM208 master (see "transferring a project ") .

Note!
For the IM 208 DP master is configured like the IM 308-C from Siemens, you may configure the VIPA module also as IM 308-C under "ComProfibus" from Siemens and export it as 2bf-file.

Transfer variants

There are the following possibilities to transfer the wld- respectively 2bf-file into the DP master:

- Transfer via MPI into the CPU (only for the 1st DP master at system)
- Transfer via MMC
- Transfer via Green Cable and SIP-Tool

Transfer via MPI into the CPU

Starting with firmware V 3.0.6 of the DP master and V 3.3.0 of the CPU, your project may be transferred via MPI into the CPU with the following approach.

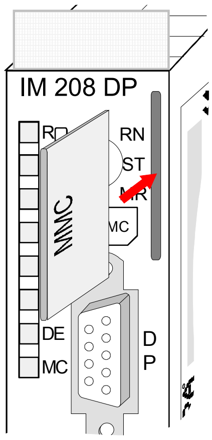
At Power ON the DP master project is transferred to the 1st DP master (IM 208DP or CPU 21xDPM) in the system by the CPU.

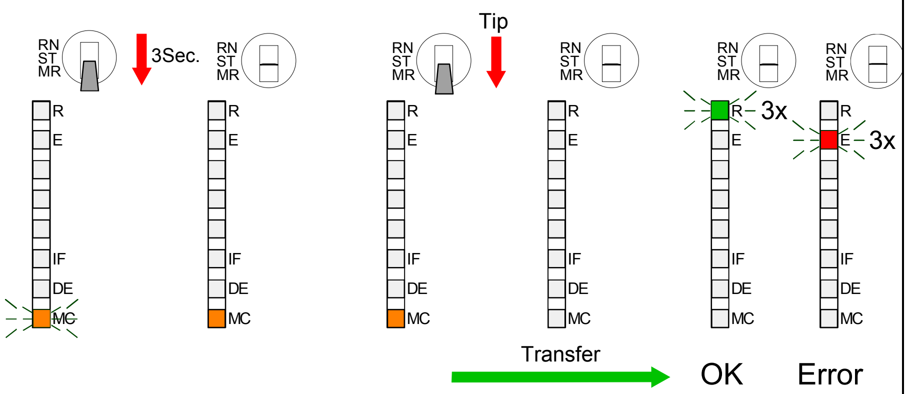
1.	Connect your PG res. your PC via MPI with your CPU. For a serial point-to-point transfer from your PC, you may also use the Green Cable from VIPA. The Green Cable has the order no. VIPA 950-0KB00 and may only be deployed at compatible modules from VIPA. Please regard the instructions to the Green Cable further down!! At deployment of the Green Cable from VIPA, the MPI interface has to be configured (PC Adapter MPI, 38400Baud).
2.	Switch your DP master to RUN.
3.	Switch on the power supply of the CPU.
4.	Transfer your project into the CPU with **PLC** > *Load to module* in the hardware configurator from Siemens. At Power ON the CPU always transferres the Profibus project to the 1st DP master. For additional saving of your project on a MMC, you plug a MMC in the CPU slot and transfer the project via **PLC** > *Copy RAM to ROM.* During write operation, the "MC"-LED at the CPU is blinking. Due to the system, the successful write operation is announced too soon. Please wait until the LED extinguishes.

Note!

At DP master firmware older than V 5.0.0 the operating mode lever of the DP master should be in RN position. Otherwise, a STOP-RUN switch causes the master to reboot and the project is deleted.

Transfer via MMC



<table>
<tr><td>1.</td><td>Transfer the wld- respectively the 2bf-file to the MMC by means of a MMC reading device.</td></tr>
<tr><td>2.</td><td>Plug-in the MMC memory module into your IM 208DP master</td></tr>
<tr><td>3.</td><td>Turn on the power supply for the System 200V.</td></tr>
<tr><td>4.</td><td>Hold the operating mode lever of the Profibus master in position MR until the blinking MC-LED switches to permanent on.</td></tr>
<tr><td>5.</td><td>Release the operating mode lever and tip it once more to MR. → The data is transferred from the MMC into the internal Flash-ROM. During data transfer all LEDs extinguish.
At successful data transfer, the green R-LED blinks 3 times.
At error, the red E-LED blinks 3 times.

</td></tr>
<tr><td>6.</td><td>Now you may release the MMC again.</td></tr>
<tr><td>7.</td><td>Switch the master from STOP to RUN. → The IM 208DP master now starts with the new project in the internal Flash-ROM.
The RUN-LED (R) and DE are on.</td></tr>
</table>

Note!

The project inside the PLC for the 1st Master takes priority over the project downloaded to Flash-ROM of the Master.

If the MMC contains a wld- and a 2bf-file, the wld-file has the priority.

Transfer via Green Cable and VIPA SIP-Tool

The method shown below can only be used at the IM 208DP with RS485-interface. The SIP-Tool is a transfer tool. It is supplied together with WinNCS from VIPA. It allows you to deploy the Green Cable from VIPA to transfer your project as wld- respectively 2bf-file into the master serial via the Profibus interface. The transferred project is stored in the internal Flash-ROM of the DP master.

The Green Cable is a programming and download cable for VIPA CPUs MP^{2}I jack and VIPA fieldbus masters. The Green Cable from VIPA is available under the order no. VIPA 950-0KB00.

The Green Cable allows you to:

- *transfer projects serial*
  Avoiding high hardware needs (MPI transducer, etc.) you may realize a serial point-to-point connection via the Green Cable and the MP^{2}I jack. This allows you to connect components to your VIPA-CPU that are able to communicate serial via a MPI adapter like e.g. a visualization system.
- *execute firmware updates of the CPUs and fieldbus masters*
  Via the Green Cable and an upload application you may update the firmware of all recent VIPA CPUs with MP^{2}I jack and certain fieldbus masters (see Note).

Important notes for the deployment of the Green Cable

Nonobservance of the following notes may cause damages on system components.

For damages caused by nonobservance of the following notes and at improper deployment, VIPA does not take liability!

Note to the application area

The Green Cable may exclusively deployed <u>directly</u> at the concerning jacks of the VIPA components (in between plugs are not permitted). E.g. a MPI cable has to be disconnected if you want to connect a Green Cable.

At this time, the following components support Green Cable:

VIPA CPUs with MP^{2}I jack and fieldbus masters from VIPA.

Note to the lengthening

The lengthening of the Green Cable with another Green Cable res. The combination with further MPI cables is not permitted and causes damages of the connected components!

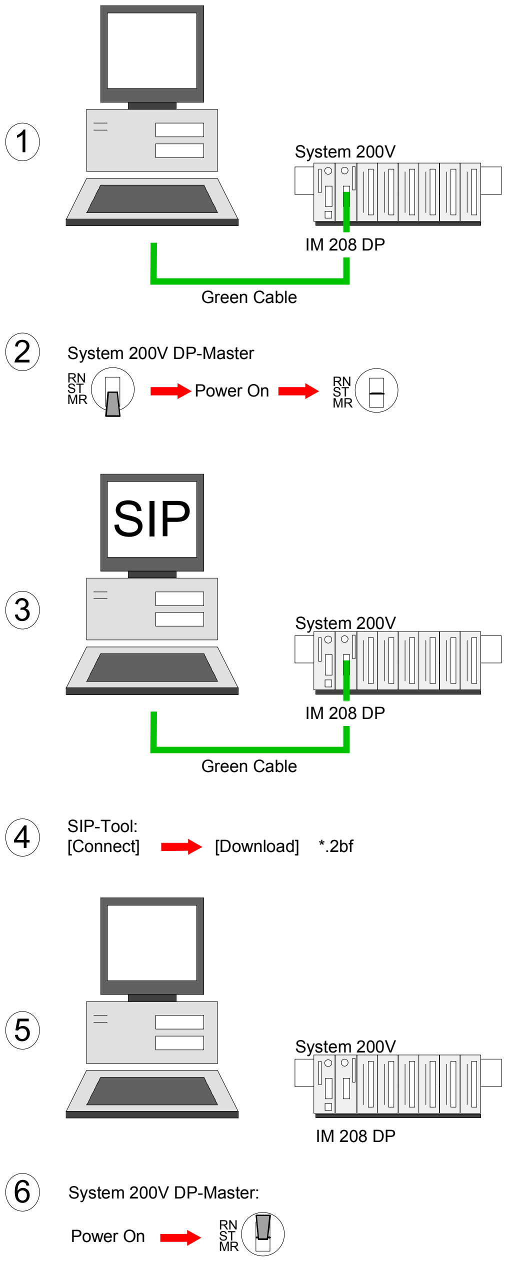
The Green Cable may only be lengthened with a 1:1 cable (all 9 Pins are connected 1:1).

Continued transfer via Green Cable and VIPA SIP-Tool

If you like to project the IM 208 Profibus DP master with the SIP-Tool, this is only possible with DP master firmware V 4.0.0 and higher and with SIP-Tool Version V 1.0.6 and higher.

Either a wld- ore a 2bf-file may be transferred to the DP master by the SIP-Tool. As described before the system data exported by the Siemens SIMATIC manager are stored in the wld-file.

The project may be exported as 2bf-file by VIPA WinNCS respectively by ComProfibus from Siemens.



1.	Disconnect the Profibus plug from the DP master. Connect the "Green Cable" to the serial interface of your PC and to the Profibus interface of the IM 208DP master.
2.	Place and hold the operating mode lever of your master module in position MR and turn on the power supply. Release the lever again. → Now your Profibus master may receive data serial via the Profibus interface.
3.	Turn on your PC and start the SIP tool that is supplied with WinNCS. Select the appropriate COM port and establish a connection by means of [Connect]. When the connection has been established, the SIP tool will display OK in the status line located at the top, otherwise an ERR message will be displayed.
4.	Click [Download], select your dpm.2bf- res. dpm.wld-file and transfer this file into the DP master
5.	Terminate the connection and the SIP tool when the data has been transferred. Disconnect the "Green Cable" from the master.
6.	Turn off the power supply of your master. Connect the master to the Profibus network and turn the power supply on again. Change the operating mode of the master to RUN (RN). → Your IM 208DP Profibus master is now connected to the network with the updated configuration. The configuration data is saved in the internal Flash-ROM.

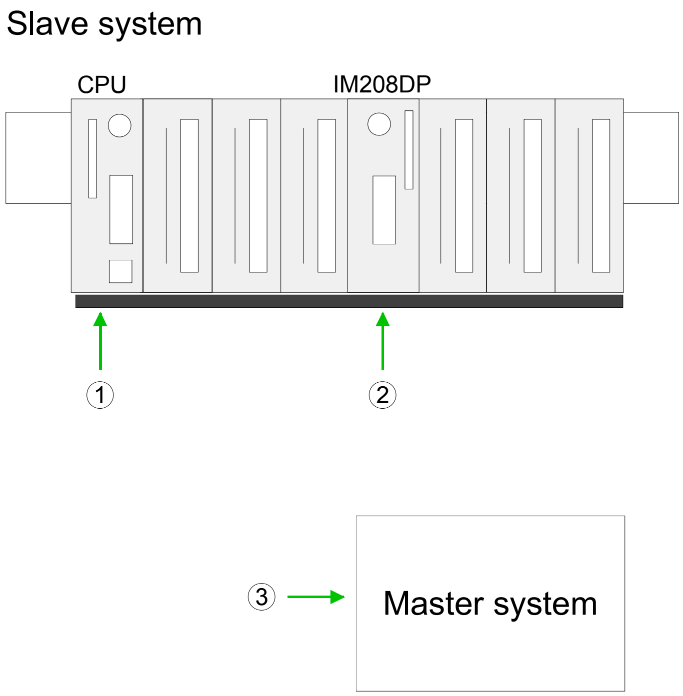
IM 208DP - Master - Slave operating mode

Overview

Starting with CPU firmware 3.72 there is the possibility to use the IM 208DP as DP slave. The Siemens GSD file for the CPU S7-315-2DP is needed for connection to a DP master system.

For hardware technical reasons this functionality is not available for the IM 208DPO master with FO.

For deployment of the IM 208DP as DP slave 3 hardware configurations are needed, which were described in the following:



1. *Hardware configuration System 200V*

   Project engineering Siemens CPU 315-2DP with virtual Profibus slave (address 1) for System 200V. The DP slave contains CPU 21x, I/O periphery and IM 208DP (set parameter *Transfer project to IM 208* to "No"). Project is to be transferred to the CPU 21x.
2. *Hardware configuration IM 208DP*

   Project engineering IM 208DP as Siemens CPU 315-2DP with Slave-Operation of the DP part. Use *Properties* to set Profibus address and I/O area and transfer the project to IM 208DP.
3. *Hardware configuration superordinate master system*

   Project engineering of the superordinate master system. Connection of the IM 208DP (slave) as Siemens CPU S7-315-2DP. Here the installation of the Siemens GSD is necessary. Use *Properties* to set Profibus address (identical to hardware configuration IM 208DP) and I/O areas as "modules". The project is to be transferred to the CPU of the master system.

Hardware configuration System 200V

- Start the Siemens SIMATIC manager.
- Install for CPU 21x project engineering the GSD *VIPA_21x.GSD.*
- Install for linking the DP master the GSD *VIPA04D5.GSD.*
- Create a new project System 300 and add a profile rail from the hardware catalog.
- Insert the CPU 315-2DP. Hardware catalog:
  Simatic300>CPU-300>CPU315-2DP **(6ES7 315-2AF03-0AB0 V1.2)**
- Create a new Profibus sub net and assign a Profibus address 2 or higher to your master.
- Add a System "VIPA_CPU21x" to the subnet. This can be found in the hardware catalog at *PROFIBUS DP* > *Additional field devices* > *IO* > *VIPA_System_200V* > *VIPA_CPU21x*.
- Assign the **Profibus address 1** to this slave.
- Set the according CPU 21x from VIPA at slot 0 by choosing it in the hardware catalog at *VIPA_CPU21x*. **Slot 0 is mandatory!**
- For including the modules connected to the VIPA-Bus, you drag and drop the according System 200V modules from the hardware catalog at *VIPA_CPU21x* to the slot following the CPU. Start with slot 1. The same is to do for the DP master (substitute).
- Set at the IM 208DP properties the parameter *Transmit project to IM 208* to "No". So the CPU can not overwrite the recent local stored DP slave project in the IM 208DP.
- Transfer the project to the CPU.

Hardware configuration System 100V

The employment of the IM208 DP as DP slave in a System100V can exclusively be made by the system expansion. Details for the assembly can be found in HB100 at "expansion an terminal modules". Here the hardware configuration takes place in the same way as with the System200V by means of the following GSD files for the System100V:

- *VIPA_11x.GSD* for project engineering of CPU 11x
- Siemens GSD for linking at DP master

Here also set the parameter *Transmit project to IM 208* to "No". Transfer your project to the CPU 11x.

Continue with the hardware configuration of the IM 208DP and the superordinate master system shown as follows.

Hardware-configuration IM 208DP

- Create a new project System 300 and add a profile rail from the hardware catalog.
- Insert the CPU 315-2DP. Hardware catalog:
  Simatic300 > *CPU-300* > *CPU315-2DP* **(6ES7 315-2AF03-0AB0 V1.2)**
- Open the *Properties* of the *DP* part.
- Choose at *Operating mode* "DP slave".
- Set at *general* a Profibus DP slave address.
- The data transfer areas are set at *configuration*. Please note only the "MS" mode is supported.
- Transfer as shown at "Transfer variants" above the system data to your IM 208DP - not to the CPU! - and set the IM 208DP to RUN.

Note!

The parameters "Input" respectively "Output" at *configuration* always take place from CPU sight.

"Input" refers to the input part and "Output" to the output part of the CPU.

Hardware-configuration superordinate master system

The Siemens GSD is necessary for project engineering at a superordinate master system.

- Start your configuration program with a new project and configure the superordinate Profibus master system.
- Add a DP slave of the station type "S7-315-2DP". This is to be found in the hardware catalog at:
  Profibus DP > *Additional field devices* > *PLC* > *SIMATIC* > *S7-315-2DP*
- Assign the Profibus address to the DP slave that you've parameterized at the slave.
- For the Profibus communication, create the same I/O range that you've parameterized at the slave in form of "modules". Please regard that a slave output area relates to a master input area and vice versa. Also the IO areas must be constantly configured without gaps.
- Save your project and transfer it into the CPU of your master system.

Note!

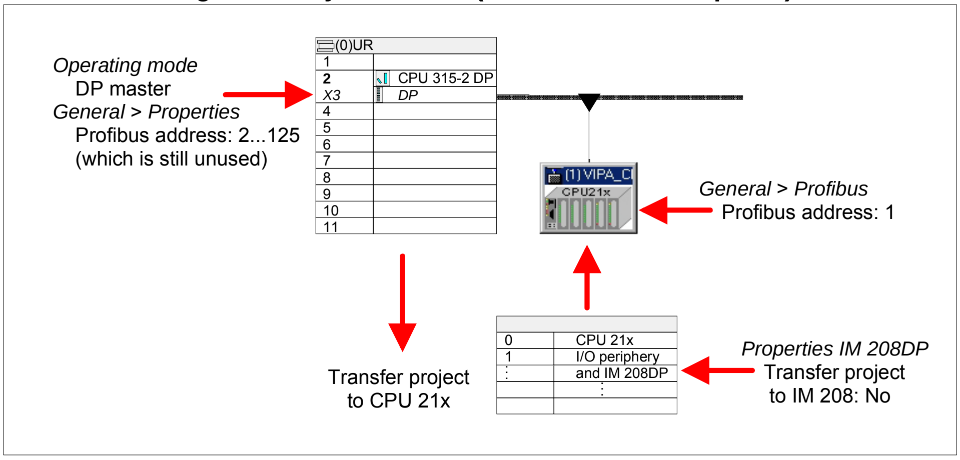
If your DP master system is a System 200V module from VIPA, you may parameterize the directly plugged-in modules by including a "DP200V" slave system.

To enable the VIPA CPU to recognize the project as central system, you have to assign the Profibus address 1 to your slave system!

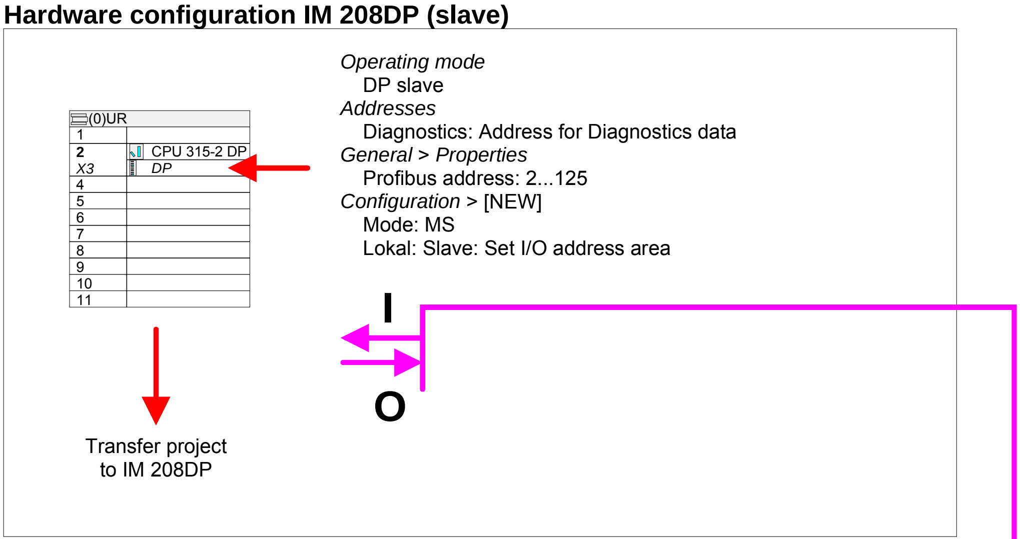
Please take care at deployment of the IM 208 Profibus DP master that this has a firmware version V 3.0 or higher; otherwise this may not be used at a CPU 21x with a firmware version V 3.0 or higher. The firmware versions are on a label at the backside of the modules.

Summary

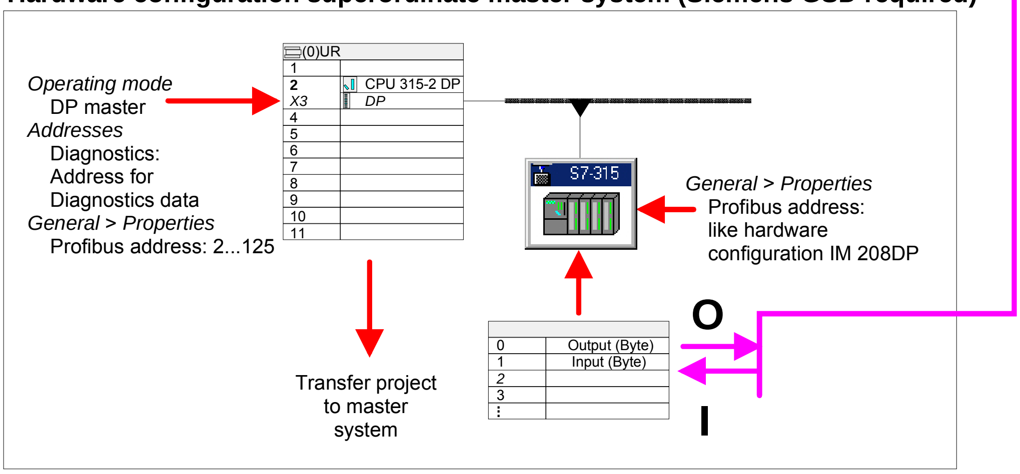
Hardware configuration System 200V (VIPA_21x.GSD required)



Hardware configuration IM 208DP (slave)



Hardware configuration superordinate master system (Siemens GSD required)



Attention!
The length specification of the I/O area of the DP slave must be identical to the bytes configured at the project engineering of the DP master. Otherwise no Profibus communication can take place and a slave loss is replied by the master.

IM 208DP - Master - Overall reset

General

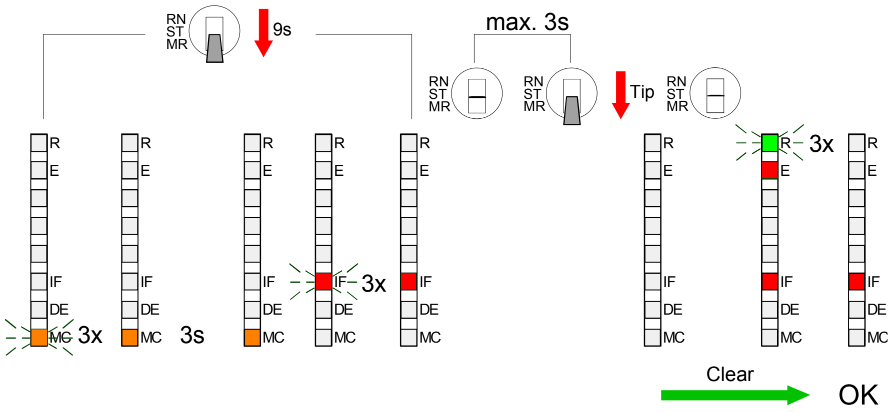
Starting with the firmware version V 3.0.6 of the DP masters, you have the possibility to request an overall reset at the DP master.
An overall reset clears all data in the Flash-ROM.

Execute an overall reset

<table>
<tr><td>1.</td><td>Turn on the power supply of the System 200V.</td></tr>
<tr><td>2.</td><td>Push the operating mode lever of the master module in position MR. Hold it for app. 9s.

→ first, the MC-LED blinks 3 times. For 3s the blinking switches into permanent on. Then, the IF-LED blinks 3 times and switches to permanent on.</td></tr>
<tr><td>3.</td><td>Release the lever and tip it within 3s once more in pos. MR.

→ The content of the Flash-ROM is deleted. The operation has been executed properly when the green R-LED blinks 3 times and the IF-LED is permanent on.



As soon as you switch the master to RUN, this boots and starts with its default parameters at the bus.

Default parameter: Address: 2, Transfer rate: 1.5MBaud</td></tr>
</table>

Project engineering via CPU after power-on to first master

If there is a valid profibus project in the CPU, this is automatically transfered via backplane bus into the RAM of the 1[st] master after Power ON - independet from position of operating mode lever.

IM 208DP - Master - Firmware update

Overview

Starting with CPU firmware version 3.3.3 a MMC inside your CPU can be used to update the firmware of CPU an DP master. The latest 2 firmware versions are to find in the service area at www.vipa.de and at the ftp server at ftp.vipa.de.

For designation the master firmware has the following name convention:

dpmxx.**bin** xx specifies the slot number the DP master is plugged in (Slot: 00 ... 31)

Attention!
When installing a new firmware you have to be extremely careful. Under certain circumstances you may destroy the DP master, for example if the voltage supply is interrupted during transfer or if the firmware file is defective.

In this case, please call the VIPA-Hotline!

Seek firmware version

A label on the rear of the module indicates the firmware version.

Load firmware and transfer it to MMC as *firmware.bin*

- Go to www.vipa.de.
- Click on Service > Download > Firmware Updates.
- Click on "Firmware for Profibus Master System 200V".
- Select the according IM 208 order no. and download the firmware to your PC.
- Rename the file to "**dpm**xx.**bin**" (xx specifies the slot number the DP master is plugged in, starting with 00) and transfer this file onto a MMC.

Note!
The server always stores the latest two firmware versions.

Preconditions for ftp access

For the display of ftp sites in your web browser you may have to execute the following adjustments:

Internet Explorer

ftp access only with version 5.5 or higher

Options > *Internet options*, Register "Advanced" in the area "Browsing":

- activate: "Enable folder view for ftp sites"
- activate: "Use passive ftp ..."

Netscape

ftp- access only with version 6.0 or higher

No further adjustments are required

If you still have problems with the ftp access, please ask your system operator.

Transfer firmware from MMC into DP master

- Get the RUN-STOP lever of your CPU in position STOP.
- Turn off the voltage supply.
- Plug the MMC with the firmware into the CPU. Please take care of the correct plug-in direction of the MMC.
- Turn on the voltage supply.
- After a short boot-up time, the alternate blinking of the LEDs SF and FC shows that the firmware file has been found on the MMC.
- You start the transfer of the firmware as soon as you tip the RUN/STOP lever downwards to MRES within 10s. The CPU shows the transfer via a LED blink line.
- During the update process, the LEDs SF, FC and MMC are alternately blinking. This may last several minutes.
- The update is successful finished when all CPU-LEDs are on. If they are blinking fast, an error occurred.
- After Power OFF - ON the Master starts with new firmware.

Note!

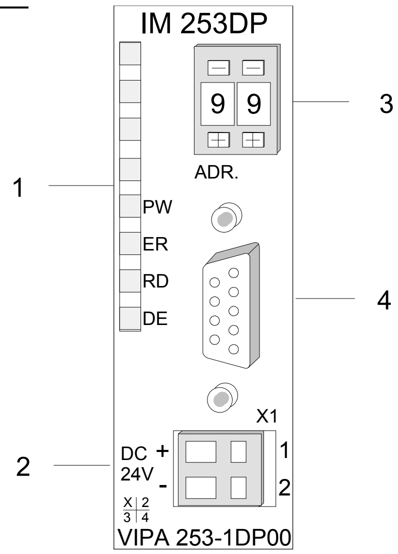
Details to the firmwareupdate can be found at ftp.vipa.de at *support*.

IM 253-1DPx0 - DP-V0 slave - Structure

Properties

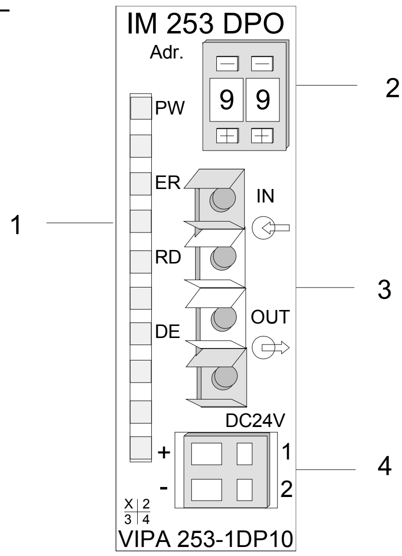
- Profibus (DP-V0)
- Profibus DP slave for max. 32 peripheral modules (max. 16 analog modules)
- Max. 152Byte input data and 152Byte output data
- Internal diagnostic protocol
- Integrated DC 24V power supply for the peripheral modules (3.5A max.)
- Supports all Profibus data transfer rates

Front view 253-1DP00



[1] LED status indicators
[2] Connector for DC 24V power supply
[3] Address selector
[4] RS485 interface

Front view 253-1DP10



[1] LED status indicators
[2] Address selector
[3] FO interface
[4] Connector for DC 24V power supply

Components

LEDs

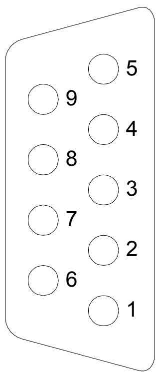
The Profibus slave modules carry a number of LEDs that are available for diagnostic purposes on the bus and for displaying the local status. The following table explains the different colors of the diagnostic LEDs.

Label	Color	Description
PW	green	Indicates that the supply voltage is available on the backplane bus. (Power).
ER	red	Turned on and off again when a restart occurs and is permanently on when an internal error has occurred.
		Blinks when an initialization error has occurred.
		Alternates with RD when the master configuration is bad (configuration error).
		Blinks in time with RD when the configuration is bad.
RD	green	Is turned on when the status is "Data exchange" and the V-bus cycle is faster than the Profibus cycle.
		Is turned off when the status is "Data exchange" and the V-bus cycle is slower than the Profibus cycle.
		Blinks when self-test is positive (READY) and the initialization has been completed successfully.
		Alternates with ER when the configuration received from the master is bad (configuration error).
		Blinks in time with ER when the configuration is bad
DE	green	DE (Data exchange) indicates Profibus communication activity.

RS485 interface

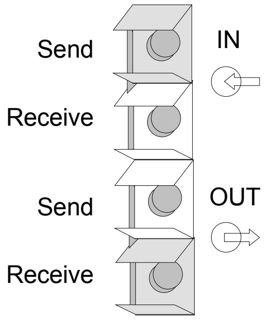
A 9pin socket is provided for the RS485 interface between your Profibus slave and the Profibus.

The following diagram shows the pin assignment for this interface:



Pin	Assignment
1	shield
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.

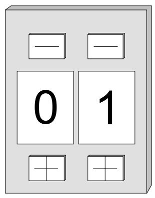
FO interface



These connectors are provided for the optical waveguide between your Profibus coupler and the Profibus.

The diagram on the left shows the layout of the interface.

Address selector



This address selector is used to configure the Profibus address for the DP slave. Addresses may range from 1 to 99. Addresses must be unique on the bus.

The slave address must have been selected before the bus coupler is turned on.

When the address is set to 00 during operation, a once-off image of the diagnostic data is saved to Flash-ROM. Please take care to reset the correct Profibus address, so at the next Power ON the right Profibus address is used!

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.

Profibus and backplane bus are galvanically isolated from each other.

Attention!

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

IM 253-2DP20 - DP-V0 slave with DO 24xDC 24V - Structure

General

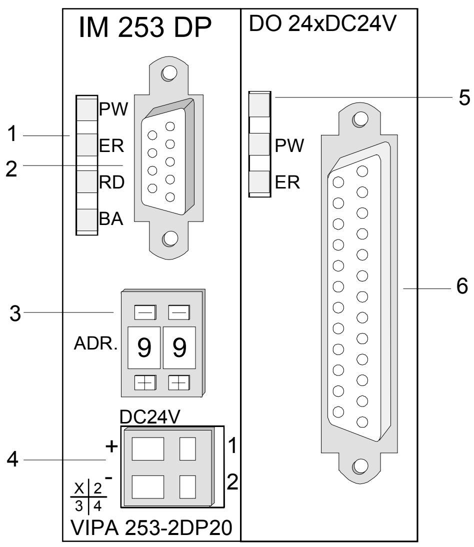
This module consists of a Profibus slave with an integrated 24port output unit. The 24 output channels are controlled directly via the Profibus. The output channels are capable of a maximum load current of 1A each. The total output current must never exceed 4A. The outputs are dc-coupled.

Properties

The following properties distinguish the Profibus output module IM 253DP, DO 24xDC 24V:

- Profibus slave
- 24 digital outputs
- dc-coupled
- Nominal output voltage DC 24V, max. 1A per channel
- Total output current max. 4A
- LED for error indication at overload, over temperature or short circuit
- Suitable for the control of small motors, lamps, magnetic switches and contactors that are controlled via Profibus.

Front view IM 253DP, DO 24xDC 24V



[1] LEDs status indicator Profibus
[2] Profibus socket
[3] Address selector
[4] Connector for DC 24V power supply
[5] LEDs status indicator output unit
[6] 25pin socket for digital output

Attention!
In stand-alone operation, the two sections of the module must be joined by means of the 1tier bus connector that is supplied with the modules!

Components

The components of the Profibus section are identical with the components of the Profibus slave modules that were described above.

LEDs Profibus

The Profibus section carries a number of LEDs that can also be used for diagnostic purposes on the bus.

Label	Color	Description
PW	yellow	Indicates that the supply voltage is available (Power).
ER	red	Turned on and off again when a restart occurs. Is turned on when an internal error has occurred. Blinks when an initialization error has occurred. Alternates with RD when the master configuration is bad (configuration error). Blinks in time with RD when the configuration is bad.
RD	green	Is turned on when the status is "Data exchange" and the V-bus cycle is faster than the Profibus cycle. Is turned off when the status is "Data exchange" and the V-bus cycle is slower than the Profibus cycle. Blinks when self-test is positive (READY) and the initialization has been completed successfully. Alternates with ER when the configuration received from the master is bad (configuration error). Blinks in time with ER when the configuration is bad
DE	yellow	DE (Data exchange) indicates Profibus communication activity.

LEDs digital output section

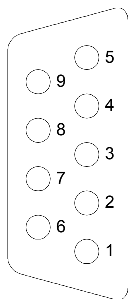
The digital output section is provided with 2 LEDs with the following function:

Designation	Color	Explanation
PW	yellow	Indicates that power is available from the Profibus section (Power).
ER	red	Is turned on at short circuit, overload or over temperature

Profibus RS485 interface

A 9pin RS485 interface is used to connect your Profibus slave to your Profibus.

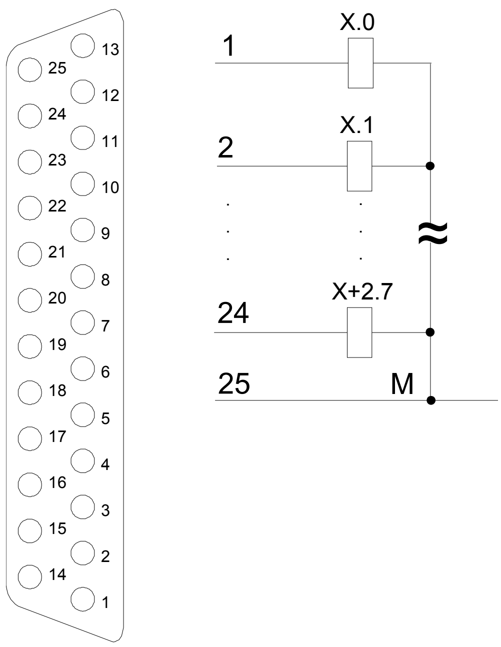
The following diagram shows the pin assignment for this interface:

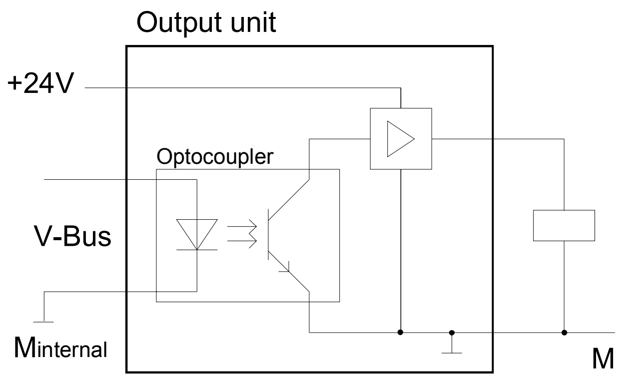


Pin	Assignment
1	shield
2	n.c.
3	RxD/TxD-P (Line B)
4	RTS
5	M5V
6	P5V
7	n.c.
8	RxD/TxD-N (Line A)
9	n.c.

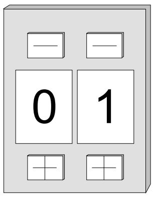
Output unit circuit and block diagram

The DC 24V power supply to the output section is provided internally by the power supply of the slave section.





Address selector



This address selector is used to configure the address for the bus coupler.

Addresses may range from 1 to 99. Addresses must be unique on the bus.

The slave address must have been selected before the bus coupler is turned on.

When the address is set to 00 during operation, a once-off image of the diagnostic data is saved to Flash-ROM. Please take care to reset the correct Profibus address, so at the next Power ON the right Profibus address is used!

Power supply

Every Profibus slave coupler 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 over current.

Profibus and backplane bus are galvanically isolated from each other.

Attention!

If PW is not on when the unit is connected to power, the internal fuse has blown!

Configuration of the outputs

Configure the slave like shown below; the project engineering is for all System 200V Profibus slaves identical.

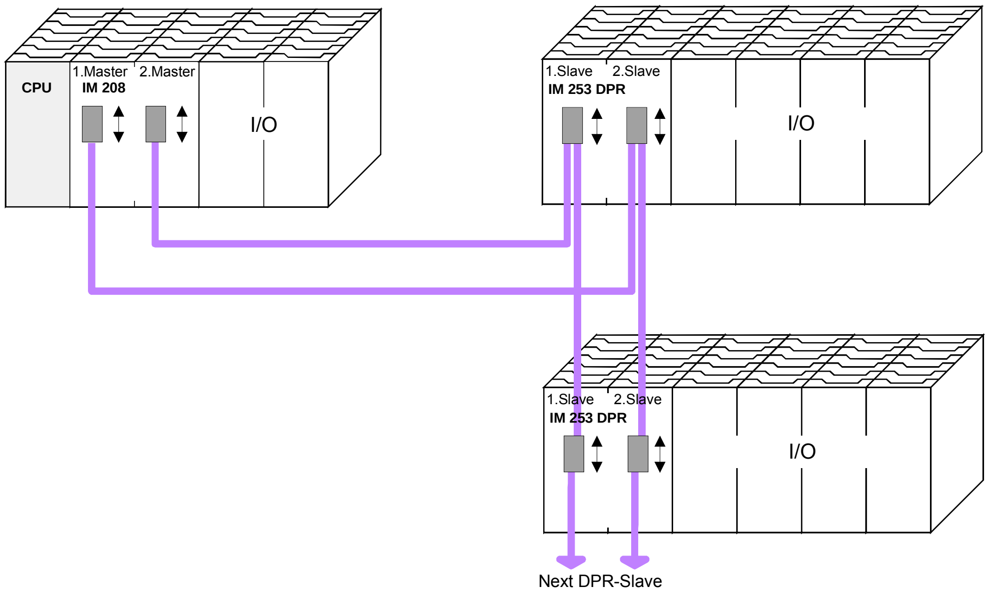
To include the 24 outputs, you should additionally plan the module VIPA 253-2DP20 for the first slot. Seen from the hardware side, the module is directly beside the slave.

IM 253-2DP50 - DP-V0 slave (redundant) - Structure

Redundant system

In principal, the IM 253DPR consists of 2 Profibus DP slave connections. The two Profibus slaves are controlling the operating modes of each other. Both slaves have the same address at the Profibus and are communicating with a redundant DP master.

Both slaves are reading the peripheral inputs. Only one slave at a time has access to the peripheral outputs. The other slave is passive and in stand-by. As soon as the active slave is failing, the passive slave accesses the peripheral outputs.



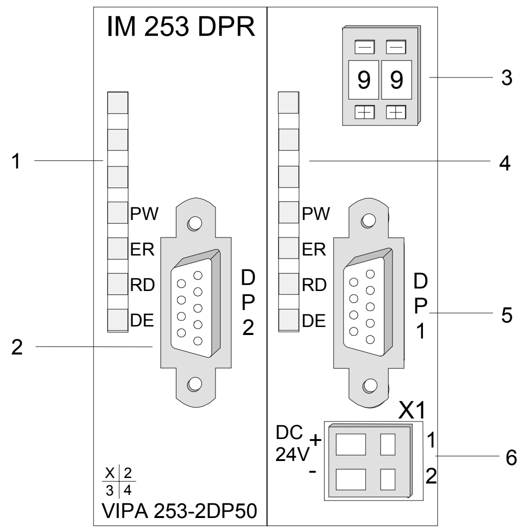
Requirements for the deployment

Please regard to use a redundant DP master for the redundant deployment of the slave module. Every master unit needs the same parameterization and bus configuration.

Properties IM 253DPR

- 2 redundant channels
- DPR slave for max. 32 peripheral modules (max. 16 analog modules)
- Max. 152Byte input data and 152Byte output data
- Internal diagnostic protocol with a time stamp
- Integrated DC 24V power supply for the peripheral modules (max. 3.5A)
- Supports all Profibus data transfer rates

Front view 253-2DP50



[1] LED Status DP2
[2] RS485 interface DP2
[3] Address selector
[4] LED Status DP1
[5] RS485 interface DP1
[6] Connector for DC 24V power supply

Components

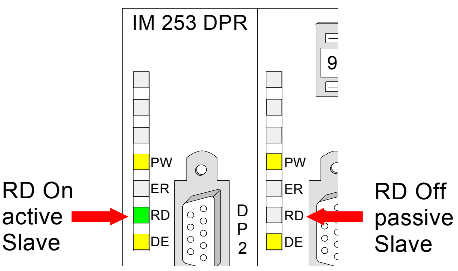
LEDs

The redundant slave includes one LED row for every slave unit that are available for diagnostic purposes The following table explains the different colors of the diagnostic LEDs.

Label	Color	Description
PW	green	Indicates that the supply voltage is available on the backplane bus. (Power).
ER	red	Turned on and off again when a restart occurs. Is turned on when an internal error has occurred. Blinks when an initialization error has occurred. Alternates with RD when the master configuration is bad (configuration error). Blinks in time with RD when the configuration is bad.
RD	green	Blinks at positive self test(READY) and successful initialization.
DE	green	DE (Data exchange) indicates Profibus communication activity.

LEDs at redundant operation

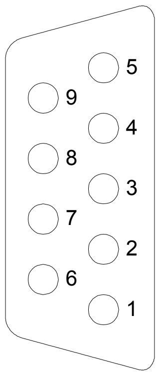
During redundant operation the active slave shows its activity via the green RD-LED, at the passive slave the RD-LED is off. At both slaves the PW- and the DE-LED are on.



RD	DE	Description
on	on	active slave (write and read)
off	on	passive backup slave (read)

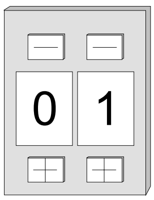
RS485 interface

Via two 9pin RS485 sockets you include the 2 channels into Profibus. Die The following diagram shows the pin assignment for this interface:



Pin	Assignment
1	shield
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.

Address selector



This address selector is used to configure the Profibus address for both DP slaves. Addresses may range from 1 to 99. Addresses must be unique on the bus.

The slave address must have been selected before the bus coupler is turned on.

When the address is set to 00 during operation, a once-off image of the diagnostic data is saved to Flash-ROM. Please take care to reset the correct Profibus address, so at the next Power ON the right Profibus address is used!

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.

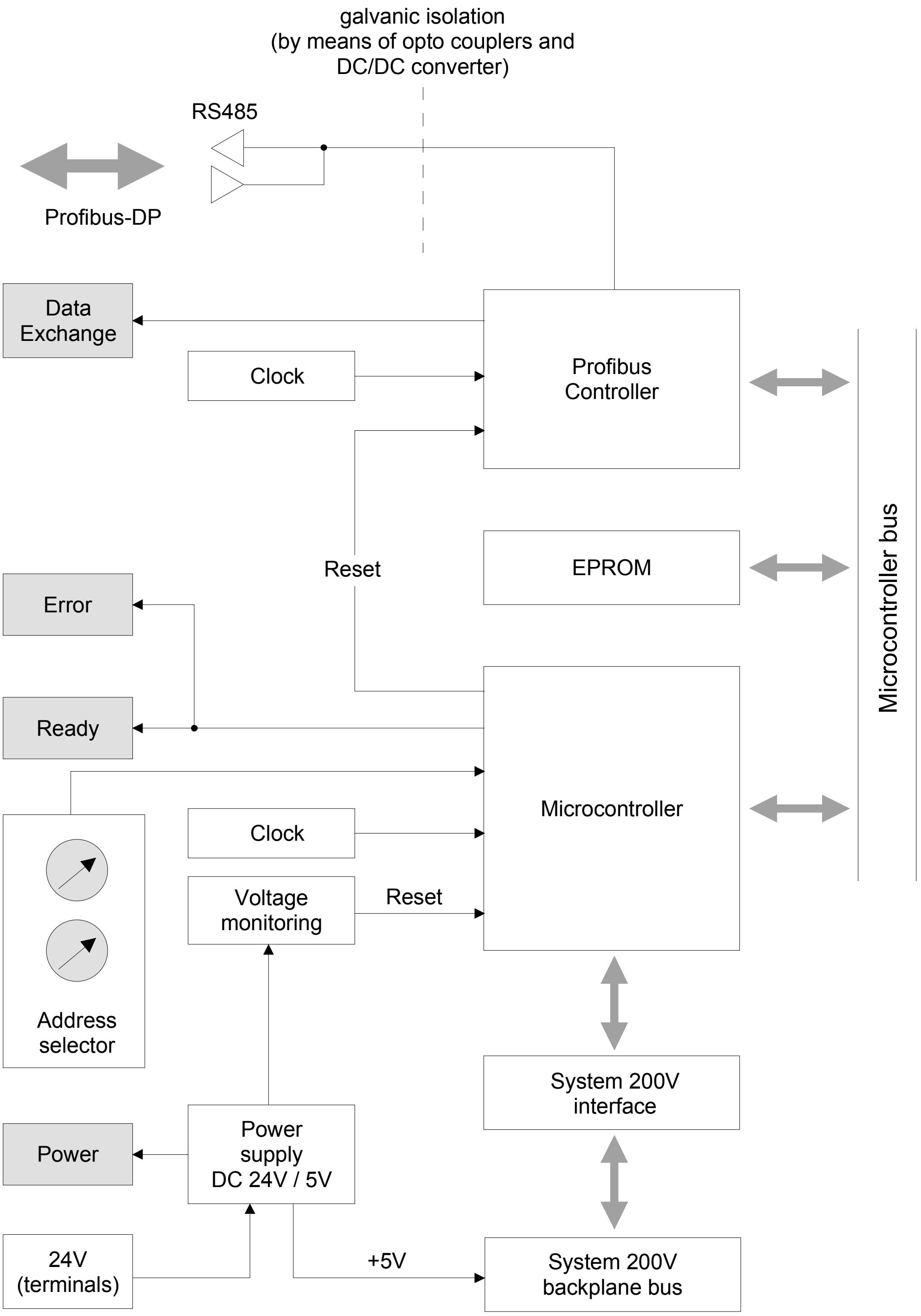
Profibus and backplane bus are galvanically isolated from each other.

Attention!

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

IM 253-xDPx0 - DP-V0 slave - Block diagram

The following block diagram shows the hardware construction of the bus couplers in principal and the internal communication:



IM 253-xDPx0 - DP-V0 Slave - Project engineering

General

The module is configured by means of your Profibus master configuration tool. During the configuration you will assign the Profibus slave modules to your master module.

The direct allocation is defined by means of the Profibus address that you have to set at the slave module.

The Slaves are projected via gsd-file at the hardware configuration.

GSD file

Each Profibus module is delivered with a floppy disk by VIPA. The floppy disc contains all GSD- and type files of the Profibus modules from VIPA as Cx000023_Vxxx.ZIP file. The assignment of the GSD-file to your slave is shown in the "Readme.txt" file of Cx000023_Vxxx.ZIP.

Please install the required files from your floppy 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.

The VIPA WinNCS configuration tool already contains all GSD-files for the VIPA components!

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 at:

Profibus DP>Additional field devices>I/O>VIPA_System_200V>
VIPA 253-1DP00

Deployment IM 253DP, DO 24xDC 24V

At deployment of Profibus DP slave modules like the VIPA 253-2DP20, choose "253-2DP20" as module type.

The slot 1 is mandatory, because the module is, seen from the hardware side, directly beside the slave.

Deployment at a IM 208DP master from VIPA

The project engineering of the IM 253DP slave at the DP master from VIPA is to be find in the description to the DP master in this chapter.

Parameterization in a redundant system

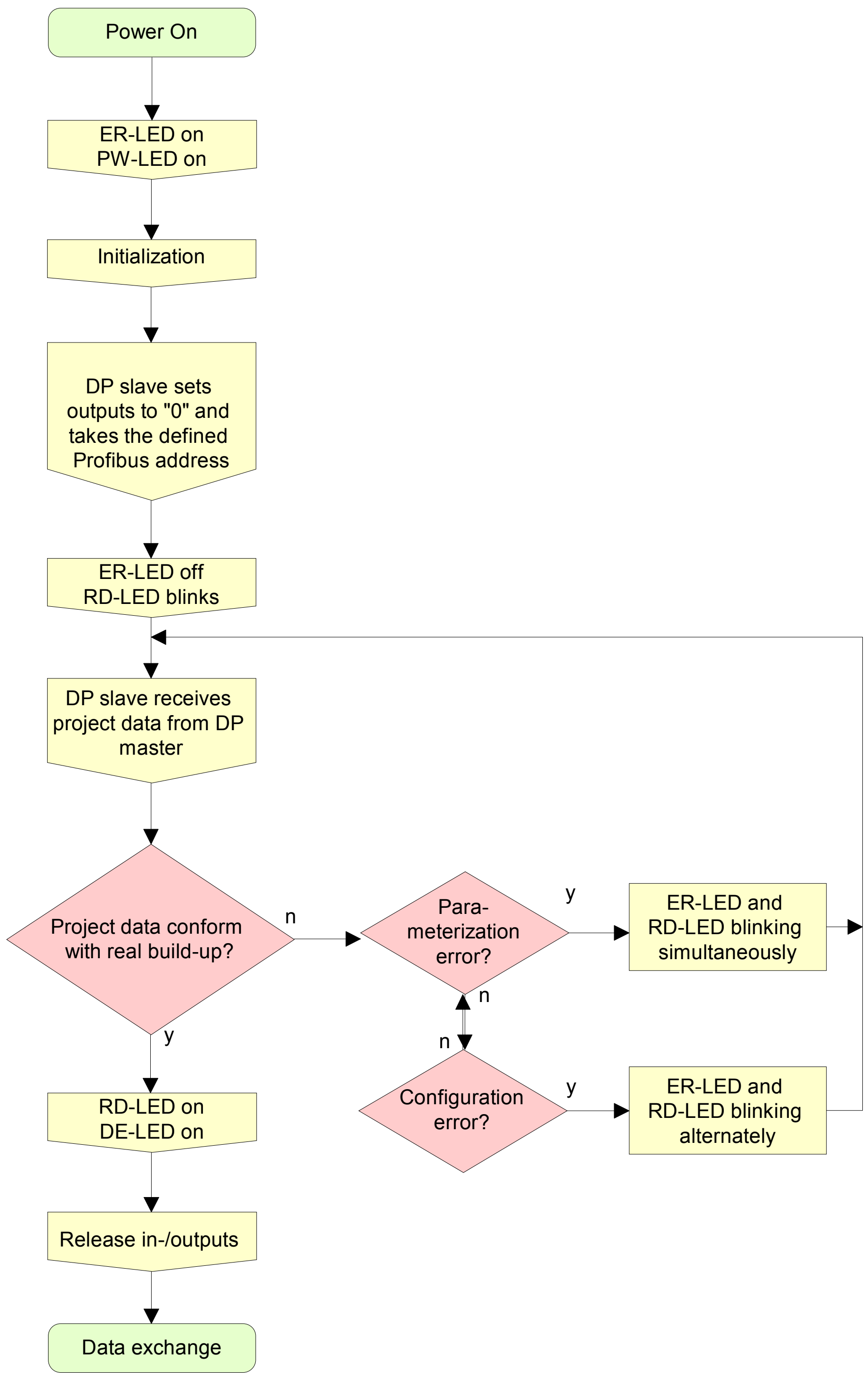
The slave section that achieves firstly the DataExchange state (due to the system, this is always the most left one), is automatically the active slave and has the parameterization access at the peripheral modules.

For assigning new parameters to your remote I/O you should notice that you need an active master-slave-system. Before the transfer of new parameters is possible, both slaves must be in WAITPARAM state.

Start-up behavior IM 253DP 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.



IM 253-xDPx0 - DP-V0 slave - Parameters

Outline

At deployment of DP slaves presented in this manual there are 4 parameters for configuration that are individually used for every slave.

Parameters

The following parameters are available:

Slot number
For reasons of compatibility to VIPA slaves with revision level 4 or lower, you may here select the start number of the slot numeration. With DP slaves rev. level 5 and higher, this parameter is ignored. The following values are possible: 0: slot number 0 (default) 1: slot number 1
Sync Mode
The SYNC-Mode synchronizes the V-Bus cycle (VIPA backplane bus communication) and the DP cycle (Profibus DP communication). This guarantees that there is one Profibus transmission per V-Bus cycle. The following values are possible: Sync Mode off: DP and V-Bus cycle are asynchronous (default) Sync Mode on: DP and V-Bus cycle are synchronous
Diagnostic
Via this parameter you influence the diagnostic function of the slaves. The following values are possible: activated: activates the diagnostic function of the slaves (default) deactivated: deactivates the diagnostic function of the slaves
Redundancy diagnostic
Via this parameter you may influence the redundant diagnostic function of the slaves and it is only accepted with redundant slaves. The following values are possible: activated: activates the red. diagnostic function of the slaves (default) deactivated: deactivates the redundant diagnostic function of the slaves

IM 253-xDPx0 - DP-V0 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.

The most recent 100 diagnostic messages along with a time stamp are stored in RAM res. saved to the Flash of every VIPA Profibus slave. These can be analyzed by means of software.

Please call the VIPA hotline for this purpose.

Internal diagnostic system messages

The system also stores diagnostic messages like the status "Ready" or "DataExchange". These are not send to the master.

The contents of the diagnostic RAM is saved by the Profibus slave in a Flash-ROM, every time the status changes between "Ready" and "DataExchange". At restart it deposits the data back to the RAM.

Saving diagnostic data manually

You can manually save the diagnostic data in Flash-ROM by changing the address switch to 00 during "DataExchange" for a short while.

Diagnostic message in case of a power failure

If a power failure or a voltage drop is detected, a time stamp is saved in the EEPROM. If there is still enough voltage left, the diagnostic data is transferred to the master.

At the next startup the time stamp in the EEPROM is used to generate an undervoltage/power-off diagnostic message and saved to the diagnostic RAM.

Diagnostic addition at IM 253DPR

At deployment of a redundant slave, the diagnostic telegram is extended with an 8Byte sized redundant state. This diagnostic addition is not internally stored. By additionally configuring the state module "State byte IM253-2DP50" as last "module" (most slot number), you are able to include 2Byte of the redundant state into the peripheral area.

This virtual state "module" is available from GSD version 1.30 on.

Structure of the DP-V0 diagnostic data via Profibus

The length of the diagnostic messages that are generated by the Profibus slave is 23Byte. This is also referred to as the *device related diagnostic data*.

When the Profibus slave sends a diagnostic message to the master, a 6Byte standard diagnostic block and 1Byte header is prepended to the 23Byte diagnostic data:

Byte 0 ... Byte 5	Standard diagnostic data	precedes message to master
Byte 6	Header device related diagnostic	only for Profibus transfers
Byte 7 ... 29	**Device related diagnostic data**	**Diagnostic data that is saved internally**
Byte x... Byte x+8	Redundancy state of a redundant DP slave	is only added at transfer via Profibus and usage of the redundant slave

Standard diagnostic data

Diagnostic data that is being transferred to the master consist of the standard diagnostic data for slaves and a header byte that are prepended to the device related diagnostic bytes. The Profibus standards contain more detailed information on the structure of standard diagnostic data. These standards are available from the Profibus User Organization. The structure of the standard diagnostic data for slaves is as follows:

Byte	Bit 7 ... Bit 0
0	Bit 0: permanently 0 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: permanently 0 Bit 6: bad configuration Bit 7: permanently 0
1	Bit 0: slave requires re-configuration Bit 1: statistical diagnostic Bit 2: permanently 1 Bit 3: Watchdog active Bit 4: Freeze-command was received Bit 5: Sync-command was received Bit 6: reserved Bit 7: permanently 0
2	Bit 0 ... Bit 6: 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

Header for device related diagnostic

This byte is only prepended to the device related diagnostic data when this is being transferred via Profibus.

Byte	Bit 7 ... Bit 0
6	Bit 0 ... Bit 5: Length device related diagnostic data incl. Byte 6 Bit 6 ... Bit 7: permanently 0

Device related diagnostic

Byte	Bit 7 ... Bit 0
7 ... 29	Device related diagnostic data that can be stored internally by the slave for analysis

Structure of the device related diagnostic data in the DP slave

As of revision level 6, all diagnostic data that is generated by the Profibus slave is stored in a ring-buffer along with the time stamp. The ring-buffer always contains the most recent 100 diagnostic messages.

You can analyze these messages by means of the "Slave Info Tool".

Since the standard diagnostic data (Byte 0 ... Byte 5) and the header (Byte 6) are not stored, the data in Byte 0 ... Byte 23 corresponds to Byte 7 ... Byte 30 that is transferred via Profibus.

The structure of the device related diagnostic data is as follows:

Byte	Bit 7 ... Bit 0
0	Message 0Ah: DP parameter error 14h: DP configuration error length 15h: DP configuration error entry 1Eh: undervoltage/power failure 28h: V-bus parameterization error 29h: V-bus initialization error 2Ah: V-bus bus error 2Bh: V-bus delayed acknowledgment 32h: diagnostic alarm System 200 33h: process alarm System 200 3Ch: new DP address was defined 3Dh: slave status is ready (only internally) 3Eh: slave status is DataExchange (only internally)
1	Module no. or slot no. 1 ... 32: module no. slot no. 0: module no. slot no. not available
2 ... 23	Additional information for message in Byte 0

Overview of diagnostic messages

The following section contains all the messages that the diagnostic data can consist of. The structure of Byte 2 ... Byte 23 depends on the message (Byte 0). When the diagnostic data is transferred to the master via Profibus, Byte 7 of the master corresponds to Byte 0 of the slave. The specified length represents the "length of the diagnostic data" during the Profibus data transfer.

0Ah

DP parameter error Length: 8

The parameter telegram is too short or too long

Byte	Bit 7 ... Bit 0
0	0Ah: DP parameter error
1	Module no. or slot no. 1 ... 32: module no. or slot no. 0: module no. or slot no. not available
2	Length user parameter data
3	Mode 0: standard mode 1: 400-mode
4	Number of digital modules (slave)
5	Number of analog modules (slave)
6	Number of analog modules (master)

14h

DP configuration error - length Length: 6

Depending on the mode, the length of the configuration message is compared to the length of the default configuration (modules detected on the V-Bus).

Byte	Bit 7 ... Bit 0
0	14h: DP configuration error - length
1	Module no. or slot no. 1 ... 32: module no. or slot no. 0: module no. or slot no. not available
2	Configuration data quantity (master)
4	Configuration data quantity (slave)
3	Mode 0: Standard mode 1: 400-mode

15h

DP configurations error - entry Length: 6

Depending on the mode and when the length of the configuration message matches the length of the default configuration the different entries in the configuration message are compared to the default configuration.

Byte	Bit 7 ... Bit 0
0	15h: DP configuration error - entry
1	Module no. or slot no. 1 ... 32: module no. or slot no. 0: module no. or slot no. not available
2	Configuration byte master (module identifier)
4	Configuration byte slave (module identifier)
3	Mode 0: Standard mode 1: 400-mode

1Eh

Undervoltage/power failure Length: 2

A time stamp is saved immediately to the EEPROM when a power failure or a voltage drop is detected. If there is still enough voltage, the diagnostic data is transferred to the master.

At the next restart, the time stamp in the EEPROM is used to generate an undervoltage/power-off diagnostic message that is saved in the diagnostic RAM.

Byte	Bit 7 ... Bit 0
0	1Eh: Undervoltage/power failure

28h

V-bus configuration error Length: 3

The configuration for the specified slot failed.

Byte	Bit 7 ... Bit 0
0	28h: V-bus configuration error
1	Module no. or slot no. 1 ... 32: module no. or slot no. 0: module no. or slot no. not available

29h

V-bus initialization error Length: 2

General backplane bus error

Byte	Bit 7 ... Bit 0
0	29h: V-bus initialization error

2Ah

V-bus bus error Length: 2

Hardware error or module failure

Byte	Bit 7 ... Bit 0
0	2Ah: V-bus error

2Bh

V-bus delayed acknowledgment Length: 2

Reading or writing from/to digital modules failed

Byte	Bit 7 ... Bit 0
0	2Bh: V-bus delayed acknowledgment

32h

System 200V diagnostic alarm Length: 16

Byte	Bit 7 ... Bit 0
0	32h: System 200V diagnostic alarm
1	Module no. or slot no. 1 ... 32: module no. or slot no. 0: module no. or slot no. not available
2 ... 14	Data diagnostic alarm

33h

System 200V process alarm Length: 16

Byte	Bit 7 ... Bit 0
0	33h: System 200V process alarm
1	Module no. or slot no. 1 ... 32: module no. or slot no. 0: module no. or slot no. not available
2 ... 14	Process alarm data

3Ch

New DP address assigned Length: 2

When the slave has received the service with "Set Slave Address" it sends the respective diagnostic message and re-boots. The slave will then become available on the bus under the new address.

Byte	Bit 7 ... Bit 0
0	3Ch: new DP address has been assigned

3Dh

Slave status is READY Length: none (internal only)

The READY status of the slave is only used internally and is not transmitted via the Profibus.

Byte	Bit 7 ... Bit 0
0	3Dh: slave status is READY

3Eh

Slave status is DataExchange Length: none (only internal)

The DataExchange status of the slave is only used internally and is not transmitted via the Profibus.

Byte	Bit 7 ... Bit 0
0	3Eh: slave status is DataExchange

Redundancy state at deployment of IM 253DPR

At deployment of a redundant slave, the diagnostic message is expanded for 8Byte data with the redundancy state. This diagnostic addition is not stored in the internal diagnostic buffer. The redundancy state has the following structure:

Redundancy state

Byte	Description
X	08h: length of redundancy state permanent at 8
X+1	80h: type of redundancy state
X+2	00h: reserved, permanent 00h
X+3	00h: reserved, permanent 00h
X+4	00h: reserved, permanent 00h
X+5	Red_State slave that communicates with the respective master)
	Bit 0 = slave is backup slave
	Bit 1 = slave is primary slave
	Bit 2 = reserved
	Bit 3 = reserved
	Bit 4 = slave is in DataExchange
	Bit 5 = reserved
	Bit 6 = reserved
	Bit 7 = reserved
X+6	Red_State of second slave
X+7	00h: reserved, permanent 00h

Include the redundancy state into the peripheral area

As from GSD version 1.30 from VIPA, the virtual module "State byte IM253-2DP50" is available in the hardware catalog. When using this module during the project engineering. You may define an address range of 2Byte where the Red_State byte of both slaves shall be stored.

Please regard that you have to configure this module always at the last slot, otherwise the slave will throw a parameterization error.

(De)activate diagnostic

Via the parameterization window of the slaves, you may influence the diagnostic functions by activating res. deactivating diagnostic or the redundancy state.

IM 253-xDPx1 - DP-V1 slave - Structure

Properties
253-1DP01
253-1DP11

- 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

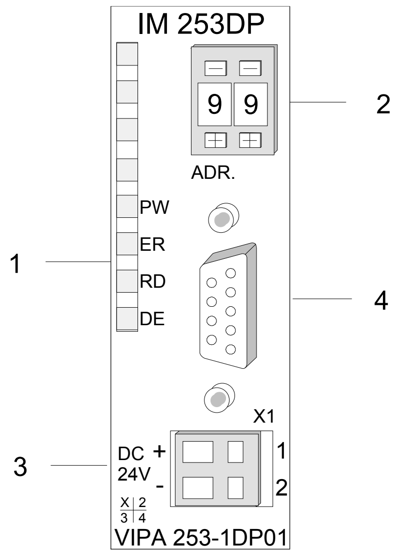
- 1 MSAC_C1 connection (Read, Write) with 244Byte data
  (4 Byte DP-V1-Header + 240Byte user data)
- 3 MSAC_C2 connections (Initiale, Read, Write, DataTransport, Initiate Abort) with each 244Byte data
  (4 Byte DP-V1-Header + 240 Byte user data)

Restrictions
253-1DP31 - ECO

The IM 253-1DP31 - ECO is functionally identical to the IM 253-1DP01 and has the following restrictions:

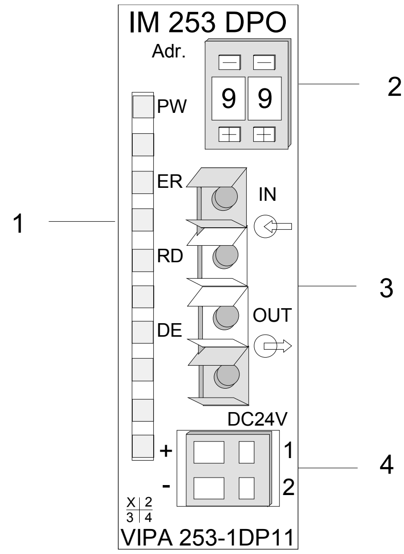
- Profibus DP slave for max. 8 periphery modules
- Integrated DC 24V power supply for the peripheral modules (0.8A max.)
- The Profibus address can be adjusted by DIP switch

Front view
253-1DP01



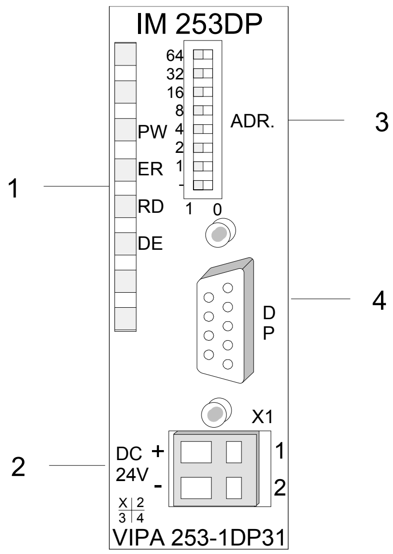
[1] LED status indicators
[2] Address selector (Coding switch)
[3] Connector for DC 24V power supply
[4] RS 485 interface

Front view
253-1DP11



[1] LED status indicators
[2] Address selector
(Coding switch)
[3] FO interface
[4] Connector for DC 24V power supply

Front view
253-1DP31 - ECO



[1] LED status indicators
[2] Connector for DC 24V power supply
[3] Address selector
(DIP switch)
[4] RS485 interface

Components

LEDs

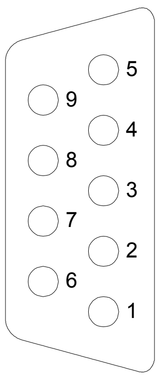
The Profibus slave modules carry a number of LEDs that are available for diagnostic purposes on the bus and for displaying the local status. The following table explains the different colors of the diagnostic LEDs.

Label	Color	Description
PW	green	Indicates that the supply voltage is available on the backplane bus. (Power).
ER	red	Turned on and off again when a restart occurs and is permanently on when an internal error has occurred.
		Blinks when an initialization error has occurred.
		Alternates with RD when the master configuration is bad (configuration error).
		Blinks in time with RD when the configuration is bad.
RD	green	Is turned on when the status is "Data exchange" and the V-bus cycle is faster than the Profibus cycle.
		Is turned off when the status is "Data exchange" and the V-bus cycle is slower than the Profibus cycle.
		Blinks when self-test is positive (READY) and the initialization has been completed successfully.
		Alternates with ER when the configuration received from the master is bad (configuration error).
		Blinks in time with ER when the configuration is bad
DE	green	DE (Data exchange) indicates Profibus communication activity.

RS485 interface

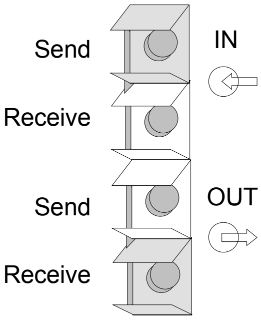
A 9pin socket is provided for the RS485 interface between your Profibus slave and the Profibus.

The following diagram shows the pin assignment for this interface:



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.

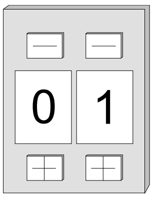
FO interface



These connectors are provided for the optical waveguide between your Profibus coupler and the Profibus.

The diagram on the left shows the layout of the interface.

Address selector

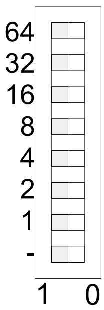


This address selector is used to configure the Profibus address for the DP slave. Addresses may range from 1 to 99. Addresses must be unique on the bus.

The slave address must have been selected before the bus coupler is turned on.

When the address is set to 00 during operation, a once-off image of the diagnostic data is saved to Flash-ROM. Please take care to reset the correct Profibus address, so at the next PowerOn the right Profibus address is used!

Address selector
IM 253-1DP31 - ECO



Contrary to the coding switched described above at the IM 253-1DP31 - ECO the Profibus address is configured by means of a DIL switch. Addresses may range from 1 to 125. Addresses must be unique on the bus.

The slave address must have been configured before the bus coupler is turned on.

When the address is set to 00 during operation, a once-off image of the diagnostic data is saved to Flash-ROM. Please take care to reset the correct Profibus address, so at the next PowerON the right Profibus address is used!

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 back plane current of the IM 253-1DP31 - ECO is limited to 0.8A.

The power supply is protected against reverse polarity.

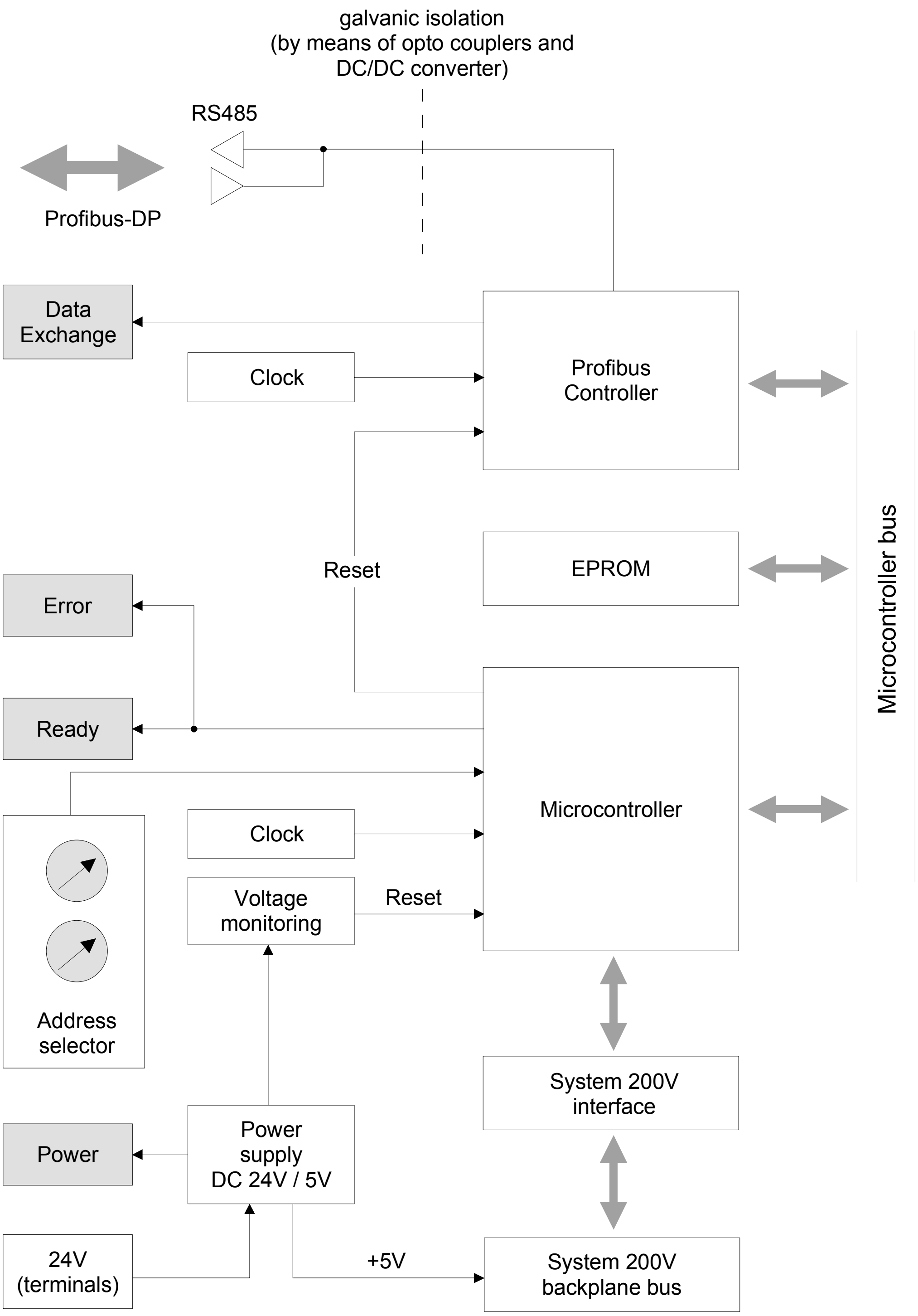
Profibus and backplane bus are isolated from each other.

Attention!

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

IM 253-xDPx1 - DP-V1 slave - Block diagram

The following block diagram shows the hardware construction of the bus couplers in principal and the internal communication:



IM 253-xDPx1 - DP-V1 slave - Project engineering

General

The module is configured by means of your Profibus master configuration tool. During the configuration you will assign the Profibus slave modules to your master module.

The direct allocation is defined by means of the Profibus address that you have to set at the slave module.

The Slaves are projected via gsd-file at the hardware configuration.

GSD- file

Each Profibus module is delivered with a floppy disk by vipa. The floppy disc contains all GSD- and type files of the Profibus modules from VIPA as Cx000023_Vxxx.ZIP file. The assignment of the GSD-file to your slave is shown in the "Readme.txt" file of Cx000023_Vxxx.ZIP.

Please install the required files from your floppy 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 e.g. the DP-V1 slave in the hardware catalog from Siemens at:

Profibus DP>Additional field devices>I/O>VIPA_System_200V>
VIPA 253-1DP01

Note!

Please use the appropriate GSD for DP-V0 for Profibus DP master which do not support DP-V1.

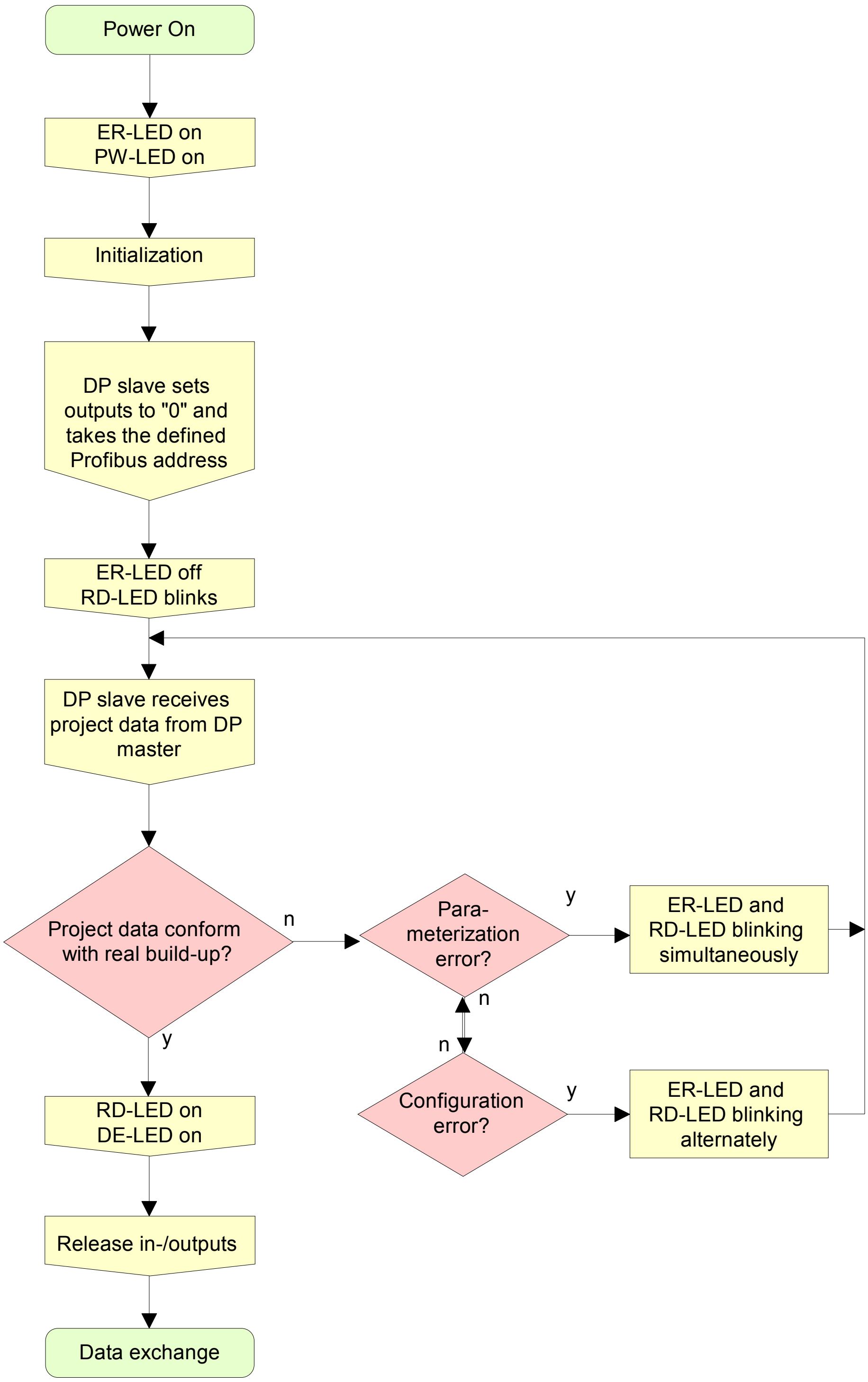
Deployment at a IM 208DP master from VIPA

The project engineering of the IM 253 DP slave at the DP master from VIPA is to be find in the description to the DP master in this chapter.

Start-up behavior IM 253DP 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.



IM 253-xDPx1 - DP-V1 slave - Parameters

Outline

At deployment of DP slaves presented in this manual there are parameters for configuration that are individually used for every slave.

Parameters DP-V0

At usage of the corresponding GSD for DP-V0 operation you have the following parameter data:

Byte	Bit 7 ... Bit 0	Default
0	Bit 1 ... 0: 0 (fix) Bit 2: 0 = WD-Timebase 10ms 1 = WD-Timebase 1ms Bit 4 ... 3: 0 (fix) Bit 5: 0 = Publisher-Mode not available 1 = Publisher-Mode available Bit 7 ... 6: 0 (fix)	00h
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 = Enhanced diagnostic enable 1 = Enhanced diagnostic disable Bit 1: 0 = Module status enable 1 = Module status disable Bit 2: 0 = Channel-specific diagnostic enable 1 = Channel-specific diagnostic disable Bit 3: 0 (fix) Bit 4: 0 = V0: Manufacturer alarm not available 1 = V0: Manufacturer alarm available Bit 5: 0 = V0: Diagnostic alarm not available 1 = V0: Diagnostic alarm available Bit 6: 0 = V0: Process alarm not available 1 = V0: Process alarm available Bit 7: 0 (fix)	70h
8	Bit 6 ... 0: 0 (fix) Bit 7: 0 = Data format Motorola 1 = Data format Intel (only at analog modules)	00h
9 ... 12	00h (fix)	00h

DP-V1 UserPrmData

At usage of a GSD for DP-V1 operation you have the following parameter data:

Byte	Bit 7 ... Bit 0	Default
0	Bit 1 ... 0: 0 (fix) Bit 2: 0 = WD-Timebase 10ms 1 = WD-Timebase 1ms Bit 4 ... 3: 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	80h
1	Bit 3 ... 0: 0 (fix) Bit 4: 0 = V1: Manufacturer alarm not available 1 = V1: Manufacturer alarm* available Bit 5: 0 = V1: Diagnostic alarm not available 1 = V1: Diagnostic alarm available Bit 6: 0 = V1: Process alarm not available 1 = V1: Process alarm available Bit 7: 0 (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-specific diagnostic enable 1 = Channel-specific diagnostic disable Bit 7 ... 3: 0 (fix)	00h
8	Bit 6 ... 0: 0 (fix) Bit 7: 0 = Data format Motorola 1 = Data format Intel (only at analog modules)	00h
9 ... 12	00h (fix)	00h

*) The IM 253-1DP31 does not support manufacturer alarm.

Data format Motorola/Intel

This parameter is exclusively evaluated with deployment of analog modules and refers to how a value is stored in the CPU address range.

In the *Motorola format* (default) the bytes were stored in descending significance i.e. the 1st byte contains the high byte and 2nd byte the low byte.

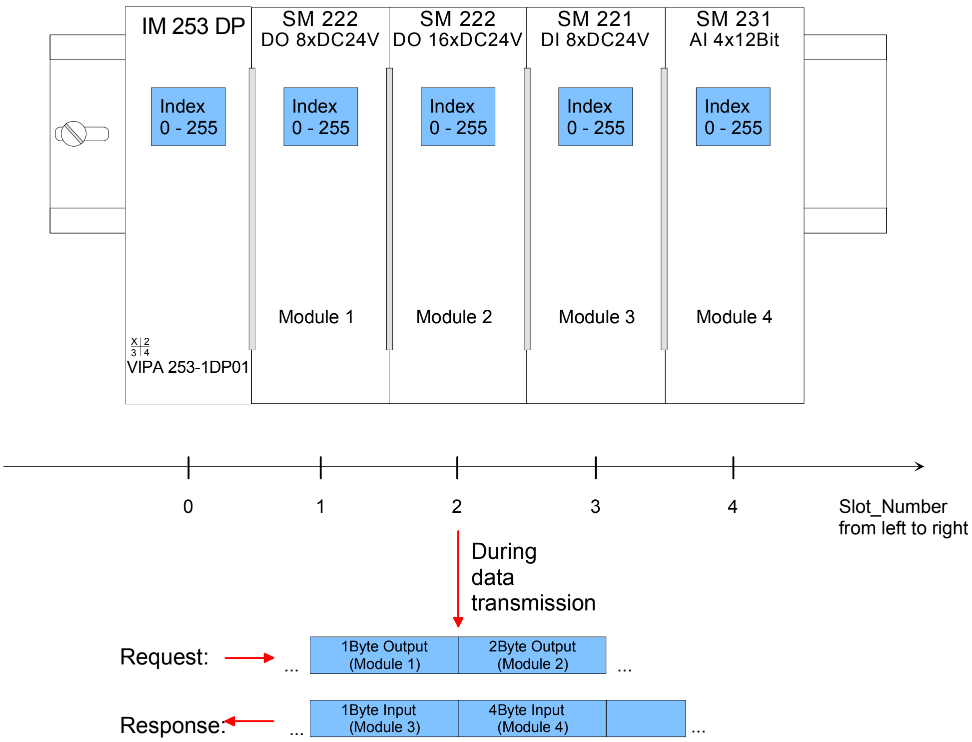
In the *Intel format* the value is switched and it is worked with ascending significance i.e. the 1st byte contains the low byte and 2nd byte the high byte.

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 DPM1 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 assigned to a module. The Slot_Number = 0 addresses the data of the PROFIBUS coupler, the 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.

Read res. write access via SFB 52 res. 53

Starting with the firmware version 1.3.0 your CPU has the SFB 52 res. 53 integrated for DP-V1 read res. write accesses. Here you may access the according component of your system by declaring the ID (Slot number as address) and index.

More detailed information is given in the description of SFB 52/53.

Data transmission

Per default, one class-1 master and one 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

At access to the DP-V1 coupler via Slot_Number = 0 you have access to the following elements via *Index*:

Index	Access	Description
A0h	R	Device name (VIPA 253-1DP01)
A1h	R	Hardware Version (V1.00)
A2h	R	Software Version (V1.00)
A3h	R	Serial number from the device (e.c. 000347 = 30h, 30h, 30h, 33h, 34h, 37h)
A4h	R	Device configuration (see module configuration and module type)
D0h	R	Number of stored diagnostic
	W	Deletes diagnostic entries
D1h	R	Diagnostic entries
	W	Stores diagnostic entries permanently in the FLASH memory
FFh	R	I&M functions
	W	

R = Read; W = Write

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 (alarm) that is stored internal

Data of the function modules

Index	Access	Description
00h	R	Diagnostic – record set 0
	W	Module parameter
01h	R	Via "Index" you may access the according diagnostic of a module by presetting a record set number. Example: Index=01h → Access to diagnostic record set 01
F1h	R	Module parameter
F2h	R	Read module process image

R = Read; W = Write

Module configuration

Via the index A3h, the module configuration of the modules at the backplane bus can be monitored.

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
SM 238	45DCh 38C4h	8 16	4 16
CP 240	1CC1h	16	16
FM 250	B5F4h	10	10
FM 250-SSI	B5DBh	4	4
FM 253, FM 254	18CBh	16	16

IM 253-xDPx1 - 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 alarms and status messages.

Additionally in the DP-V1 slave from VIPA the last 100 alarm 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!

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 DP-V1 diagnostic data via Profibus

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 norm diagnostic data:

Byte 0 ... Byte 5	Norm diagnostic data	
Byte 6 ... 10	Identifier related diagnostic *	
x ... x+11	Module state*	
7...13 ·(x ... x+2)	Channel related diagnostic*	
x ... x+19	Alarm*	Internal stored diagnostic

*) Can be enabled or disabled via parameterization

Diagnostic data IM 253-1DP31 - ECO

Due to the restrictions there are the following diagnostic data for the IM 253-1DP31 - ECO:

Byte 0 ... Byte 5	Normdiagnose-Daten	
Byte 6 ... 7	Kennungsbezogene Diagnose*	
x ... x+5	Modulstatus*	
10...13 ·(x ... x+2)	Kanalbezogene Diagnose*	
x ... x+19	Alarm*	Internal stored diagnostic

*) Can be enabled or disabled via parameterization

Norm diagnostic data

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

The slave *norm diagnostic data* have the following structure:

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

Enhanced diagnostic

Via the *Enhanced diagnostic,* which can be activated by parameterization, you gain information at which slot number (module) an error has occurred. More detailed information about the error is available via the *Module state* and the *channel specific diagnostic*.

Note!

Note that the length of the *enhanced diagnostic* of the IM 253-1DP31 - ECO is limited to 2.

Enhanced diagnostic

Byte	Bit 7 ... Bit 0
X	Bit 5 ... 0: 000101 (fix) Length of the Enhanced diagnostic* Bit 7 ... 6: 01 (fix) Code for Enhanced 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

*) Bit 5 ... 0: 000010 at 253-1DP31 - ECO

Module state

Via the *Module state,* which can be activated by parameterization, you gain information about the error that occurred at a module.

Note!

Note that the length of the *Module state* of the IM 253-1DP31 - ECO is limited to 6.

Module state

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

*) Bit 5 ... 0: 000110 at 253-1DP31 - ECO

Channel specific Diagnostic

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

Channel-specific diagnostic

Byte	Bit 7 ... Bit 0
X	Bit 5 ... 0: ID number of the module that delivers the channel-specific diagnostic (000001 ... 011111)* z.B.: Slot 1 has ID no. 0 Slot 32 has ID no. 31 Bit 7, 6: 10 (fix) Code for channel-specific 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: Process interrupt lost 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

*) Bit 5 ... 0: 000001...001000 (slot 1...8) at 253-1DP31 - ECO

The maximum number of *channel specific 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 specific diagnostic*. For each channel always 3 Byte are used.

Interrupts

The interrupts section of the slave diagnostic provides information on the type of interrupt and the cause that triggered the input. The interrupt section has a maximum of 20bytes. A maximum of one interrupt can be used per slave diagnostic. The interrupt component is always the last part of the diagnostic frame.

Contents

The contents of the interrupt information depend on the type of interrupt:

- In the case of *diagnostic interrupts*, the diagnostic data record 1 is send as interrupt information (as of Byte x+4)
- In the case of *process interrupts*, the additional information is 4bytes long. These data is module specific and is described at the concerning module.

Alarm status

If there is a diagnostic event for channel (/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:

Alarm status Byte x ... x+3

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

Alarm status at diagnostic alarm 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 0 ... 3: Module class 1111: Digital module 0101: Analog module 1000: FM 1100: CP Bit 4: Channel information available Bit 5: User information available Bit 6: always "0" Bit 7: always "0"
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: always "0"
x+7	Bit 0: reserved Bit 1: reserved Bit 2: reserved Bit 3: reserved Bit 4: reserved Bit 5: reserved Bit 6: Process interrupt lost Bit 7: reserved

Continued …

… Continue

Alarm status at diagnostic alarm Bytes x+8 to x+19
(corresponds 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	Lenght of the channel-specific diagnostic
x+10	Number of channels per module
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

Alarm status at process alarm Bytes x+4 to x+7

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

IM 253-xDPx1 - DP-V1 slave - Firmware update

Overview

The firmware update for the DP-V1 slave VIPA 253-1DP01 is at this time only available with Siemens CPUs. For this your firmware is online transferred from the hardware configurator to the CPU which passes the firmware on to the according DP slave via the connected DP master using Profibus.

Note!

The DP slaves IM 253-1DP31 - ECO and IM 253-1DP11 don't support a firmware update!

Approach

- Make firmware file available
- Load project into the hardware configurator
- Transfer firmware

Supply firmware file *header.upd*

The most recent firmware for the DP-V1 Profibus slaves is to find at

ftp.vipa.de/support/firmware/System%20200V/DP_Slave/IM253-1DP01

as package Px000019_Vxxx.zip with xxx=version.

Extract and copy the file *header.upd* into your work directory.

Load project into hardware configurator

- Open the hardware configurator with the configured DP slave.
- Click on the DP slave and choose **PLC** > *Update Firmware*. This menu option is only available when the highlighted DP slave supports the function "Update firmware ".
  → the dialog window "Update firmware " appears.
- Choose your work directory via the button "Search" where the file *header.upd* is stored. Choose *header.upd*. → You will see information for which modules and from which firmware version on the chosen file is convenient.
- Activate the control field "Activate firmware after loading" because only then the new firmware is copied to the Flash and click then on [Execute].
  → it is proofed if the chosen file is valid and at positive result the file is transferred to the DP slave.

Note!

During runtime the firmware update at the DP slave is executed after app. 3s. Please regard that the DP slave executes a reboot which may cause the DP master to remain in STOP res. may influence your user application.

IM 253-xDPx1 - DP-V1 slave - I&M data

Overview

Identification and maintenance data (I&M) are stored information in a module which support you at:

- check of the system configuration
- discover of hardware changes
- remove errors in a system

Identification data (I data) are information of the module e.g. order number, serial number, which can be found printed at the module.

I data are manufacturer information and can only be read.

Maintenance data (M data) are information like location and date of installation. M data were produced and stored during project engineering

By means of I&M data the modules can online be identified. Starting with Profibus firmware V110 the data are available at the Profibus slaves.

Note!
Only one DP master may access at one time the I&M data.

Structure

The data structure of the I&M data corresponds to the specifications of Profibus guideline - order no. 3.502, version 1.1 from May 2003.

I&M data	Access	Preset	Explanation
Identification data 0: IM_INDEX: 65000			
MANUFACTURER_ID	read (2Byte)	22B hex (555 dez)	Name of the manufacturer (555 dez = VIPA GmbH)
ORDER_ID	read (20Byte)	depends on the module	Order number of the module VIPA 253-1DP01/31
SERIAL_NUMBER	read (16Byte)	depends on the module	Serial number of the module for clear identification.
HARDWARE_REVISION	read (2Byte)	depends on the module	Hardware revision of the module which is incremented on changes at the firmware.

continued ...

... continue

SOFTWARE_REVISION	read (4Byte)	Firmware version Vxyz	Firmware version of the module. An increase of the firmware version also increases the hardware revision
REVISION_COUNTER	read (2Byte)	0000 hex	reserved
PROFILE_ID	read (2Byte)	F600 hex	Generic Device
PROFILE_SPECIFIC_TYPE	read (2Byte)	0003 hex	at I/O modules
IM_VERSION	read (2Byte)	0101 hex	Information about the version of the I&M data. (0101 hex = Version 1.1)
IM_SUPPORTED	read (2Byte)	001F hex	Information about available I&M-Data (IM_INDEX: 650000 ...65004)
Maintenance data 1: IM_INDEX: 65001			
TAG_FUNCTION	read / write (32Byte)	_	Clear module ID inside the system
TAG_LOCATION	read / write (22Byte)	_	Location of installation of the module
Maintenance data 2: IM_INDEX: 65002			
INSTALLATION_DATE	read / write (16Byte)	_	Date and if applicable time of installation of the module
RESERVED	read / write (38Byte)	_	reserved
Maintenance data 3: IM_INDEX: 65003			
DESCRIPTOR	read / write (54Byte)	_	Commentary to the module
Maintenance data 4: IM_INDEX: 65004			
SIGNATURE	read / write (54Byte)	_	Commentary to the module

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 126 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.

Fiber optic system

- Only one fiber optic master may be used on one line.
- Multiple masters may be deployed with a single CPU as long as they are located on the same backplane bus (please take care not to exceed the max. current consumption).
- The maximum length of a FO link between two slaves may not exceed 300m with HCS-FO and 50m with POF-FO, independent from the baud rate.
- The number of bus participants depends on the baud rate:

≤ 1.5MBaud	→	17 participants incl. master
3MBaud	→	15 participants incl. master
6MBaud	→	7 participants incl. master
12MBaud	→	4 participants incl. master

- The bus does not require termination.

Note!

You should place covers on the unused sockets on any fiber optic device (e.g. the jack for the following participant at the bus end) to prevent being blinded by the light or to stop interference from external light sources. You can use the supplied rubber stoppers for this purpose. Insert the rubber stoppers into the unused openings on the FO interface.

Electrical system

- The bus must be terminated at both ends.
- Masters and slaves may be installed in any combination.

Combined system

- Any FO master may only be installed on an electrical system by means of an **O**ptical **L**ink **P**lug, i.e. slaves must not be located between a master and the OLP.
- Only one converter (OLP) is permitted between any two masters.

Installation and integration with Profibus

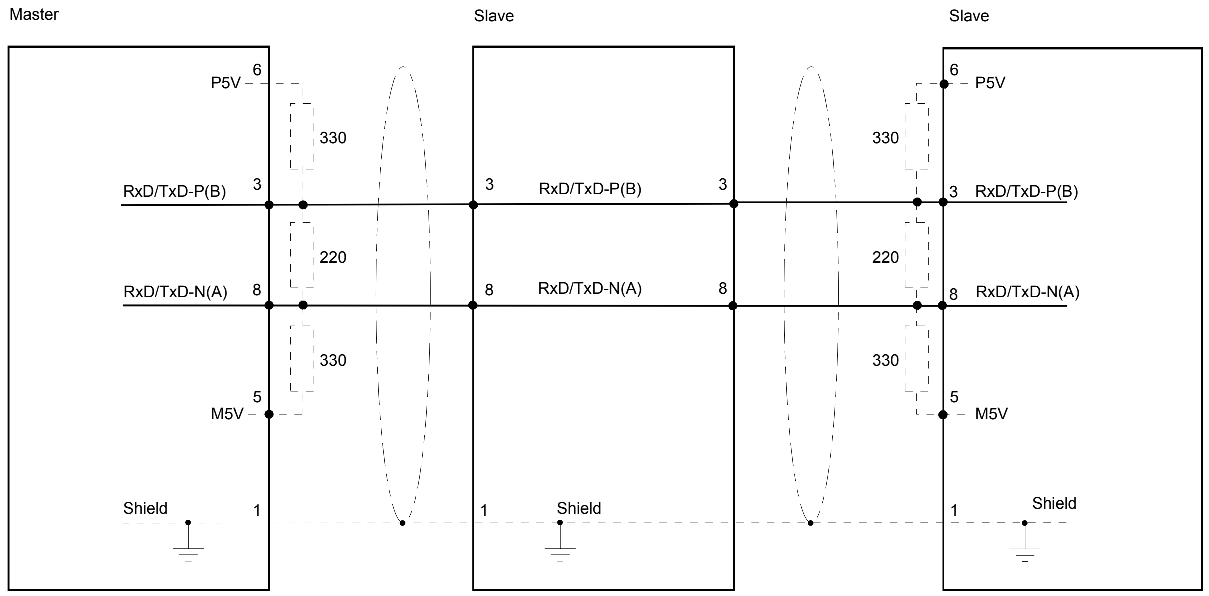
- Assemble your Profibus system using the required modules.
- Adjust the address of the bus coupler to an address that is not yet in use on your system.
- Transfer the supplied GSD-file into your system and configure the system as required.
- Transfer the configuration into your master.
- Connect the 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. The Profibus line must be terminated with ripple resistor.

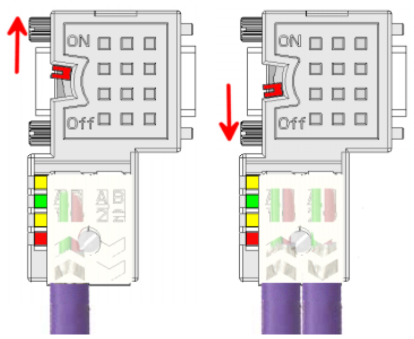
Bus connection

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



Termination with "EasyConn"

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



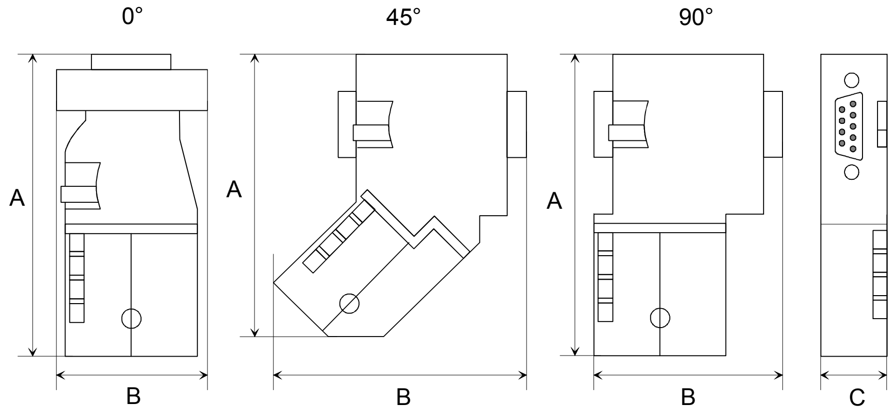
Attention!
The terminating resistor is only effective, if the connector is installed at a slave and the slave is connected to a power supply.

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

"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.



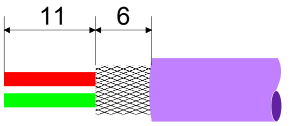
	0°	45°	90°
A	64	61	66
B	34	53	40
C	15.8	15.8	15.8

all in mm

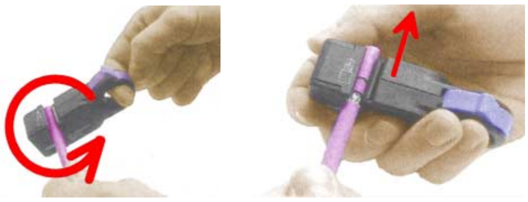
Note!

To connect this plug, please use the standard Profibus cable type A with solid wire core according to EN50170.

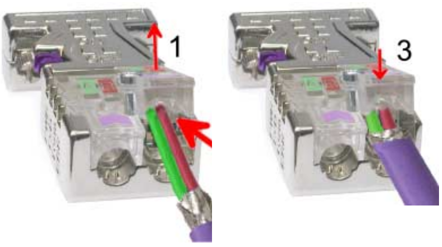
Under the order no. 905-6AA00 VIPA offers the "EasyStrip" de-isolating tool, that makes the connection of the EasyConn much easier.



Dimensions in mm



Assembly



- Loosen the screw.
- Lift contact-cover.
- Insert both wires into the ducts provided (watch for the correct line color 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).

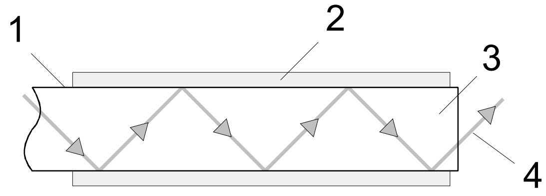
Please note: The green line must be connected to A, the red line to B!

Profibus with FO link

The fiber optic cable/optical waveguide (FO) transfers signals by means of electromagnetic waves at optical frequencies. Total reflection will occur at the point where the coating of the fiber optic cable meets the core since the refractive index of this material is lower than that of the core. This total reflection prevents the ray of light escaping from the fiber optic conductor and it will therefore travel to the end of the fiber optic cable.

The FO cable is provided with a protective coating.

The following diagram shows the Structure of a fiber optic cable:



[1] Fiber coating
[2] Protective cover
[3] Fiber core
[4] Ray of light

The fiber optic system employs pulses of monochromatic light at a wavelength of 650nm. If the fiber optic cable is installed in accordance with the manufacturers guidelines, it is not susceptible to external electrical interference. Fiber optic systems have a linear structure. Each device requires two lines, a transmit and a receive line (dual core). It is not necessary to provide a terminator at the last device.

The Profibus FO network supports a maximum of 126 devices (including the master). The maximum distance between two devices is limited to 50m.

Advantages of FO over copper cables

- wide bandwidth
- low attenuation
- no cross talk between cores
- immunity to external electrical interference
- no potential difference
- lightning protection
- may be installed in explosive environments
- low weight and higher flexibility
- corrosion resistant
- safety from eavesdropping attempts

FO cable FO connector

VIPA recommends to use FO connector and cable supplied by Hewlett Packard (HP):

HP order no.: FO cable

HFBR-RUS500, HFBR-RUD500, HFBR-EUS500, HFBR-EUD500

HP order no.: FO connector

With crimp-type assembly: HFBR-4506 (grey), HFBR-4506B (black)

Without crimp-type assembly: HFBR-4531

For more see following page.

Fiber optic cabling under Profibus

The VIPA fiber optic Profibus coupler employs dual core plastic fiber optic cable as the communication medium. Please keep the following points in mind when you connect your Profibus FO coupler: predecessor and successor must always be connected by means of a dual core FO cable.

The VIPA bus coupler carries 4 FO connectors. The communication direction is defined by the color of the connector (dark: receive line, light: send line).

When the bus has been turned on, you recognize the receive line by the light, while the darker line is the send line.

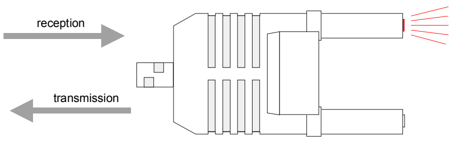
The connectors Hewlett Packard (HP) are available in two different versions:

FO connector with crimp-type assembly

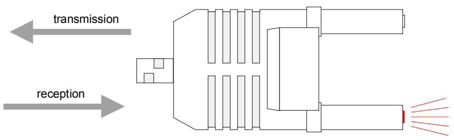
FO connector without crimp-type assembly

FO connector with crimp-type assembly

Connection for predecessor



Connection for successor



HP order no.: HFBR-4506 (gray)

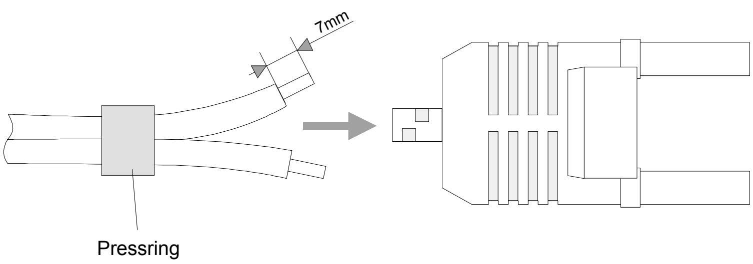
HFBR-4506B (black)

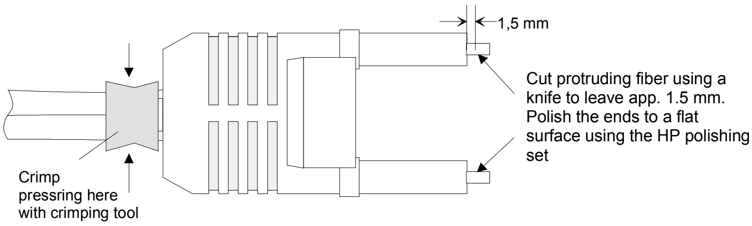
Advantages: polarity protection.

You can only install the connector so that the side of the connector shown here faces to the right.

Disadvantages: special tool required

You require a special crimping tool from Hewlett Packard (HP order no.: HFBR-4597) for the installation of the press ring required for strain relief.





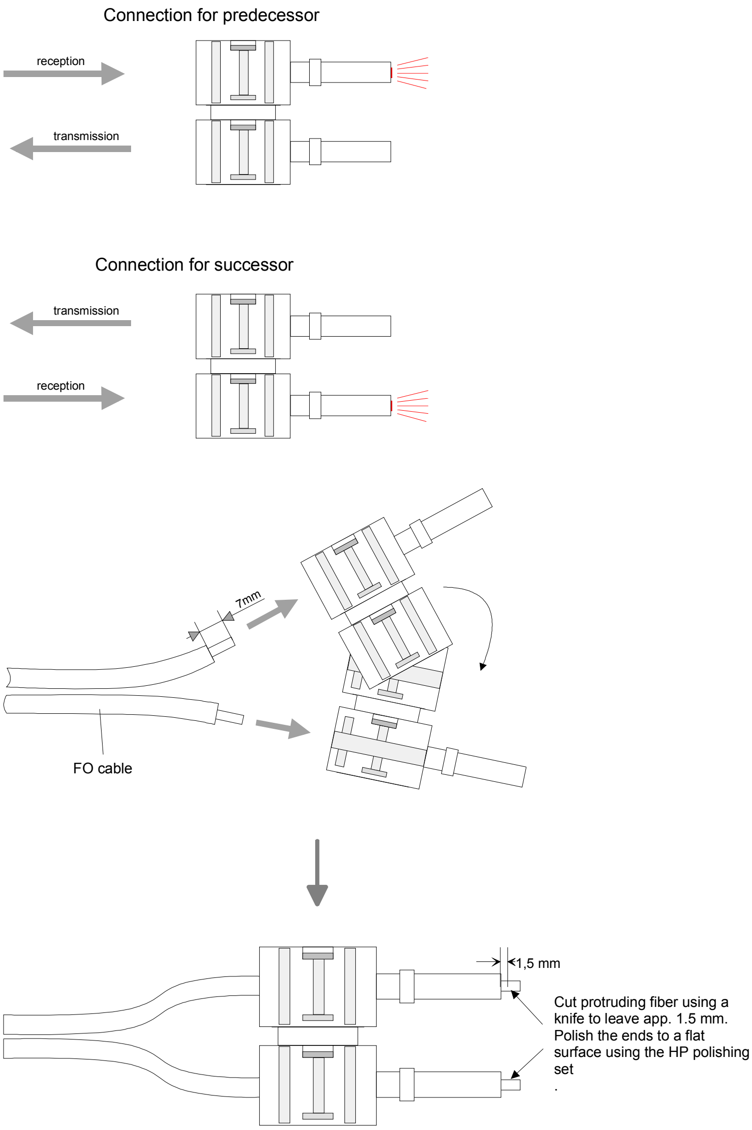
Connector installation

You install the connector by first pushing the press-ring onto the dual core FO cable. Separate the two cores for a distance of app. 5cm. Use a stripper to remove the protection cover for app. 7mm.

Insert the two cores into the plug so that the ends of the fiber optic cable protrude at the front. Keep an eye on the polarity of the cores (s.a.).

Push the press-ring onto the plug and crimp the ring by means of the crimp tool. The description of how to trim and polish of the ends of the FO cores is identical to the 2nd connector type shown below.

FO connector without crimp-type assembly



HP order no.: HFBR-4531

Advantages: no special tool required.

This shell of this type of plug is provided with an integrated strain relief.

The fiber optic cable is clamped securely when you clip the two sections of the shell together.

This system can be used to prepare simplex and duplex plugs. You can assemble a simplex plug by clipping the two sections of a shell together and a duplex plug by clipping two plugs together.

Disadvantages: no protection against polarity reversal.

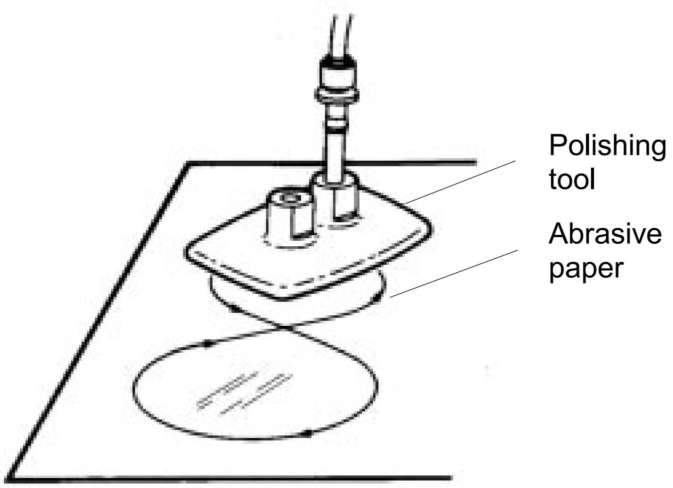
These plugs can be inserted in two positions. Please check the polarity when you have turned on the power. The light emitting fiber is the fiber for reception.

Assembling a plug:

2 complete plugs are required to assemble a duplex plug. Separate the two cores for a distance of app. 5cm. Use a stripper to remove the protection cover so that app. 7mm of the fiber is visible.

Insert the two cores into the plug so that the ends of the fiber optic cable protrude at the front. Keep an eye on the polarity of the cores (s.a.).

Cutting and polishing the ends of the FO cable



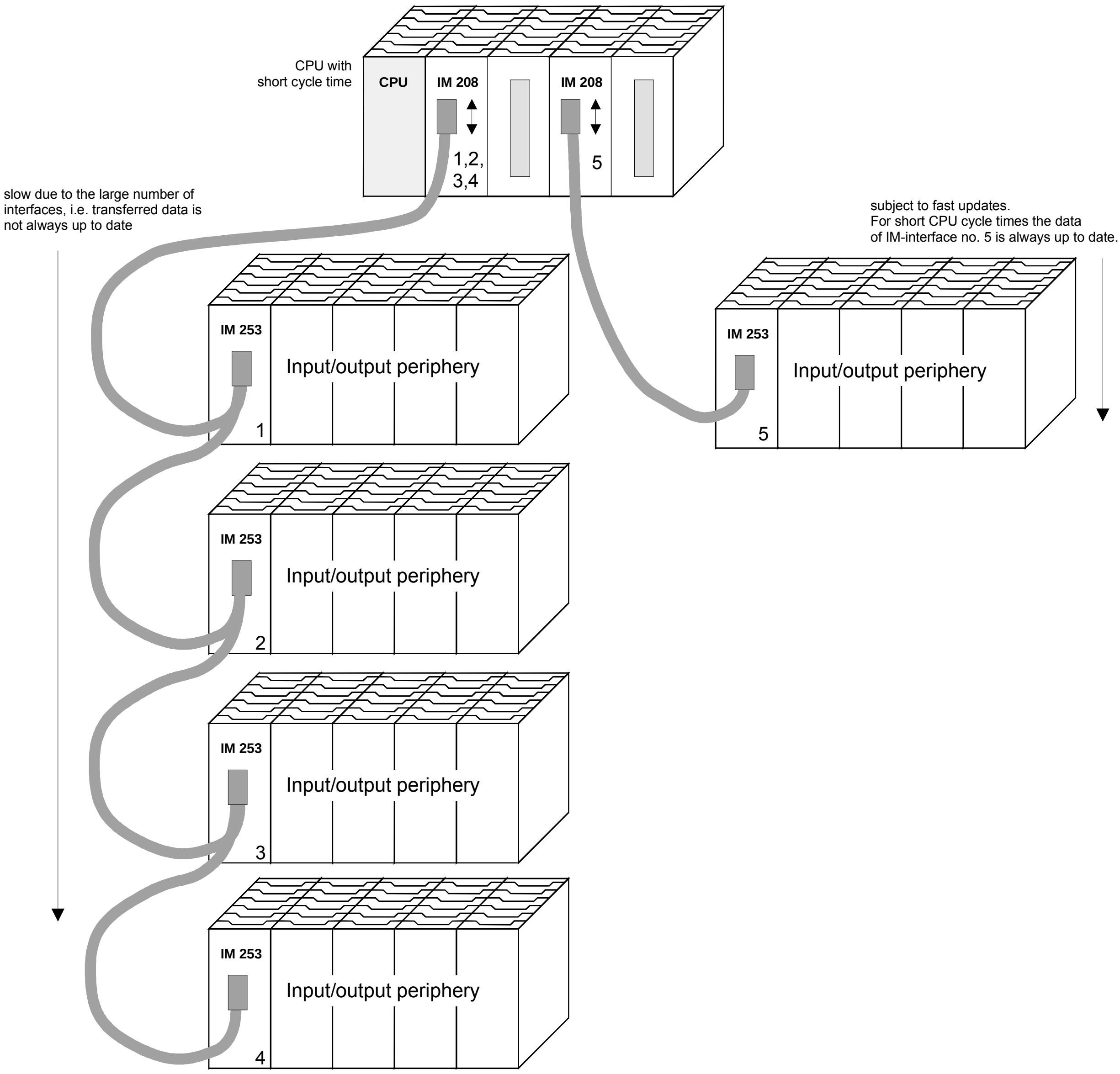
Cut protruding fiber using a knife so that app. 1.5mm are still visible. Polish the ends to a flat surface using the HP polishing set (HP order no.:HFBR-4593).

Insert the plug into the polishing tool and polish the fiber to achieve a plane surface as shown in the figure. The instructions that are included with the set contain a detailed description of the required procedure.

Example for a Profibus network

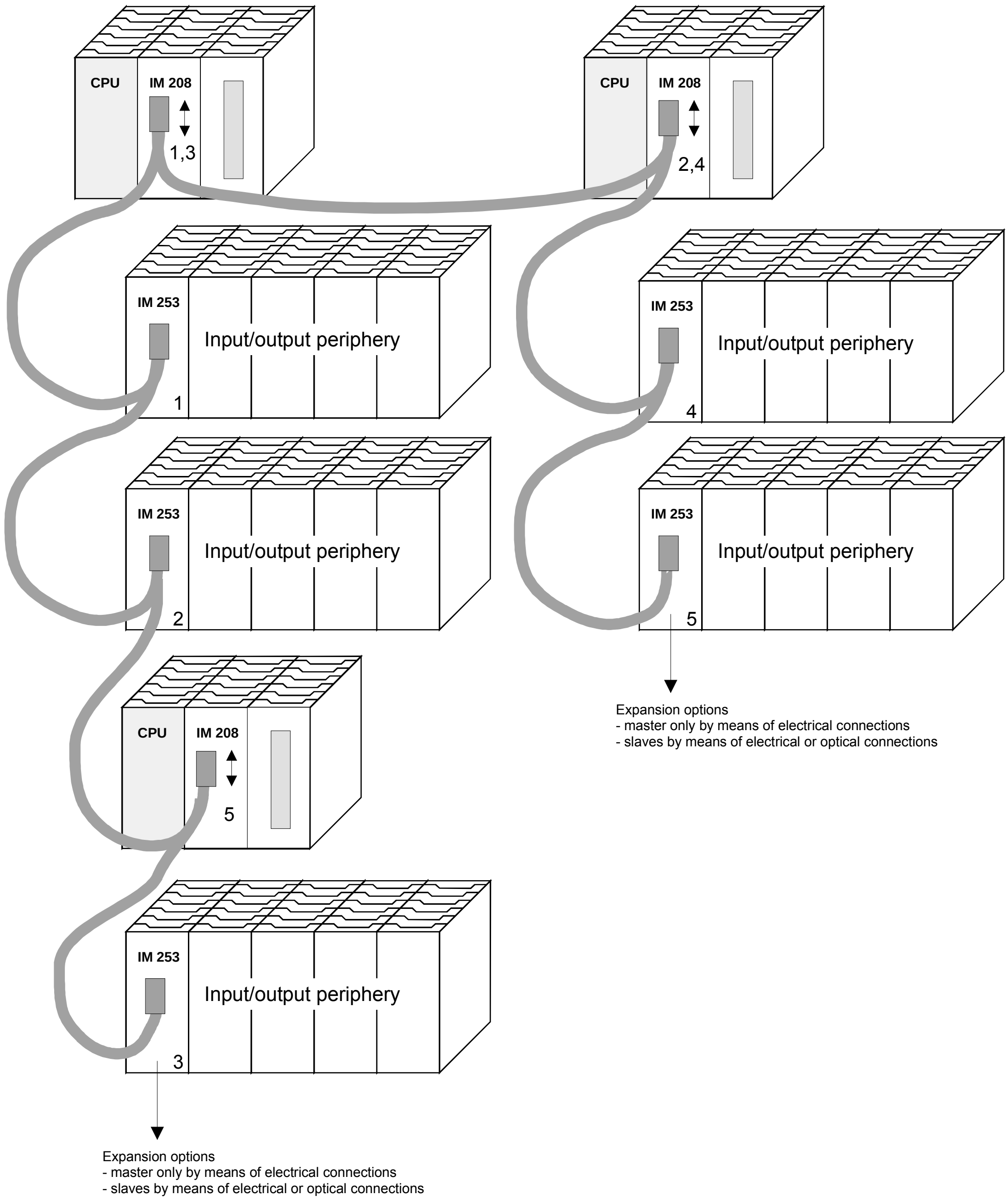
One CPU and multiple master connections

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 alarms.

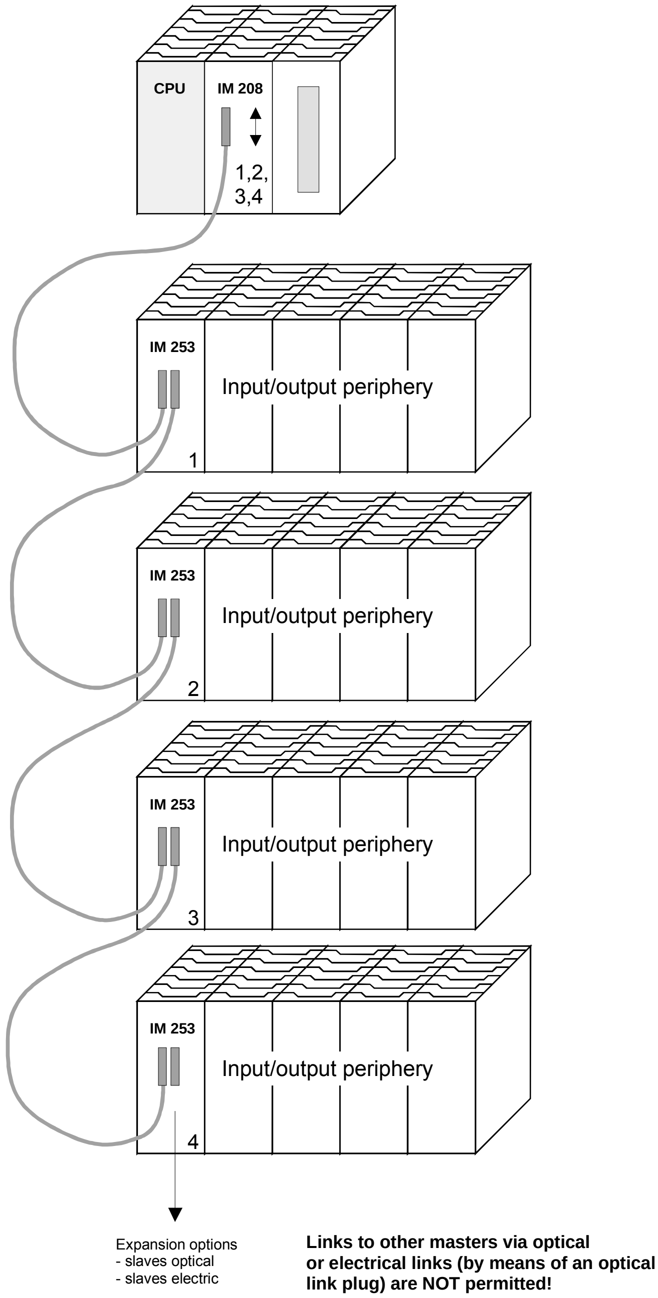


Multi master system

Multiple master connections on a single bus in conjunction with a number of slaves:

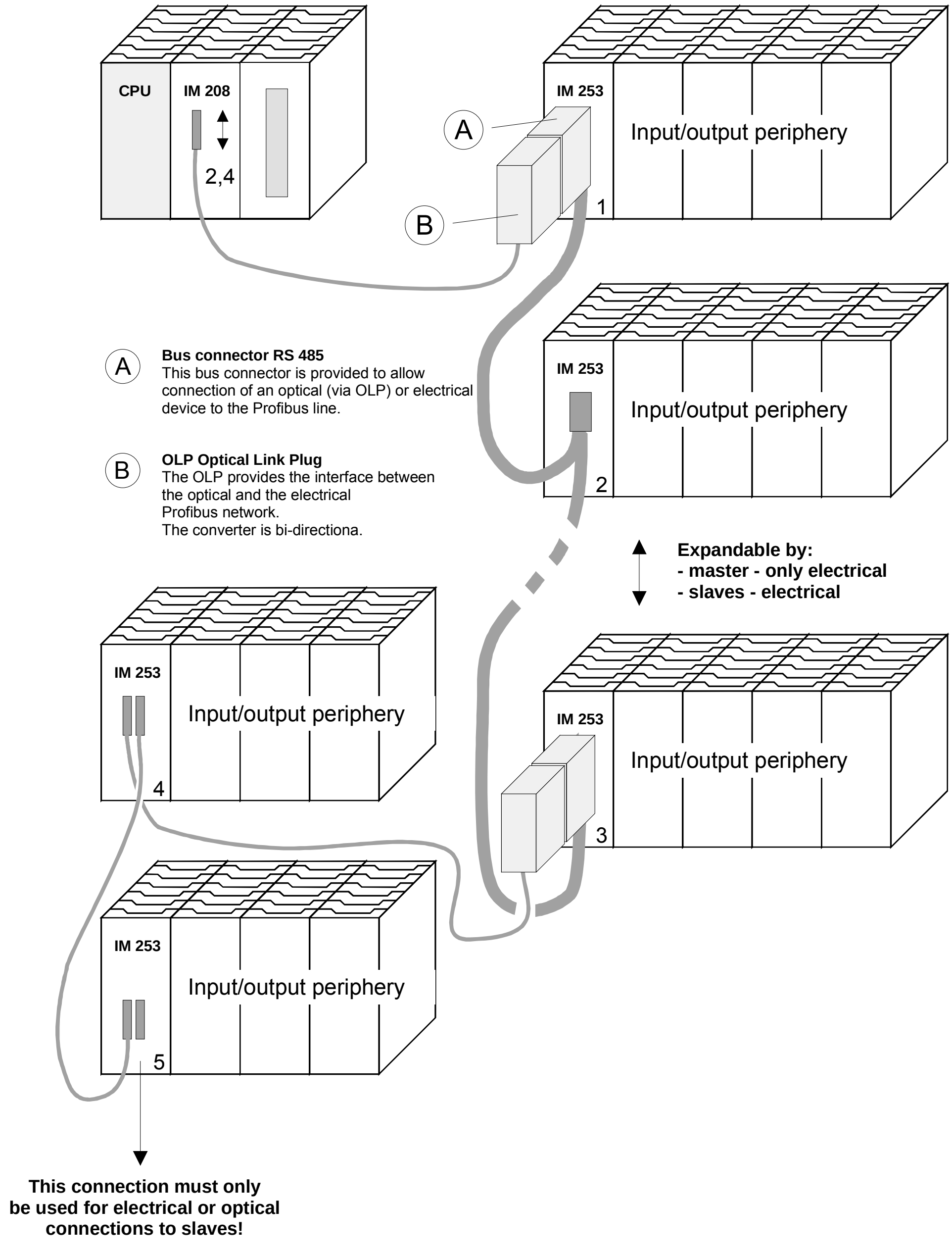


Optical Profibus



Combination of optical and electrical Profibus

In a combined fiber optical Profibus system only one converter (OLP) may be installed between any two masters!



Commissioning

Overview

- Assemble your Profibus system.
- Configure your master system.
- 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 using the wanted modules.

Every Profibus slave coupler has an internal power supply. This power supply requires an external DC 24V power supply. In addition to the circuitry of the bus coupler, the voltage supply is also used to power any modules connected to the backplane bus.

Profibus and backplane bus are galvanically isolated from each other.

Addressing

Adjust the address of every Profibus slave module as required.

Configuration in the master system

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

Transferring your project

A number of different transfer methods are employed due to the fact that a number of different hardware versions of the VIPA Profibus master modules are existing. These transfer methods are described in the master configuration guide for the respective hardware version.

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.

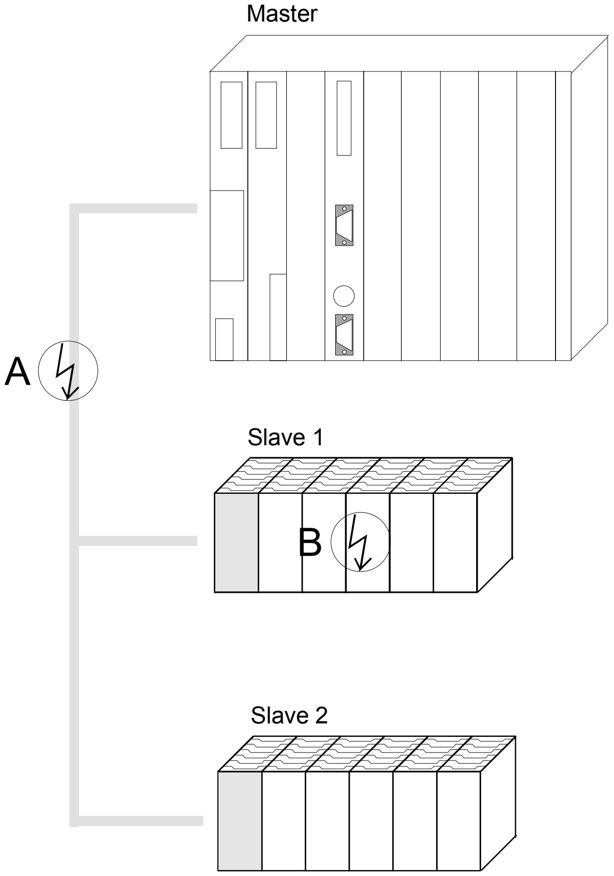
You should always keep an eye on the correct polarity!

Note!

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

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

Sample projects for Profibus communication

Example 1

Problem

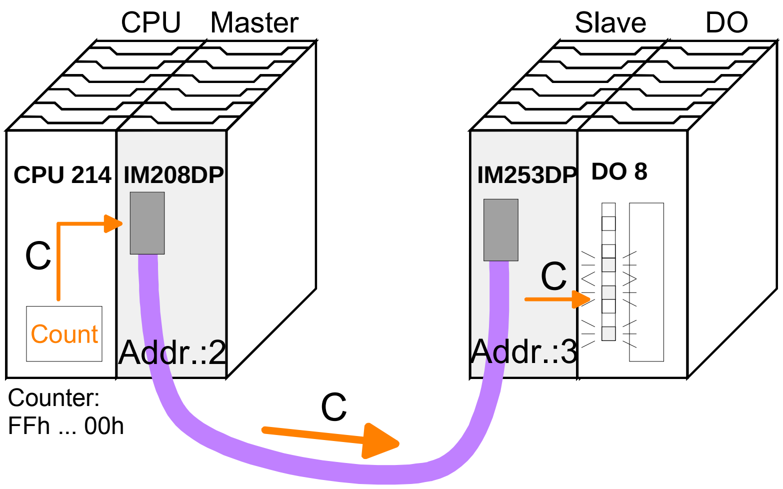
The following example describes a communication between a master and a slave system.

The master system consists of a CPU 21x (here CPU 214-1BA02) and a DP master IM 208DP. This system communicates via Profibus with a IM 253DP and an output module.

Via this system, counter values should be exchanged via Profibus and monitored at the output module. The counter values have to be created in the CPU.

Problem in detail

The CPU has to count from FFh to 00h and transfer the counter value cyclically into the output area of the Profibus master. The master sends this value to the DP slave. The received value shall be monitored at the output module (at address 0).



Project data

CPU 214 and IM 208DP (Master)

Counter value: MB 0 (FFh ... 00h)

Profibus address: 2

IM 253DP and DO (Slave)

Profibus address: 3

Output area: Address 0, length: 1Byte

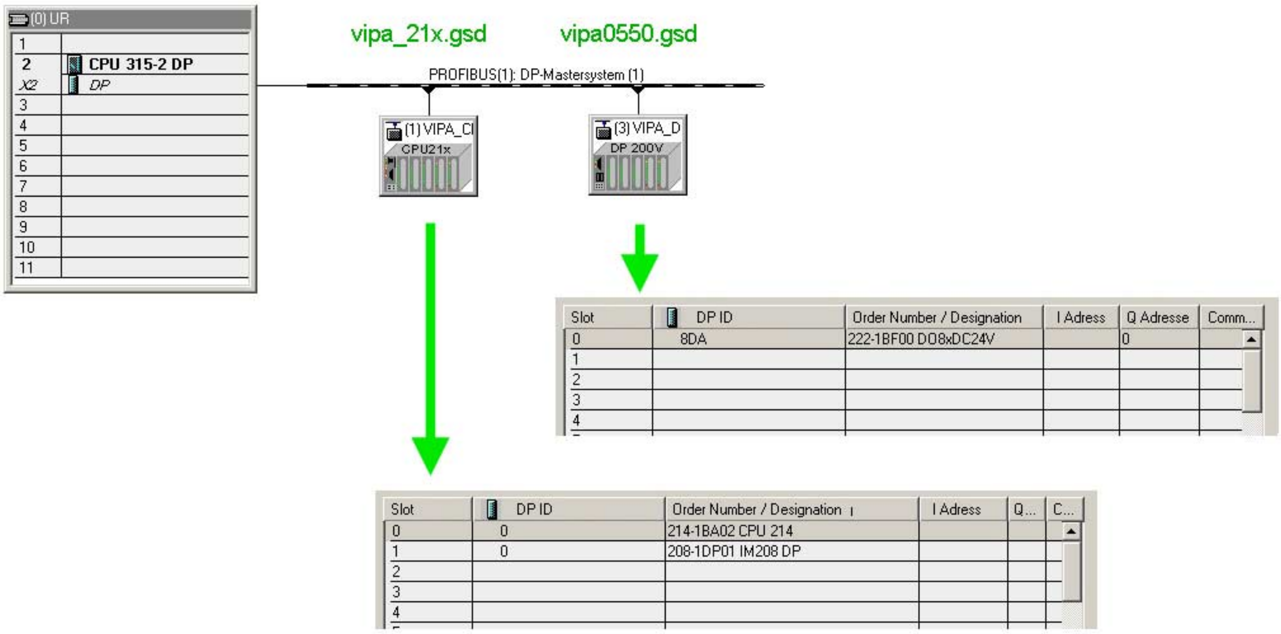
Engineering IM 208DP

To be compatible with the Siemens SIMATIC Manager, you have to execute the following steps for the System 200V:

- Start the Hardware configurator from Siemens
- Install the GSD-file vipa_21x.gsd
- Project a CPU 315-2DP with DP master (master address 2)
- Add a Profibus slave **"VIPA_CPU21x"** with address 1.
- Include the CPU **214-1BA02** at slot 0.
- Include the DP master 208-1DP01 at slot 1.

To connect your IM 253DP, you have to execute the following steps after including the GSD-file vipa0550.gsd:

- Add the Profibus slave **"VIPA_DP200V_2"** with address 3.
  You will find the DP slave in the hardware catalog from Siemens at:
  Profibus DP>Additional field devices>I/O>VIPA_System_200V
- Include the digital output module 222-1BF00 at slot 0.
- Assign the output address 0.



User application in the CPU

For the user application in the CPU, we use the OB35. The OB35 is a time OB, where the call cycle is defined in the CPU properties.

OB 35 (Time-OB)

```
L    MB   0    counter from FFh to 00h
L    1
-I
T    MB   0    remember new counter value
T    AB   0    transfer new counter value to output byte 0
               via Profibus
BE
```

The call cycle of the OB35 may be defined in the "properties" of your CPU 315-2DP at *prompter alarm*. Type for example 100ms.

Transfer and execute project

Now the programming is complete. Transfer your project into the CPU and execute the program.

- Connect your PU res. PC with your CPU via MPI.

  If your PU doesn't support MPI, you may use the VIPA "Green Cable" to establish a point-to-point connection.

  The "Green Cable" has the order number VIPA 950-0KB00 and may only be used with VIPA CPUs of the Systems 100V, 200V, 300V and 500V. For the employment, the following settings are required:

  - Choose the interface parameterization „PC Adapter (MPI) in your project engineering tool at **Options** > *Configure PU/PC interface.* If needed, you have to add this first.
  - Click on [Properties] and set the wanted COM port and the baud rate 38400 at "Local interface".
- Configure the MPI-interface of your PC.
- Via **PLC** > *Load to module* you transfer your project into the CPU.
- If you want to save your project on MMC additionally, plug-in a MMC and transfer your user application via **PLC** > *Copy RAM to ROM*.

  During the write process the "MC"-LED at the CPU is blinking. Due to the system, the completion of the write operation arrives too soon. It is only completed when the LED has been extinguished.

As soon as CPU and DP master are in RUN, the counter values are transferred via Profibus and monitored at the output module of the DP slave.

Example 2

Problem

This example shows a communication between a CPU 21x (here CPU 214-1BA02) with IM 208 DP master and a CPU 21xDP (here CPU 214-2BP02).

Via this system, counter values should be exchanged via Profibus and monitored at the output module of the respective partner.

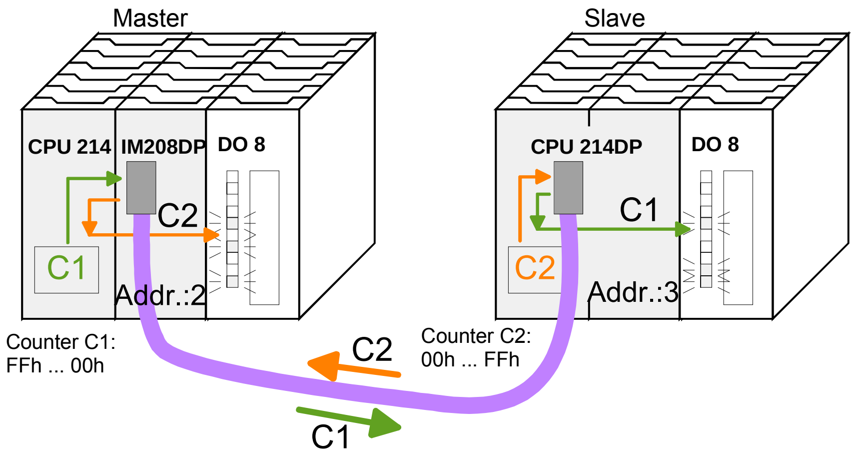
Problem in detail

The CPU 214 has to count from FFh to 00h and transfer the counter value cyclically into the output area of the Profibus master. The master sends this value to the DP slave of the CPU 214DP.

The received value shall be stored in the input periphery area of the CPU and monitored via the backplane bus at the output module (at address 0).

Vice versa, the CPU 214DP has to count from 00h to FFh, store the value in the output area of the CPU slave and transfer it to the master via Profibus.

This value is monitored at the output module of the CPU 214 (address 0).



Project data

CPU 214 and DP master

Counter value:	MB 0 (FFh ... 00h)	
Profibus address:	2	
Input area:	Address 10	Length: 2 Byte
Output area:	Address 20	Length: 2 Byte

CPU 214DP

Counter value:	MB 0 (00h...FFh)	
Input area:	Address 30	Length: 2 Byte
Output area:	Address 40	Length: 2 Byte
Parameter data:	Address 800	Length: 24 Byte (fix)
Diagnostic data:	Address 900	Length: 6 Byte (fix)
Status data:	Address 1020	Length: 2 Byte (fix)
Profibus address:	3	

Engineering CPU 214 of the DP master

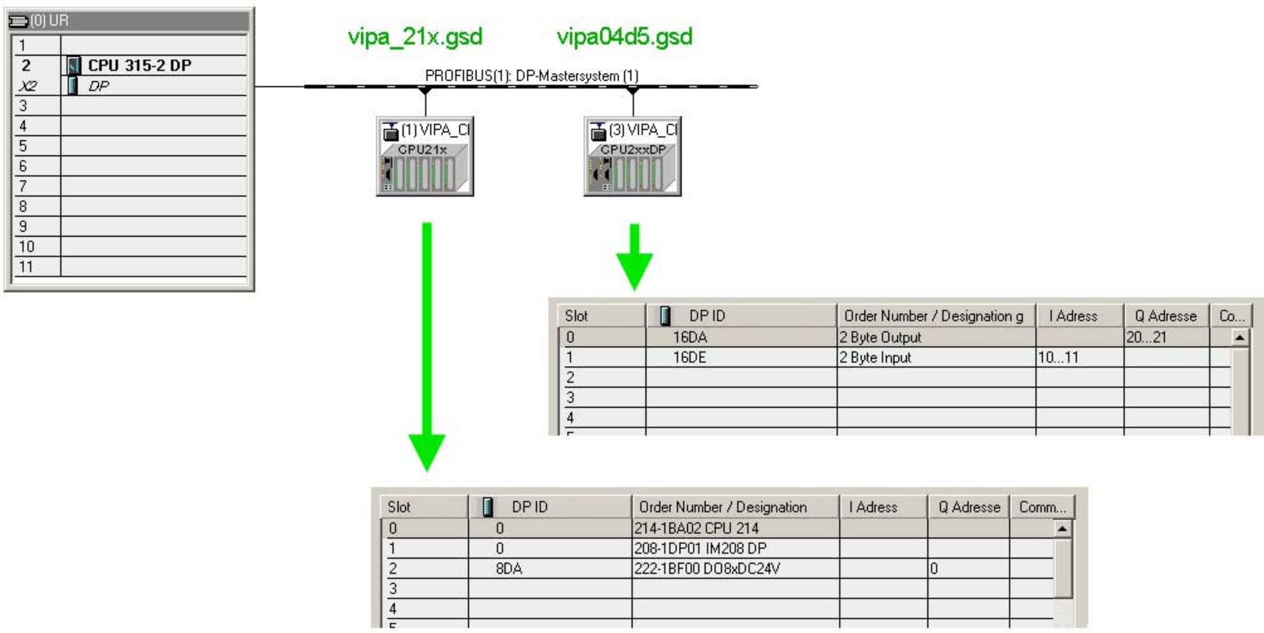
To be compatible with the STEP®7 projecting tool from Siemens, you have to execute the following steps for CPU 214 and DP master:

- Start the Hardware configurator from Siemens
- Install the GSD-file vipa_21x.gsd
- Project a CPU 315-2DP with DP master (master address 2)
- Add a Profibus slave **"VIPA_CPU21x"** with address 1.
- Include a CPU **214-1BA02** at slot 0 of the slave system
- Include the DP master 208-1DP01 (place holder) at slot 1 and include the output module 222-1BF00 at slot 2.
- Give the output module 222-1BF00 at slot 0.

Profibus link-up of the CPU 214DP

To connect your real CPU 214DP, you have to execute the following steps after including the GSD-file vipa04d5.gsd:

- Add the Profibus slave **"VIPA_CPU2xxDP"** (address 3)
- Include the "2 Byte Output" element at slot 0 and choose the output address 20.
- Include the "2 Byte Input" element at slot 1 and choose the input address 10.
- Save your project.



User application in the CPU 214

The user application in the CPU 21x has 2 tasks to execute, shared between two OBs:

- Test the communication via control byte.
  Load the input byte from Profibus and monitor the value at the output module.

OB 1 (cyclic call)

```
L    B#16#FF
T    QB  20            control byte for slave CPU
L    B#16#FE           load control value 0xFE
L    IB  10            control byte from slave
<>I                    CPU correct?
BEC                    no -> End
                       -------------------------------
                       Data transfer via Profibus
L    IB  11            load input byte 11 (output data
                       of the CPU214DP) and
T    QB   0            transfer to output byte 0
BE
```

- Read counter value from MB 0, decrement it, store in MB 0 and transfer it to the CPU 21xDP via Profibus.

OB 35 (Time-OB)

```
L    MB   0            counter from 0xFF to 0x00
L    1
-I
T    MB   0
T    QB  21            Transfer to output byte 21
                       (input data of the CPU214DP)
BE
```

Transfer project and execute

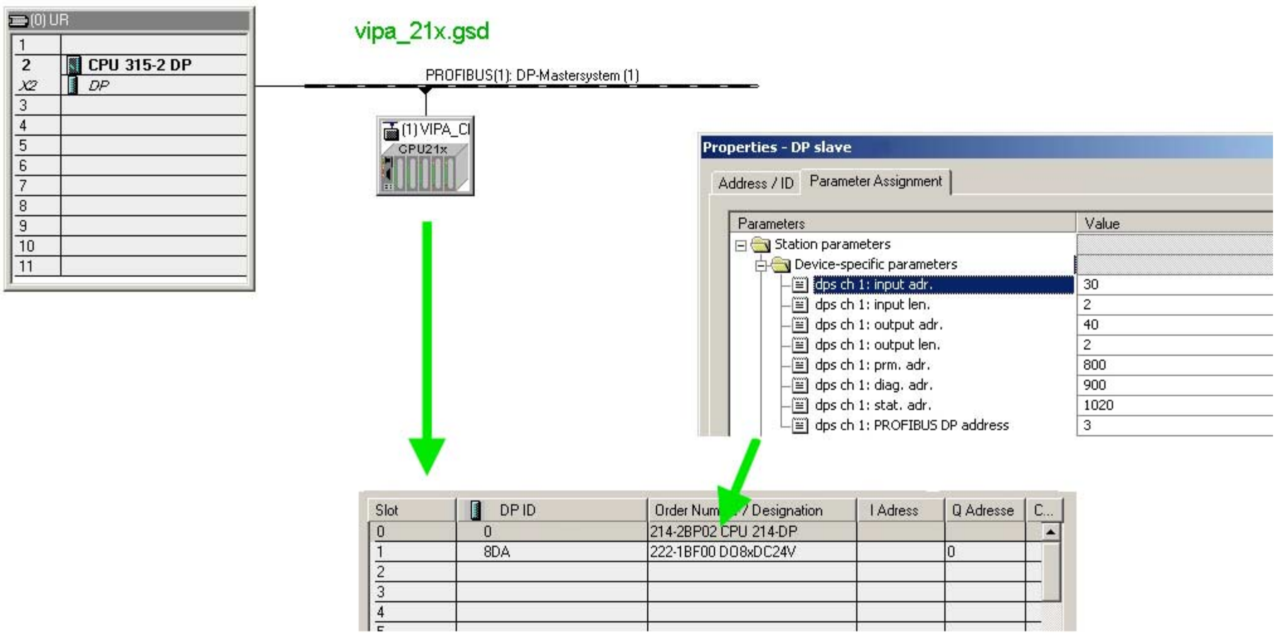
Transfer your project with the hardware configuration into the CPU and execute the program. The hardware configuration of CPU 214 and DP master is now finished.

The following pages describe the project engineering of the CPU 214DP.

Engineering CPU 214DP

To be compatible with the Siemens SIMATIC Manager, you have to execute the following steps for the CPU 214DP:

- Start the Hardware configurator from Siemens
- Install the GSD-file vipa_21x.gsd
- Project a CPU 315-2DP with DP master (master address 2)
- Add a Profibus slave **"VIPA_CPU21x"** with address 1.
- Include the CPU **214-2BP02** at slot 0
- Select the following parameters for the CPU 214DP:
  - Input Add.: 30
  - Input Length: 2
  - Output Add.: 40
  - Output Length: 2
  - Prm. Add.: 800
  - Diag. Add.: 900
  - Stat. Add.: 1020
  - Profibus DP Add.: 3
- Include the output module 222-1BF00 at slot 1 and give them the output address 0.
- Safe your project.



User application in the CPU 214DP

Like shown above, the user application has 2 tasks, shared between two OBs:

- Load the input byte from the Profibus slave and monitor the value at the output module.

OB 1 (cyclic call)

```
L    PIW 1020           load status data and store it
T    MW 100             in the bit memory word

AN   M  100.5           commissioning by DP master
BEC                     successful? no -> End

A    M  101.4           receive data valid?
BEC                     no -> End
L    B#16#FF            load control value and compare with
L    PIB  30            control byte (1st input byte)
<>I
BEC                     receive data not valid


L    B#16#FE            control byte for Master-CPU
T    PQB  40
                        -------------------------------
                        Data transfer via Profibus

L    PIB  31            load periphery byte 31 (input
                        data from Profibus slave) and
T    IB   0             transfer into output byte 0

BE
```

- Read counter value from MB 0, increment it, store it in MB 0 and transfer it via Profibus to CPU 214.

OB 35 (Time-OB)

```
L    MB   0             counter from 0x00 to 0xFF
L    1
+I
T    MB   0

T    PQB  41            Transfer counter value to
                        periphery byte 41 (Output data
                        of the Profibus slave)
BE
```

Transfer project and execute

Transfer your project with the hardware configuration into the CPU (see Example 1) and execute the program.

As soon as the CPUs and DP master are in RUN, the counter values are transferred via Profibus and monitored at the according output module.

Technical data

Profibus DP master

IM 208DP

Electrical data	VIPA 208-1DP01
Power supply Current consumption Power loss Isolation Status indicators Connections/interfaces	via backplane bus max. 450mA 2 W ≥ AC 500V via LEDs on the front 9pin D-type socket Profibus connector
Profibus interface	
Connection Network topology Medium Data transfer rate Total length Max. no. of stations	9pin D-type socket Linear bus, active bus terminator at both ends, tap lines are permitted. Screened twisted pair cable, under certain conditions unscreened lines are permitted. 9.6kBaud to 12MBaud 100m without repeaters for 12MBaud, 1000m with repeaters 32 stations in any segment without repeaters. Extendible to 126 stations when using repeaters.
Combination with peripheral modules	
max. no of slaves max. no. of input bytes max. no. of output bytes	125 256 (1024 since V3.0.0) 256 (1024 since V3.0.0)
Dimensions and weight	
Dimensions (WxHxD) in mm Weight	25.4x76x78 110g

IM 208DPO

Electrical data	VIPA 208-1DP11
Power supply	via backplane bus
Current consumption	max. 450mA
Power loss	2 W
Isolation	≥ AC 500V
Status indicator	via LEDs on the front
Connections/interfaces	2pin socket for fiber optic cable Profibus interface
Profibus interface	
Connection	2pin socket for fiber optic cable
Network topology	Linear structure with dual FO cable, no bus terminator required
Medium	dual-core fiber optic cable
Data transfer rate	12MBaud
Total length	at POF-FO: max. 50m between stations at HCS-FO: max. 300m between stations
Max. no. of stations	17 stations incl. Master
Combination with peripheral modules	
max. no of slaves	16
max. no. of input bytes	256 (1024 since v3.0.0)
max. no. of output bytes	256 (1024 since v3.0.0)
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x78
Weight	110g

Max. number of stations

The maximum number of DPO participants depends on the baud rate. The table shows the max. number incl. master:

Baud rate	max. no. of participants
≤ 1.5MBaud	17
3MBaud	15
6MBaud	7
12MBaud	4

Profibus DP slave

IM 253DP

Electrical data	VIPA 253-1DP00 (DP-V0)	VIPA 253-1DP01 (DP-V0/V1)	VIPA 253-1DP31 - ECO (DP-V0/V1)
Power supply	DC 24V (20.4 ... 28.8V) ext. power supply at front		
Current consumption	max. 1A		max. 0.3A
Output current backplane bus	max. 3.5A		max. 0.8A
Isolation	≥ AC 500V		
Status indicator	via LEDs on the front		
Connections/interfaces	9pin D-type socket	Profibus connector	
Profibus interface			
Connection	9pin D-type socket		
Network topology	Linear bus, active bus terminator at both ends, tap lines are permitted.		
Medium	Screened twisted pair cable, under certain conditions unscreened lines are permitted.		
Data transfer rate	9.6kBaud to 12MBaud		
Total length	100m without repeater for 12MBaud; 1000m with repeater		
Max. no. of stations	32 stations in any segment without repeater. Extendible to 126 stations when using repeaters.		
Diagnostic functions			
Standard diagnostic	The last 100 results are stored in Flash-ROM.		
Extended diagnostic	-	possible	
Data			
Input data	max. 152Byte	max. 244Byte	
Output data	max. 152Byte	max. 244Byte	
Combination with peripheral modules			
max. no of modules*	32		8
max. digital I/Os	32		8
max. analog I/Os	16		8
Dimensions and weight			
Dimensions (WxHxD) in mm	25,4x76x78		
Weight	80g		

*) depends on the power consumption

IM 253DPO

Electrical data	VIPA 253-1DP10 (DP-V0)	VIPA 253-1DP11 (DP-V0/V1)
Power supply Current consumption (in no-load operation) Current consumption (rated value) Output current backplane bus Power loss Isolation Status indicator Connections/interfaces	DC 24V (20.4 ... 28.8V), ext. power supply at front 50mA 1A max. max. 3.5A 2.5W ≥ AC 500V via LEDs on the front 4pole FO socket	 Profibus connection
Profibus interface		
Connection Network topology Medium Data transfer rate Total length Max. no. of stations	4pole socket for fiber optic cable Linear structure with dual FO cable, no bus termination required dual-core fiber optic cable 12MBaud at POF-FO: max. 50m between stations at HCS-FO: max. 300m between stations 17 stations incl. master (see below)	
Diagnostic functions		
Standard diagnostic Extended diagnostic	The last 100 results are stored in Flash-ROM. no	 possible
Data		
Input data	max. 152Byte	max. 244Byte
Output data	max. 152Byte	max. 244Byte
Combination with peripheral modules		
max. no of modules max. digital I/Os max. analog I/Os	32 (depending on current consumption) 32 16	
Dimensions and weight		
Dimensions (WxHxD) in mm Weight	25.4x76x78 80g	

Max. number of stations

The maximum number of DPO participants depends on the baud rate. The table shows the max. number incl. master:

Baud rate	max. no. of participants
≤ 1.5MBaud	17
3MBaud	15
6MBaud	7
12MBaud	4

Profibus DP slave (redundant)

IM 253DPR (DP-V0)

Electrical data	VIPA 253-2DP50
Power supply Current consumption (in no-load operation) Current consumption (rated value) Output current backplane bus Power loss Isolation Status indicator Connections/interfaces 2 channels	DC 24V (20.4 ... 28.8V), ext. power supply at front 50mA 1A max. max. 3.5A 2.5W ≥ AC 500V via LEDs on the front 9pin D-type socket (2x) Profibus connector DP1 / DP2
Profibus interface	
Connection Network topology Medium Data transfer rate Total length Max. no. of stations	9pin D-type socket (2x) Linear bus, active bus terminator at both ends, tap lines are permitted. Screened twisted pair cable, under certain conditions unscreened lines are permitted. 9.6kBaud to 12MBaud (automatic adjustment) 100m without repeater for 12MBaud; 1000m with repeater 32 stations in any segment without repeater. Extendible to 126 stations when using repeaters.
Diagnostic functions	
Standard diagnostic Extended diagnostic	The last 100 results are stored in Flash-ROM. -
Combination with peripheral modules	
max. no of modules max. digital I/Os max. analog I/Os	32 (depending on current consumption) 32 16
Dimensions and weight	
Dimensions (WxHxD) in mm Weight	50.8x76x78 120g

Profibus DP slave (combination module)

IM 253DP
DO 24xDC 24V
DP-V0

Electrical data	VIPA 253-2DP20
Power supply Current consumption Output current backplane bus Isolation Status indicator Connections/interfaces	DC 24V (20.4 ... 28.8V), ext. power supply at front max. 5A max. 3.5A ≥ AC 500V via LEDs on the front 9pin D-type socket Profibus connector
Profibus interface	
Connection Network topology Medium Data transfer rate Total length Max. no of stations	9pin D-type socket Linear bus, active bus terminator at both ends. Screened twisted pair cable, under certain conditions unscreened lines are permitted. 9.6kBaud to 12MBaud (automatic adjustment) 100m without repeaters for 12MBaud; 1000m with repeaters 32 stations in any segment without repeaters. Extendible to 126 stations when using repeaters.
Diagnostic functions	
Standard diagnostic Extended diagnostic	The last 100 results are stored in Flash-ROM. -
Combination with peripheral modules	
max. no of modules max. digital I/Os max. analog I/Os	31 (depending on current consumption) 31 16
Output unit	
Number of outputs Nominal load voltage Output current per channel Status indicator	24 DC 24V (20.4...28.8V) supplied internally via Profibus coupler 1A (sum current max. 4A) Power (PW) fuse OK, Error (ER) short circuit, overload
Programming data	
Output data	4Byte (3Byte are used)
Dimensions and weight	
Dimensions (WxHxD) in mm Weight	50.8x76x78 150g

Chapter 4 Interbus

Overview

This chapter contains all the information that you require to connect your System 200V periphery to Interbus.

A description of the Interbus basics is followed by details of the Interbus coupler, its installation and commissioning.

The chapter is concluded by the technical data.

Below follows a description of:

- System overview and Interbus basics
- Hardware structure, deployment and commissioning of the Interbus coupler
- Technical data

Content

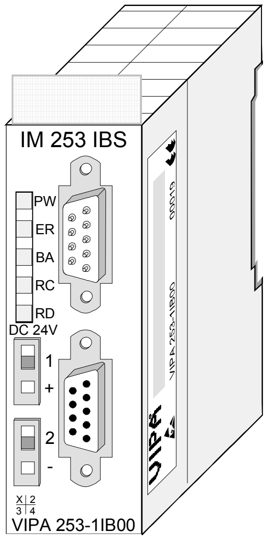
Topic **Page**

Chapter 4 Interbus....................4-1
System overview....................4-2
Basics....................4-3
IM 253IBS - Interbus coupler - Structure....................4-7
Connection to Interbus....................4-10
Deployment with Interbus....................4-11
Commissioning....................4-15
Technical data....................4-18

System overview

You can use the VIPA Interbus slave to connect up to 16 input and 16 output modules of the System 200V to your Interbus.

At present one Interbus slave module is available from VIPA.



Order data

Order number	Description
VIPA 253-1IB00	Interbus Slave

Basics

General

Interbus is a pure master/slave system that has very few protocol overheads. For this reason it is well suited for applications on the sensor/actuator level. Interbus was developed by PHOENIX CONTACT, Digital Equipment and the Technical University of Lemgo during the 80s. The first system components became available in 1988. To this day the communication protocol has remained virtually unchanged. It is therefore means that it is entirely possible to connect devices of the first generation to the most recent master interfaces (generation 4).

Interbus for sensor and actuator level

The widespread use of Interbus for sensor/actuator level applications may be ascribed to the relatively simple interfacing requirements that are supported by protocol driver chips. These reduce the number of external components required for direct input or output interfacing to a minimum.

Interbus devices are subject to the DIN standard 19258 that defines levels 1 and 2 of the protocol amongst others.

Interbus as shift register

The Interbus system is designed as a ring-type network with a central master-slave access procedure. It has the structure of a distributed shift register. The different registers of the devices connected to the ring are a portion of this shift register. The master shifts the data through this shift register. The ring structure of the network permits simultaneous transmission and reception of data. Data may be sent in both directions on the ring, which uses a single cable.

ID register

Every Interbus module has an ID register (identification register). This register contains information on the type of module, the number of input and output registers as well as status and error flags.

Interbus master

The Interbus coupler can be used to control the peripheral modules of the System 200V via Interbus. In this case the bus coupler replaces the CPU. The Interbus master reads and writes data from/to inputs and outputs respectively. The master is the link to other systems. Every master can control a maximum of 4096 input/output points. These may be located on the local bus or they may be distributed amongst secondary structures connected by means of bus couplers.

It is possible to connect remote ring systems to the main ring to provide a structured system. These remote ring systems are connected by means of bus terminal modules. You can also use these bus terminal modules for long distance communications.

Restrictions on the data capacity

The hardware overhead for Interbus devices increases in proportion with the width of the data. It is for this reason that the maximum data width was limited to 20Byte input data and 20Byte output data.

Secondary Interbus segments (peripheral busses) can be connected or disconnected by means of the respective bus coupler. For this reason the bus can remain operational even if a fault occurs on a peripheral bus connection. The faulty segment can be disconnected from the bus.

Operating modes

Interbus has two modes of operation:

- ID cycle
  An ID cycle is issued when the Interbus system is being initialized and also upon request. During the ID cycle the bus master reads the ID register of every module connected to the bus to generate the process image.
- Data cycle
  The actual transfer of data occurs during the data cycle. During the data cycle the input data from the registers of all devices is transferred to the master and the output data is transferred from the master to the devices. This is a full duplex data transfer.

Communication medium

Although Interbus appears to have a simple linear structure (a single line linking the master with every module), it has the structure of a ring that includes the outbound line and the return line in a single cable. The last device on the ring closes the loop. On most devices this is an automatic function that occurs when no further line segments are connected.

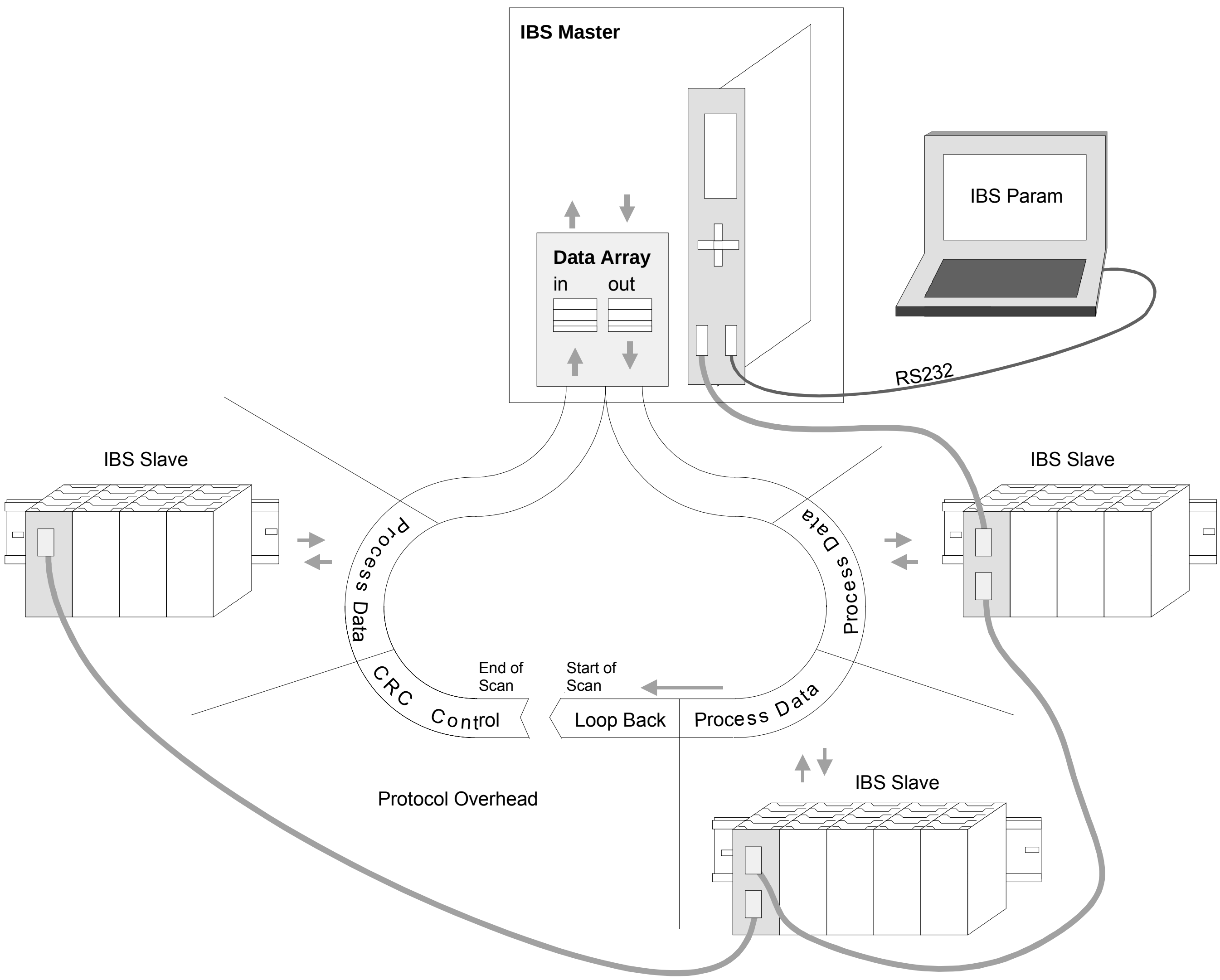
The physical level of Interbus is based upon the RS422 standard. The signals are connected by means of twisted pair lines. The outbound signal as well as the return signal of Interbus is re-routed via the same cable and every connected station. Communications between 2 devices require a 5core cable due to the ring-based structure and the common logic ground. At a data communications rate of 500kBaud two adjacent stations on the ring may be located at a distance of no more than 400m. The integral repeater function of every device on the bus allows a total distance of up to 13km. The maximum number of devices on the bus is limited to 512.

Process data transfer

Interbus is based upon a ring structure that operates as a cyclic shift register. Every Interbus module inserts a shift register into the ring. The number of I/O points supported by the module determines the length of this shift register. A ring-based shift register is formed due to the fact that all the devices are connected in series and that the output of the last shift register is returned to the bus master. The length and the structure of this shift register depend on the physical construction of the entire Interbus system.

Interbus operates by means of a master-slave access method where the master also provides the link to any high-level control system. The ring-structure includes all connected devices actively in a closed communication loop.

In comparison to client-server protocols where data is only exchanged when a client receives a properly addressed command, Interbus communications is cyclic in nature and data is exchanged at constant intervals. Every data cycle addresses all devices on the bus.



Transfer of control and inspection information

Process data words also contain control and inspection information. This information is only transferred once at the beginning or at the end of the peripheral data of any data cycle. This is why this system is also referred to as a cumulative frame procedure.

Communication principle

The communication principle is independent of the type of data being transferred:

Process data that must be transferred to the periphery is stored in the output buffer of the master in the same sequence as the output stations are connected to the bus. The transfer occurs when the master shifts the "loop-back word" through the ring. Following the loop-back word, all the output data is placed on the bus. This means that the data is shifted through the shift register. The information from the process is returned as input data to the input buffer of the master at the same time as the output data is being sent.

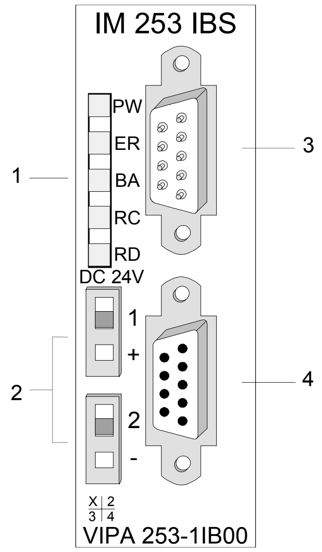
The output data is located at the correct position in the shift registers of the different stations when the entire cumulative frame telegram has been sent and read back again. At this point, the master issues a special control command to the devices on the bus to indicate the end of the data transfer cycle.

When the data check sequence has been processed, output data for the process is transferred from the shift registers. This is stored in the devices connected to the bus and transferred to the respective periphery. At the same time, new information is read from the periphery into the shift registers of the input devices in preparation for the next input cycle. This procedure is repeated on a cyclic basis. This means that the input and output buffers of the master are also updated cyclically. Interbus data communications is therefore full duplex in nature; i.e. both input data and output data are transferred during a single data cycle.

The shift register structure eliminates the need for addresses for every device as is common in other fieldbus systems. The address is defined by the location of the device in the ring.

IM 253IBS - Interbus coupler - Structure

Structure



[1] LED status indicators
[2] Power supply connector for the external 24V supply
[3] Interbus plug inbound interface
[4] Interbus socket outbound interface

Components

LEDs

The module has a number of LEDs are available for diagnostic purposes on the bus. The following table explains the purpose and the color of the different LEDs.

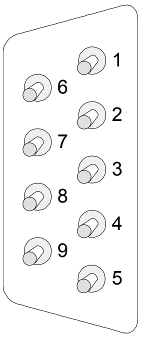
Label	Color	Description
PW	green	Power LED
		Indicates that the supply voltage is available.
ER	red	Error
		Application error.
BA	green	Bus active
		The BA LED (bus active) indicates an active Interbus data transfer.
RC	green	Remote bus Check
		The RC LED (remote bus Check) indicates that the connection to the previous Interbus device is OK (on) or that it has been interrupted (off).
RD	red	Remote bus disabled
		The RD LED (remote bus Disabled) indicates that the outbound remote bus has been disabled.

Jacks and plugs

The interfaces for the inbound and the outbound bus lines are located on the front of the module. These consist of 9pin D-type connectors.

The following diagram shows the pin assignment for this interface:

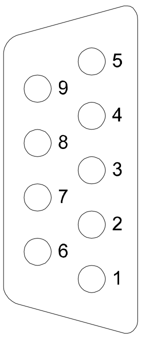
Inbound bus line (9pin D-type plug)



Pin	Assignment
1	DO
2	DI
3	GND1
4	GND *)
5	n.c.
6	/DO
7	/DI
8	+5V*) (90 mA)
9	reserved

*) power for the fiber optic converter.
This voltage is not isolated galvanically!

Outbound bus line (9pin D-type socket)



Pin	Assignment
1	DO
2	DI
3	GND
4	reserved
5	+ 5V (90 mA)
6	/DO
7	/DI
8	reserved
9	RBST

Power supply

The Interbus coupler has an internal power supply. This power supply requires an external voltage of DC 24V. In addition to the internal circuitry of the bus coupler, the supply voltage is also used to power any devices 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.

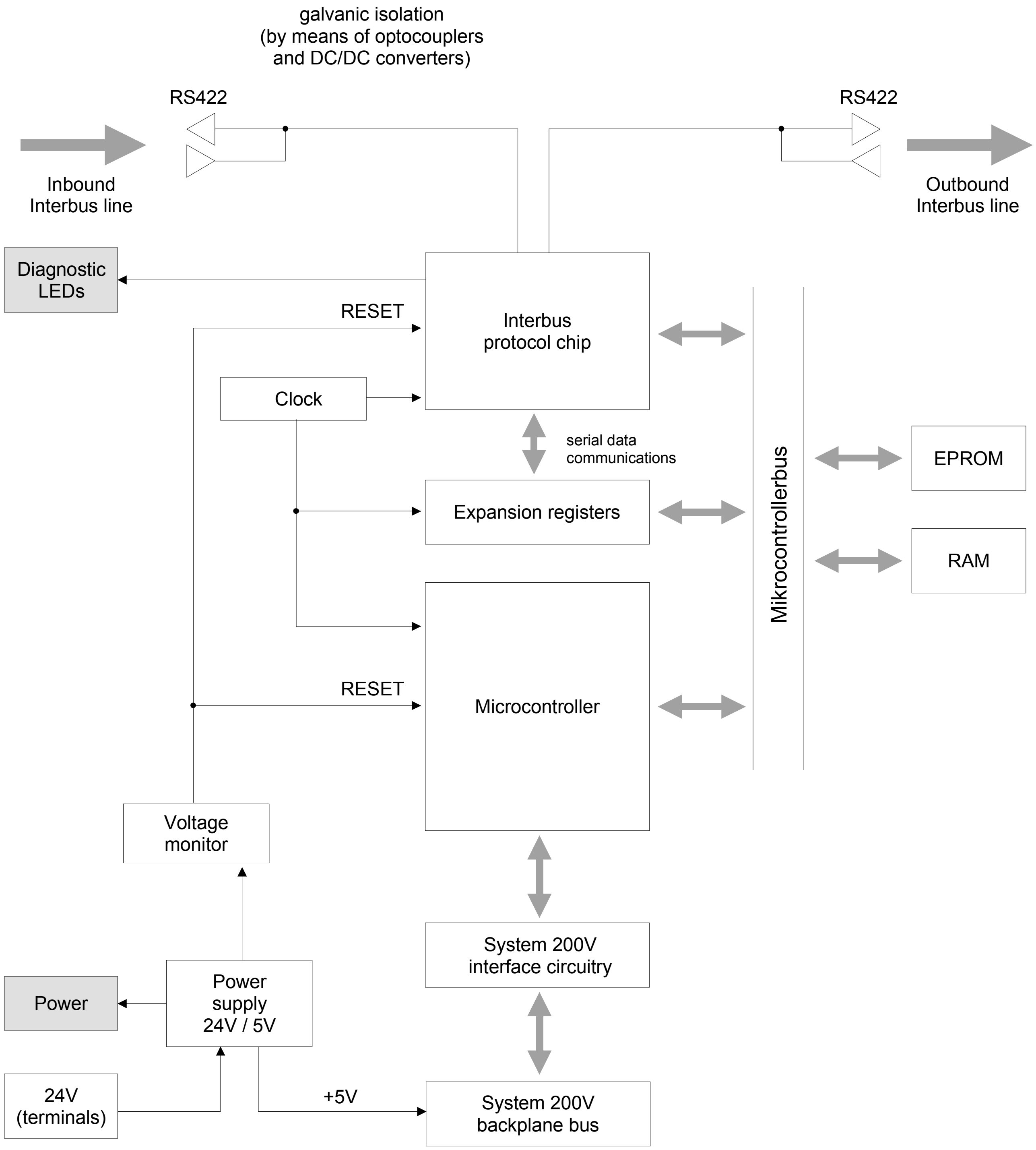
The power supply is protected against reverse polarity.

Interbus and the backplane bus are isolated from each other.

Note!
Please pay attention to the polarity of the power supply!

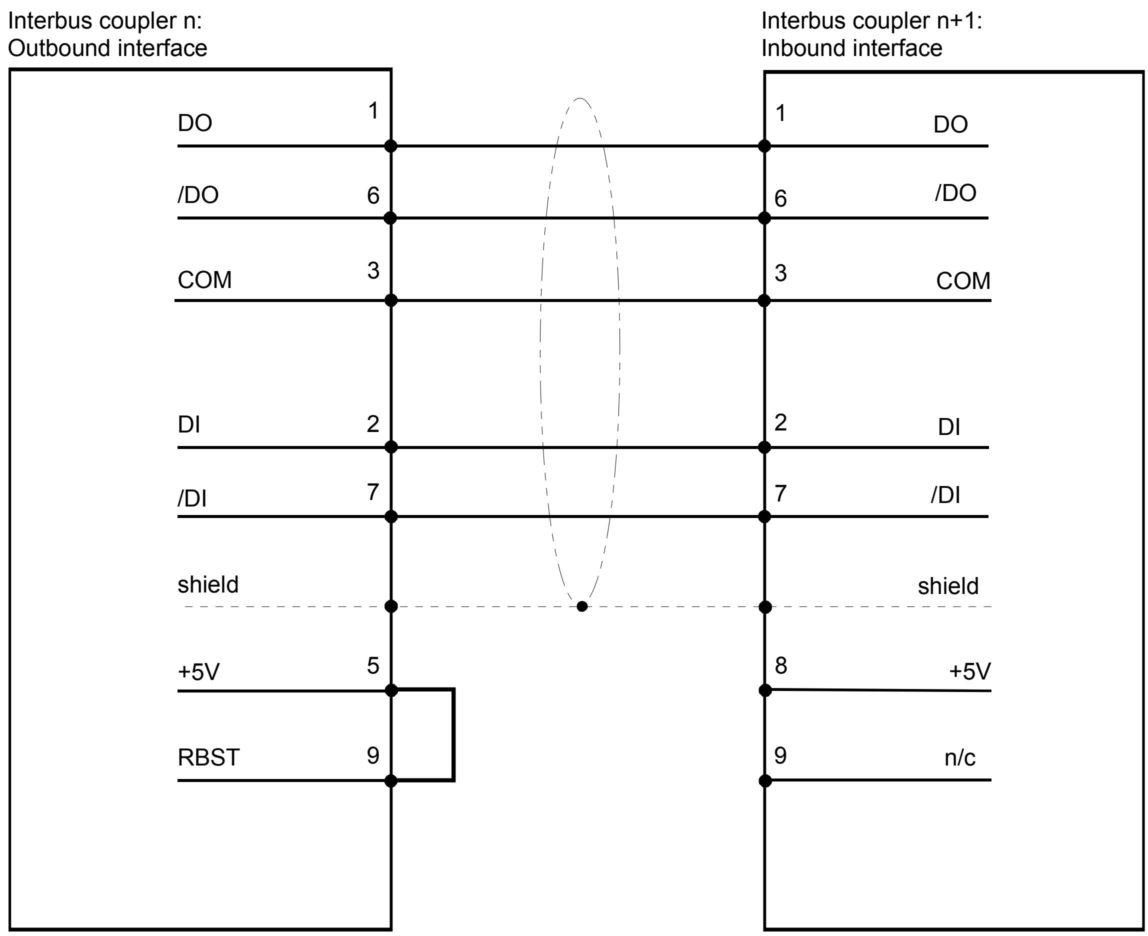
Block diagram

The following block diagram shows the hardware structure of the bus coupler:



Connection to Interbus

Interbus wiring requirements



Isolation

Due to the fact that Interbus remote bus segments can be distributed over large areas, it is necessary that individual segments are isolated galvanically to prevent problems that could be caused by potential differences. However, according to the recommendations of the Interbus club, it is sufficient to provide galvanic isolation between inbound remote bus interfaces and the remainder of the circuitry. For this reason the outbound remote bus interface is at the same potential as the rest of the circuitry and the backplane bus.

Use metallic covers for plugs and apply the screen of the cable to the plug case.

Note!

Please ensure that the link between pins 5 and 9 is installed on the plug for ”subsequent modules” as any subsequent slaves would not be detected if the link was not present!

Deployment with Interbus

Process image

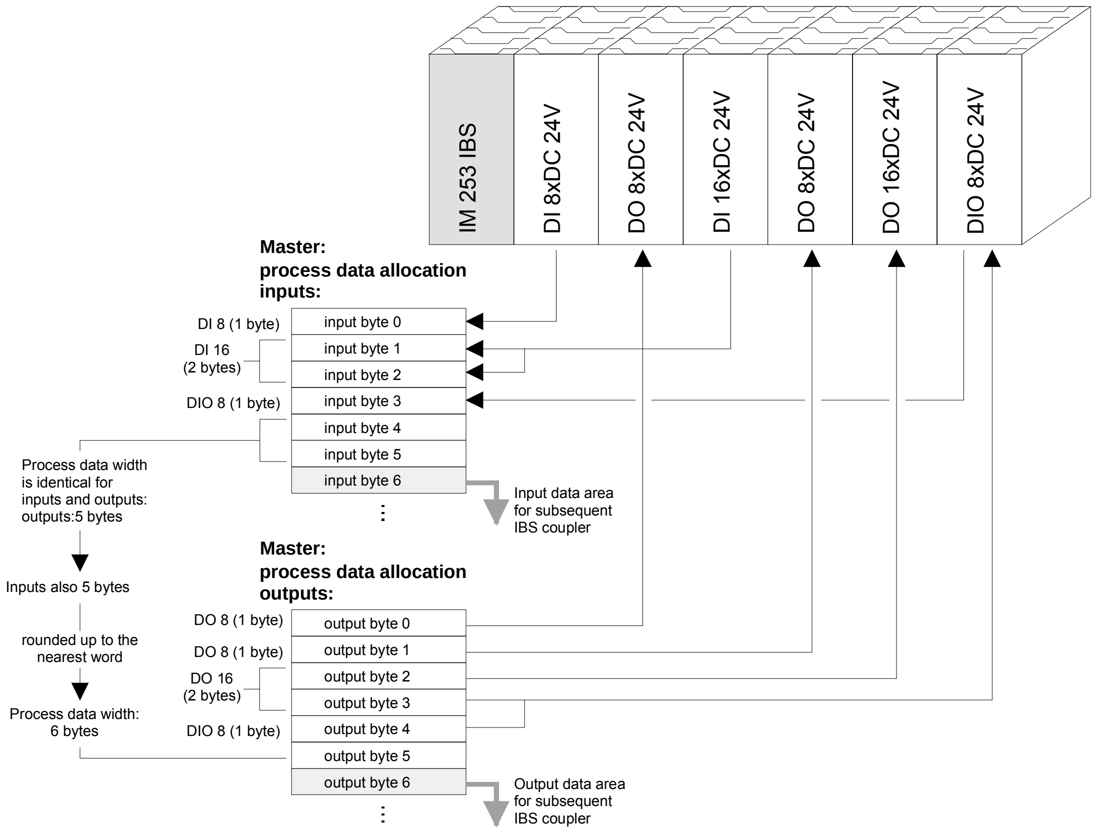
The bus coupler determines the configuration of the installed modules after power on and enters the respective data into the internal process image. This process image is sent to the master. From the process images the master generates a process data list for all couplers connected to the bus. The following two figures show the process data allocation list.

The bus coupler uses the following set of rules to generate the internal process image:

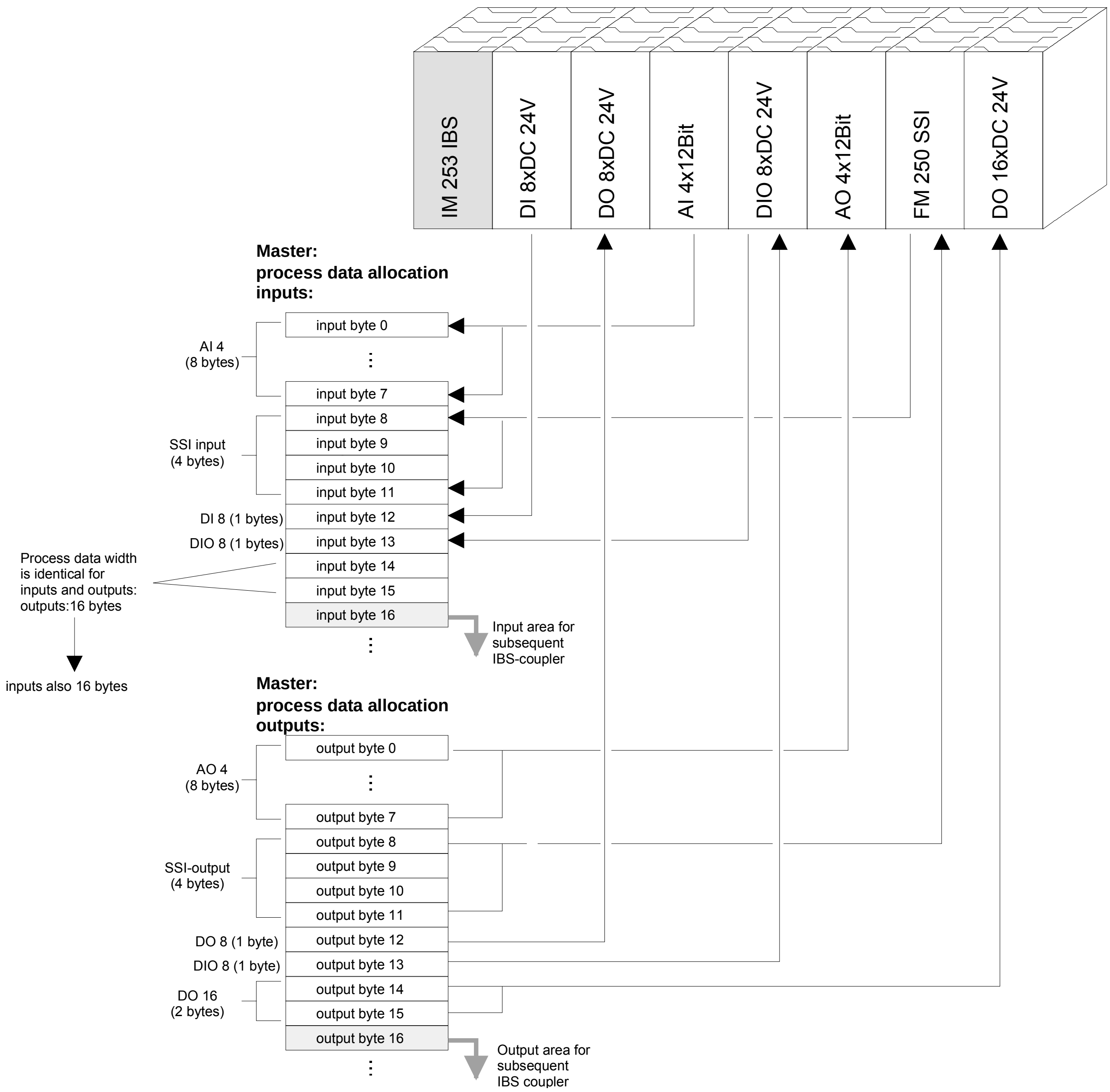
- Digital signals are bit orientated, i.e. each channel is associated with one bit in the process image.
- Separate areas exist for input and output data.
- In the input and output areas non-digital modules are always placed before digital modules.
- The sequence of these allocations depends on the plug-in location starting from the bus coupler.
- Where the data width differs between inputs and outputs the larger of the two determines the data width used by the Interbus coupler. This is always rounded up to a complete word (max. 20Byte).

The following figures are intended to show the allocation of the process data within the Interbus master.

Purely digital periphery



Combination of digital / analog periphery



Cyclic process data communications

A process image is employed to exchange input and output data. Communication with digital inputs and outputs is provided by separate data buffers which store the input and output conditions of the modules.

ID code and ID length

During the ID cycle that is executed when the Interbus system is being initialized the different modules connected to the bus identify themselves with their individual functionality and the word length. When the Interbus coupler is turned on, it determines its Interbus length during the initialization phase of the bus modules and generates the respective ID code. Depending on the configuration the Interbus coupler replies with a message identifying it as an analog or a digital remote bus device with variable word length.

Structure of the Interbus ID code

The Interbus ID code consists of 2Byte. The MSB (Byte 2) describes the length of the data words that will be transferred. Where the width of the input and output data differs, the larger value is used for the Interbus data width. The remaining 3Bit are reserved.

When the module is identified by means of the ID code, the master can only be informed of the data width by means of a word. It is for this reason that the data width is always an even number.

The LSB (Byte 1) describes the type of bus module, i.e. the type of signal and other performance criteria like remote bus, peripheral bus module, PCP, ENCOM or DRIVECOM. Bit 1 and 2 determine the direction of the data.

Byte	Bit 7 ... Bit 0
1	Bit 1 ... Bit 0: Direction of data transfer: 00: not used 01: output 10: input 11: input/output Bit 3 ... Bit 2: terminal type Bit 7 ... Bit 4: terminal class The type and class are determined by the Interbus-Club
2	Bit 4 ... Bit 0: Data width 0 to 10 words (binary) Bit 7 ... Bit 5: reserved

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 for a station is guaranteed by the Interbus data communication protocol. Synchronous scanning guarantees the consistency of the entire process image. Inconsistencies can arise due to asynchronous accesses to the data areas of the Interbus master from the control CPU. You can find information on secure access methods to the master interface in the respective manuals.

The basic data consistency is only guaranteed for 1Byte. This means that the bits belonging to a single byte were read or written as a single unit. This byte-related consistency suffices when digital signals are being processed. However, when the data length exceeds a byte, for instance for analog values, then the data consistency must be expanded. You must ensure that you transfer consistent data properly from the Interbus master into your PLC.

For further information please refer to the manual for your Interbus master.

Restrictions

You may combine a maximum of 16 input and 16 output modules with an Interbus coupler. The maximum data width for the input and output data is 10 words.

The configuration of the bus coupler or peripheral modules via the Interbus PCP protocol is not supported.

When the bus coupler is being initialized addresses are assigned to the System 200V peripheral module that are used by the bus coupler to communicate with the module under normal operating conditions. It is not possible to remove or insert any module while the system is active. This is due to the fact that addresses are only assigned after a POWER-ON or a RESET and since the data width of Interbus modules must not change while the system is operational.

In accordance with RS422 standards any remote bus segment (= distance between any two stations) may be at distances up to 400m. The maximum total extent of the system is 12.8km.

Note!

Before the change is implemented, the respective bus coupler must be powered off. Please ensure that you change the initialization in the master in accordance with the changes to the periphery!

Commissioning

Assembly and integration with Interbus

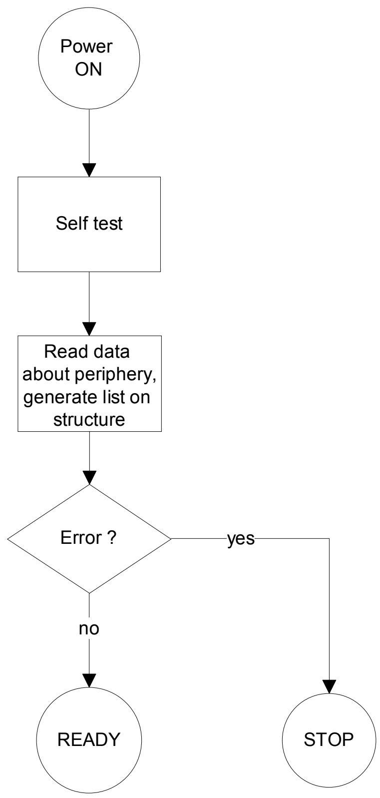
- Assemble your Interbus coupler using the required modules.
- Configure the Interbus coupler by means of the configuration tool that was supplied with the master.
- Connect the Interbus cable to the coupler and turn the power on.

Initialization phase

During the power-on self-test the bus coupler checks the functionality of its components and communications via the backplane bus. The self-test is active while the PW LED is on. When the test has been completed successfully the RC and BA LEDs are on.

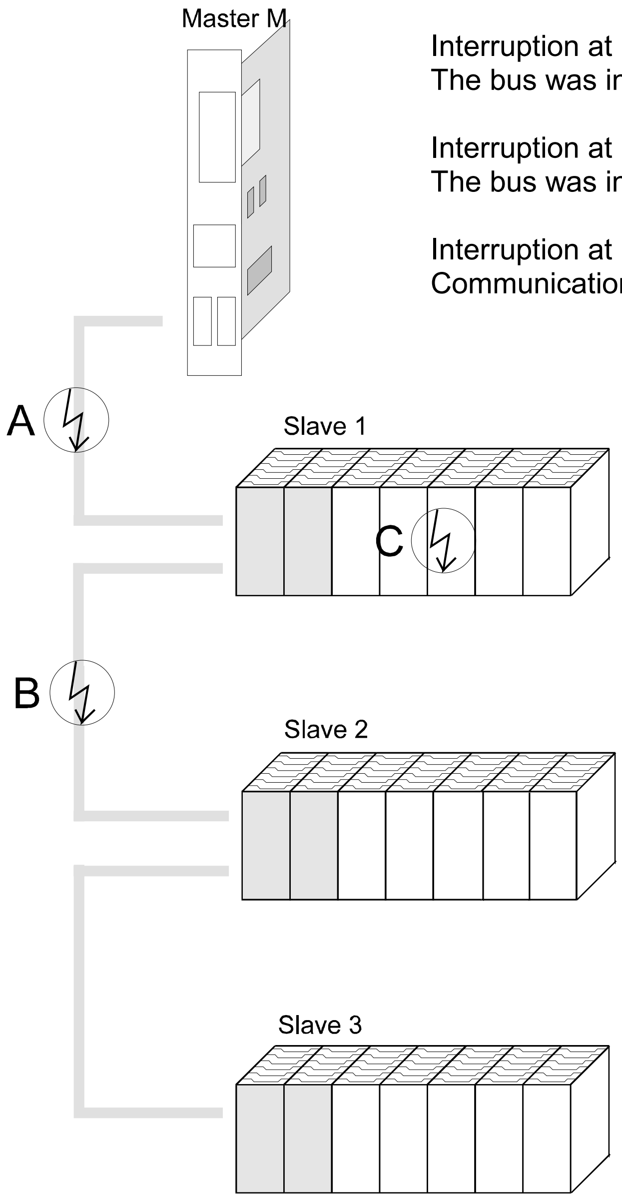
Now the peripheral structure is read in. First the number of modules connected to the bus is determined. Then the modules are identified by means of their type identifier. When the peripheral structure has been registered the location identifiers for the modules are generated. This is then transferred to the modules via the backplane bus. This procedure prepares an internal configuration list that is not externally accessible. These location identifiers provide the basis for directly addressed communications. When an error is recognized, the status of the bus coupler is set to STOP. Once the bus coupler has been initialized properly its status is set to READY.

When an error has been removed, the bus coupler can only be returned to normal operation by switching it off and on.



Diagnostic LEDs in an example

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



Interruption at position A
The bus was interrupted between the master and slave1.

Interruption at position B
The bus was interrupted between slave1 and slave2

Interruption at position C
Communications via the backplane bus was interrupted.

Slave 1	Interruption at position		
LED	A	B	C
ER	off	off	on
BA	off	off	on
RC	off	on	on
RD	on	on	off

Slave 2	Interruption at position		
LED	A	B	C
ER	off	off	off
BA	off	off	on
RC	off	off	on
RD	on	on	off

Slave 3	Interruption at position		
LED	A	B	C
ER	off	off	off
BA	off	off	on
RC	off	off	on
RD	on	on	on

Configuration of the master

As mentioned before, Interbus generates a data area containing both input and output bytes. The assignment of the modules connected to the bus coupler and the bits and bytes of the process image is provided by the bus coupler.

The Interbus master exchanges a contiguous input and output data block with every Interbus coupler. The data modules of the PLC or the configuration software allocate the bytes contained in this data block to the addresses of the process image.

Master-Software	Configuration software	Manufacturer
PLC-interfaces version <4	SYS SWT	Phoenix Contact
PLC-interfaces version <4	IBM CMD	Phoenix Contact
PC-interfaces version <3	SYS SWT	Phoenix Contact
general	SYS SWT	Phoenix Contact

Technical data

Interbus coupler
IM 253IBS

Electrical data	VIPA 253-1IB00
Power supply	DC 24V (20.4 ... 28.8) via front from ext. power supply
Current consumption (in no-load operation)	50mA
Current consumption (rated value)	800mA
Output current backplane bus	max. 3.5A
Power loss	2 W
Isolation	≥ AC 500V , according to DIN 19258
Status indicators	via LEDs located on the front
Connections / interfaces	9pin D-type (plug) inbound remote bus 9pin D-type (socket) outbound remote bus
Interbus interface	
Connection	remote bus, 9pin D-type as per DIN 19258
Network topology	Ring with an integrated return line
Medium	Screened twisted pair cable
Data transfer rate	500kBit/s
Total length	12.8km
Distance between two stations	400m
digital inputs/outputs	max.160 input bits and 160 output bits
max. no. of stations	256
Combination with peripheral modules	
max. no. of modules	16
max. digital I/O	16 (process data width 20 I / 20 O)
max. analog I/O	4 (process data width 10 I / 10 O)
	no configuration possible
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x78
Weight	80g

Chapter 5 CANopen

Overview

This chapter contains the description of the VIPA CANopen master/slave. The introduction to the system is followed by the description of the modules.

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 description of the Emergency Object and NMT as well as the technical data conclude the chapter.

Below follows a description of:

- CAN-Bus basics
- The VIPA CANopen master/slaves
- The baudrate and module-ID settings
- Deployment of the CANopen slave on the CAN-Bus with a message description
- Description of the CAN specific objects
- Technical data

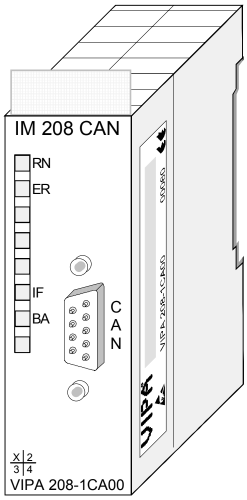
Content

Topic	Page
Chapter 5 CANopen	**5-1**
System overview	5-2
Basics	5-4
IM 208CAN - CANopen master - Structure	5-6
IM 208CAN - CANopen master - Project engineering	5-8
IM 208CAN - CANopen master - Firmware update	5-15
IM 208CAN - CANopen master - Mode	5-16
IM 208CAN - CANopen master - Process image	5-17
IM 208CAN - CANopen master - Messages	5-19
IM 208CAN - CANopen master - Object directory	5-24
IM 253CAN - CANopen slave - Structure	5-38
IM 253CAN, DO 24xDC 24V - Structure	5-43
IM 253CAN - CANopen slave - Fast introduction	5-47
IM 253CAN - CANopen slave - Baudrate and module-ID	5-51
IM 253CAN - CANopen slave - Message structure	5-52
IM 253CAN - CANopen slave - PDO	5-54
IM 253CAN - CANopen slave - SDO	5-58
IM 253CAN - CANopen slave - Object directory	5-60
IM 253CAN - CANopen slave - Emergency Object	5-101
IM 253CAN - CANopen slave - NMT - network management	5-102
Technical data	5-104

System overview

CANopen-Master IM 208CAN

The following CANopen master is available from VIPA:

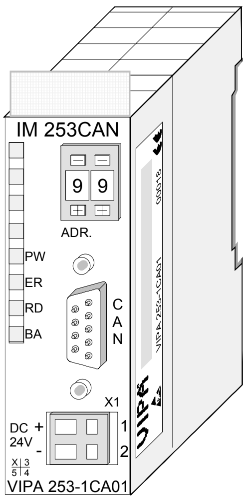


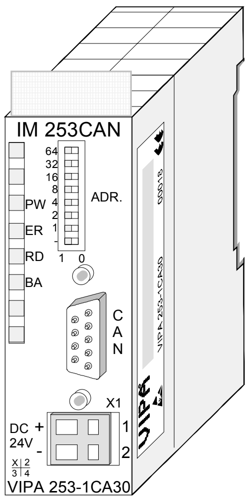
Order data

Type	Order number	Description	Page
IM 208CAN	VIPA 208-1CA00	CAN-Bus CANopen master 1MBaud, up to 125 slaves	5-6

CANopen slave IM 253CAN

Currently the following CANopen bus couplers are available from VIPA:

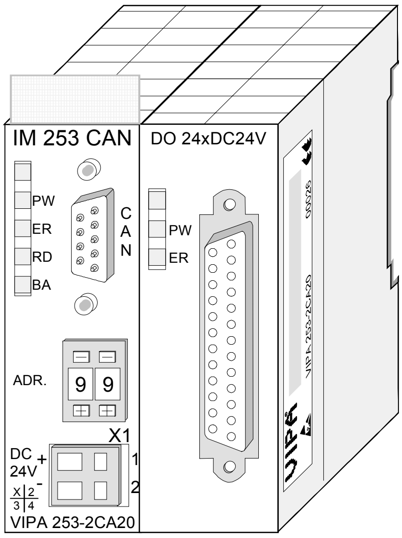




Order data

Type	Order number	Description	Page
IM 253CAN	VIPA 253-1CA01	CAN-Bus CANopen slave	5-38
IM 253CAN	VIPA 253-1CA30	CAN-Bus CANopen slave - ECO	5-38

CANopen slave (Combi modules)



Order data

Type	Order number	Description	Page
IM 253CAN	VIPA 253-1CA20	CAN-Bus CANopen slave with DO 24xDC 24V	5-43

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

Communication 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 baudrate.

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.

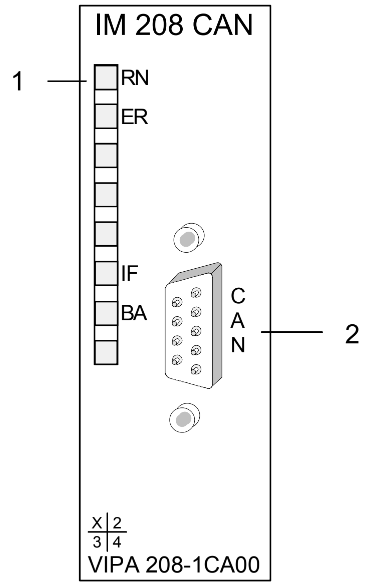
CAN-Bus access control is performed by means of a collision-free, bit-based arbitration algorithm. Collision-free means that the final winner of the arbitration process does not have to repeat his message. The station with the highest priority is selected automatically when more than one station accesses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

IM 208CAN - CANopen master - Structure

Properties

- 125 CAN slaves can be connected to one CANopen master
- Project engineering under WinCoCT from VIPA
- Diagnosis ability
- 40 Transmit PDOs
- 40 Receive PDOs
- PDO-Linking
- PDO-Mapping
- 1 SDO as Server, 127 SDO as Client
- Emergency Object
- NMT Object
- Node Guarding, Heartbeat
- In-/output range 0x6xxx each max. 64Bytes
- In-/output range 0xAxxx each max. 320Bytes

Structure IM 208CAN



[1] LED status indicators
[2] CAN-Bus socket

Components

LEDs

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

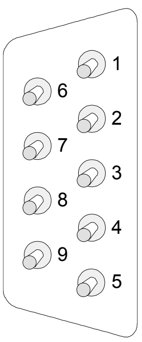
Name	Color	Description
RN	green	ON: CPU is in RUN
		OFF: CPU is in STOP
ER	red	ON: During initialization and at slave failure
		OFF: All slaves are in the state "operational"
BA	yellow	BA (**B**us **a**ctive) shows communication via CAN bus.
		ON: state "operational"
		Blinks with 1Hz: shows state "pre-operational".
IF	red	ON: „**I**nitialisierungs**f**ehler“ (i.e. initialization error) at wrong parameterization.
		OFF: Initialization is OK.

Note!
If all LEDs are blinking with 1Hz, the CAN master awaits valid parameters from the CPU. If the CAN master is not supplied with parameters by the CPU his LEDs get off after 5s.

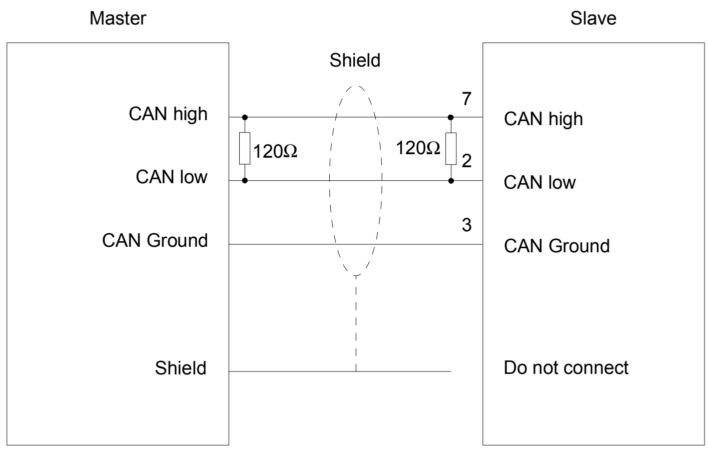
CAN-interface

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

The following diagram shows the pin assignment for the interface.:



Pin	Assignment
1	reserved
2	CAN low
3	CAN Ground
4	reserved
5	Screen
6	Ground 24V
7	CAN high
8	reserved
9	+DC 24V



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

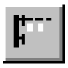
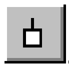
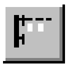
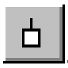
Power supply The CANopen master receives the voltage supply via the backplane bus.

IM 208CAN - CANopen master - Project engineering

The project engineering of the CANopen master happens in WinCoCT (**Win**dows **C**AN**o**pen **C**onfiguration **T**ool) from VIPA. You export your project from WinCoCT as wld-file. This wld-file can then be imported into the hardware configurator from Siemens.

Fast introduction

For the deployment of System 200V modules and the CAN master, you have to include the System 200V modules into the hardware catalog via the GSD-file from VIPA. For the project engineering in the hardware configurator you have to execute the following steps:

- Start WinCoCT and project the CANopen network.
- Create a master group with  and insert a CANopen master via .
- Activate the master function via "Device Access" and "Device is NMT Master".
- Activate in the register "CANopen Manager" Device is NMT Master and confirm your entry.
- Set parameters like diagnosis behavior and CPU address ranges with "Set PLC Parameters".
- Create a slave group with  and add your CANopen slaves via .
- Add modules to your slaves via "Modules" and parameterize them if needed.
- Set your process data connections in the matrix via "Connections" and proof your entries if needed in the process image of the master.
- Save the project and export it as wld-file.
- Switch to the SIMATIC manager from Siemens and copy the data block from the CAN-wld-file into the block directory.
- Project the Profibus-DP master system in the hardware configurator with the following Siemens-CPU: CPU 315-2DP (6ES7 315-2AF03-0AB0)
- The DP master receives an address >1.
- Add the System 200V DP slave system from the hardware catalog to the master system.
- The System 200V DP slave system always requires the address 1.
- Save all and transfer the PLC project together with the wld-file via MPI into the CPU.

In the following you'll find a description of this steps.

Note!
Starting with the firmware version 3.5.0, please use the hardware catalog CPU **6ES7-315-2AF03** V1.2 from Siemens for the project engineering of the VIPA standard CPUs of the Systems 100V, 200V, 300V and 500V!

Precondition for the project engineering

The hardware configurator is a part of the Siemens SIMATIC Manager. It serves the project engineering. The modules that can be parameterized with are monitored in the hardware catalog.

For the deployment of the System 200V modules, the inclusion of the System 200V modules into the hardware catalog is necessary. This happens via a GSD-file from VIPA.

Note!

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

Include GSD-file

- Copy the delivered VIPA GSD-file VIPA_21x.gsd into your GSD-directory... \siemens\step7\s7data\gsd
- Start the hardware configurator from Siemens
- Close all projects
- Choose **Options** > *Install new GSD-file*
- Select **VIPA_21x.GSD**

Now the modules of the System 200V from VIPA are integrated in the hardware catalog and can be projected.

Note

To be compatible to the Siemens SIMATIC Manager, the System 200V CPUs from VIPA have to be projected as

CPU 315-2DP (6ES7 315-2AF03-0AB0)!

To be able to directly address the modules, you have to include them in the hardware configurator from Siemens in form of a virtual Profibus system. By including the GSD-file from VIPA, you are able to access the complete function range of the modules.

The concrete project engineering happens in the CANopen configuration tool WinCoCT. You may export your project as wld-file and transfer it as DB into your PLC program.

WinCoCT

WinCoCT (**Win**dows **CANo**pen **C**onfiguration **T**ool) is a configuration tool developed from VIPA to allow the comfortable project engineering of CANopen networks.

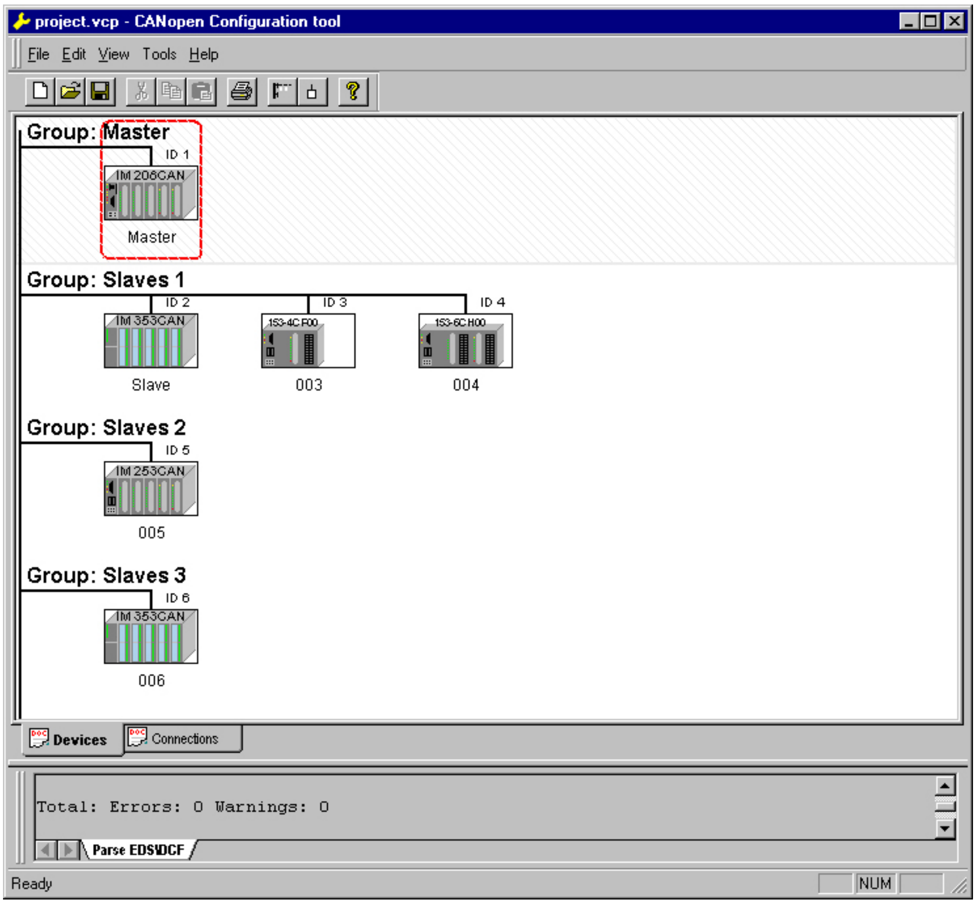
WinCoCT monitors the CANopen network topology in a graphical user interface. Here you may place, parameterize and group field devices and controls and engineer connections.

The selection of the devices happens via a list that can be extended for your needs with an EDS-file (**E**lectronic **D**ata **S**heet) at any time.

A right click onto a device opens a context menu consisting partly of static and partly of dynamic components.

For the configuration of the process data exchange, all process data are monitored in a matrix with the device inputs as rows and the device outputs as columns. Mark a cross point to create the wanted connection.

The telegram collection and optimization is executed by WinCoCT.



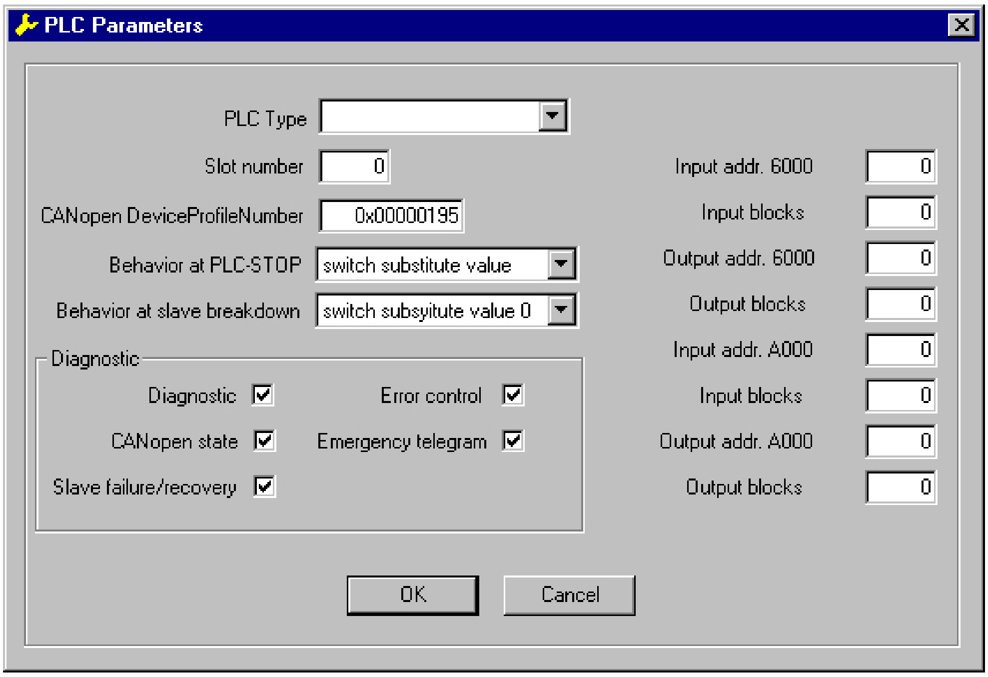
Set project parameters

Via **Tools** > *Project options* you may preset CAN specific parameters like baud rate, selection of the master etc.

More detailed information is to find in the WinCoCT manual.

Parameter CAN master

WinCoCT allows you to preset VIPA specific parameters for the CAN master by doing a right click onto the master and call the following dialog window with Set PLC-Parameters:



PLC Type

Reserved for later extensions

Slot number

Plug-in location no. at the bus

0: For the addressing of the CAN master integrated in the CPU

1 ... 32: For the addressing of CAN master at the standard bus

CANopen DeviceProfileNumber

Fix at 0x195

Behavior at PLC-STOP

Here you can define the reaction of the output channels if the CPU switches to STOP. The following values are available:

Switch substitute value 0: Sets all outputs to 0

Keep last value: Keeps the recent state of the outputs.

Behavior at Slave breakdown

Here you set the reaction for the slave input data in case of a slave failure.

Switch substitute value 0: The data is set to 0.

Keep the last value: The recent date remain unchanged.

Diagnostic

This area allows you to define the diagnostic reaction of the CAN master.

Diagnostic: Activates the diagnostic function

CANopen state: When activated, the CAN master sends its state "preoperational" or "operational" to the CPU. You may request the state via SFC 13.

Slave failure/recovery: When activated, the OB 86 is called in the CPU in case of slave failure and reboot.

Error control: If this option is selected, the NMT master sends all Guarding errors as diagnosis to the CPU, that calls the OB 82.

Emergency Telegram: At activation, the NMT master sends all Emergency telegrams as diagnosis to the CPU, that calls the OB 82.

Address range in the CPU

The following fields allow you to preset the address ranges in the CPU for the CANopen master in- and output ranges. Each block consists of 4Byte.

Input addr. 6000, Input blocks

PI basic address in the CPU that are occupied from 0x6000 CAN input data. For input blocks max. 16 (64Byte) can be entered.

Output addr. 6000, Output blocks

PO basic address in the CPU that are occupied from 0x6000 CAN output data. For output blocks max. 16 (64Byte) can be entered.

Input addr. A000, Input blocks

PI basic address in the CPU that are occupied from 0xA000 CAN input network variables. For input blocks max. 80 (320Byte) can be entered.

Output addr. A000, Output blocks

PO basic address in the CPU that are occupied from 0xA000 CAN output network variables. For output blocks max. 80 (320Byte) can be entered.

Activate CANopen slave in the CANopen Manager

To enable the master to access a CANopen slave, you have to register it at the according master via WinCoCT. Right click onto your CAN master, choose "Device access" and switch to the register "CANopen Manager".

Via [Change] you can register every single slave res. via [Global] all slaves at your master and preset the error behavior.

Please don’t forget to apply the settings into your project engineering by clicking on [Apply to slaves].

Steps of the project engineering

The following text describes the approach of the project engineering with an abstract sample:

The project engineering is divided into three parts:

- CAN master project engineering in WinCoCT and export as wld-file
- Import CAN master project engineering
- Project engineering of the modules

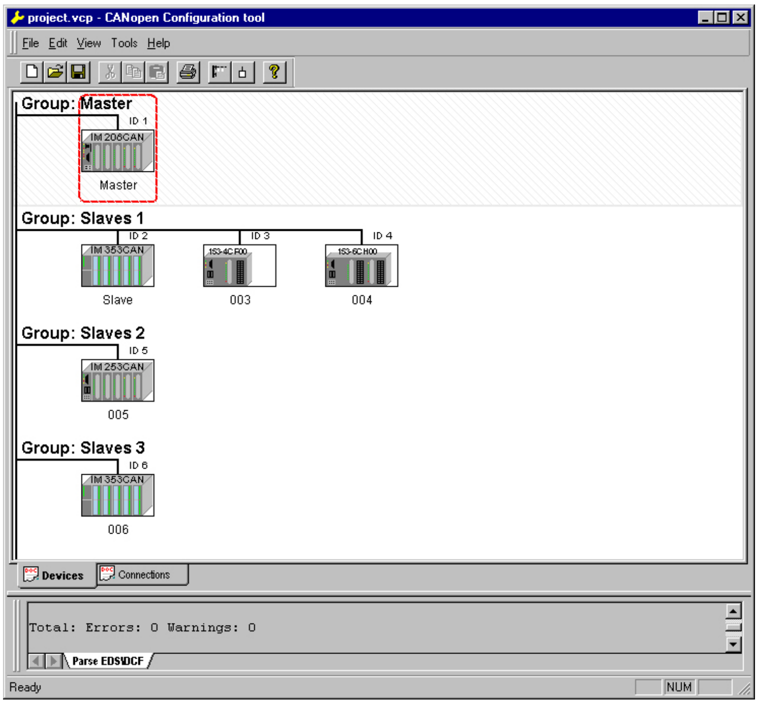
Preconditions

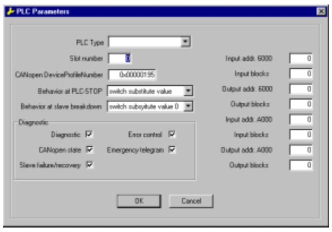
For the project engineering of a CANopen system, the most recent EDS-file has to be transferred into the EDS-directory of WinCoCT.

For the deployment of the System 200V modules, you have to include the System 200V modules with the GSD-file VIPA_21x.gsd from VIPA into the hardware catalog.

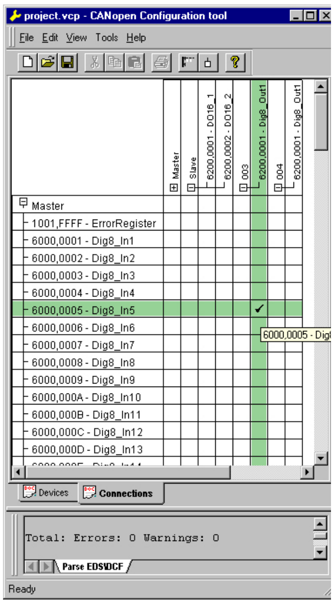
CAN master project engineering in WinCoCT

- Copy the required EDS-files into the EDS-directory and start WinCoCT.
- Create a master group via and insert a CANopen master via (VIPA_208_1CA00.eds).
- Create a slave group with and add your CANopen slaves via .
- Right click on the according slave and add the needed modules via „Modules“.
- Parameterize the modules with [Parameter] res. via the according object directory.
- Right click on the master and open the dialog "Device Access".
- Activate Device is NMT Master in the register "CANopen Manager" and register the according slaves at the master. Don’t forget to apply your settings into your project engineering with [Apply to slaves]!





- Right click onto the master and open the VIPA specific dialog "Set PLC Parameters". Here you may adjust the diagnosis behavior and the address ranges that the master occupies in the CPU.
  Under "Slot number" type the slot no., where your CAN master is plugged. At export, WinCoCT creates the according DB no. + 2000.



- Change to the register "Connections" in the main window. Here the process data are shown in a matrix as inputs (1st column) and as outputs (1st row).
  To monitor the process data of a device with a "+" click on the according device.
- For helping you, you may only define a connection when the appearing cross has green color. Select the according cell with the mouse pointer in row and column in the matrix and click on it. → The cell is marked with a "🗑". You can control the connection by changing into "Devices", click on the master and monitor the process image of the master via "Device Access".
- Save your project.
- Via **File** > *Export* your CANopen project is exported into a wld-file. The name is the combination of project name + node address + ID **Mas**ter/**Sl**ave.

Now your CANopen project engineering under WinCoCT is ready.

Import into PLC program and transfer to CAN master

- Start the SIMATIC manager from Siemens with your PLC project and open the wld-file via **File** > *Memory Card File* > *open*.
- Copy the DB 2xxx into your block directory.
- Start the hardware configurator from Siemens with a new project and insert a profile rail from the hardware catalog.
- Place the following Siemens CPU onto plug-in location 2:
  CPU 315-2DP (6ES7 315-2AF03-0AB0). For the project engineering of the VIPA standard CPUs of the Systems 100V, 200V, 300V and 500V please use starting with the firmware version 3.5.0 the CPU **6ES7-315-2AF03** V1.2 from Siemens from the hardware catalog!
- If for example your CAN master module is directly placed beside the CPU, you project your CAN master on plug-in location 4.
- Starting with plug-in location 5, you include your System 200V modules on the standard bus in the plugged sequence.
- Parameterize your CPU res. the modules when needed. The parameter window is opened when you double click on the according module.
- Save your project and transfer it to your CPU.

After the transfer the CPU recognizes the DB for the CAN master and passes the contents of the DB on to the according CAN master at STOP-RUN change.

IM 208CAN - CANopen master - Firmware update

Overview

Starting with CPU firmware version 3.4.8 a MMC inside your CPU can be used to update the firmware of CPU an CAN master. The latest 2 firmware versions are to find in the service area at www.vipa.de and at the ftp server under ftp.vipa.de. For more details see manual HB97-CPU.

For designation the master firmware has the following name convention:

canxx.**bin** xx specifies the slot number the CAN master is plugged in (Slot: 01 ... 32)

Attention!
When installing a new firmware you have to be extremely careful. Under certain circumstances you may destroy the CAN-Master, for example if the voltage supply is interrupted during transfer or if the firmware file is defective.

In this case, please call the VIPA-Hotline!

Seek firmware version

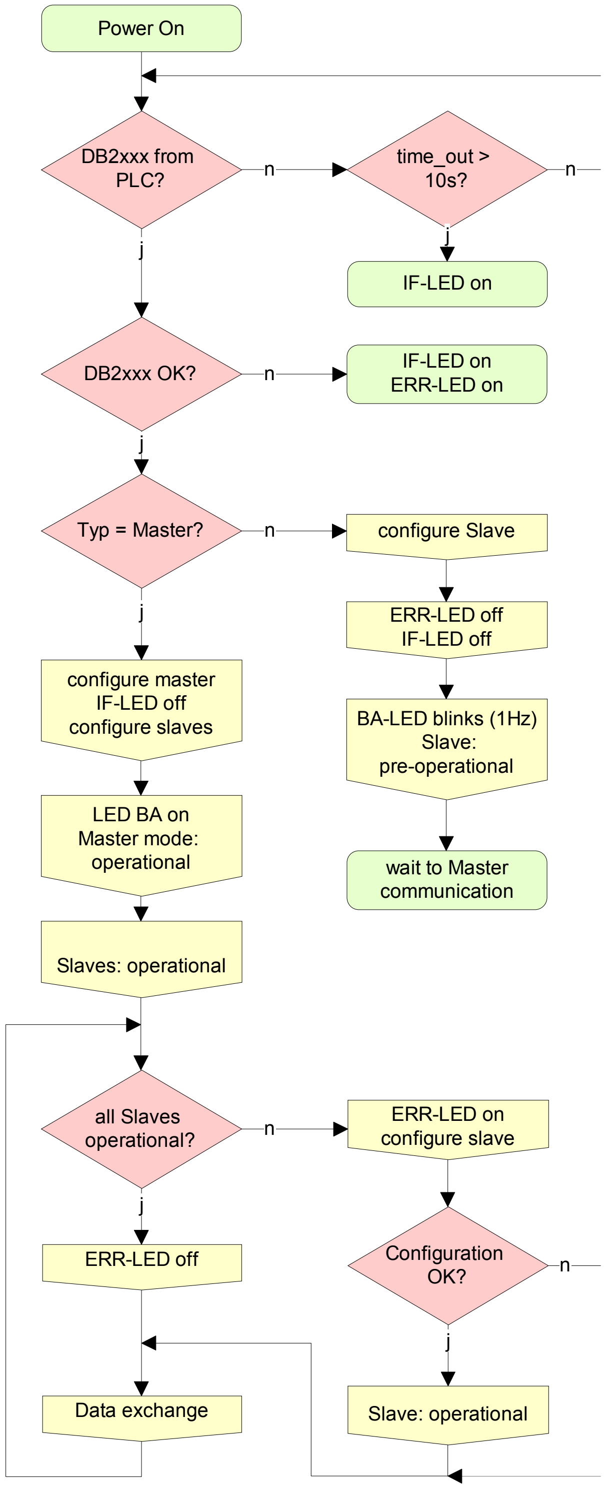
A label on the rear of the module indicates the firmware version.

Load firmware and transfer it to MMC as canxx.*bin*

- Go to www.vipa.de.
- Click on Service > Download > Firmware Updates.
- Click on "Firmware for CAN Master System 200V".
- Select the according IM 208 order no. and download the firmware to your PC.
- Rename the file to "canxx.**bin**" (xx specifies the slot number the DP master is plugged in, starting with 01) and transfer this file onto a MMC.

Note!
The server always stores the latest two firmware versions.

IM 208CAN - CANopen master - Mode



STOP → RUN (automatically)

After POWER ON and at valid project data in the CPU, the master switches automatically into RUN. The master has no operating mode lever.

After POWER ON, the project data is automatically send from the CPU to the CAN master. This establishes a communication to the CAN slaves.

At active communication and valid bus parameters, the CAN master switches into the state "operational". The LEDs RUN and BA are on.

At invalid parameters, the CAN master remains in STOP and shows the parameterization error via the IF-LED.

RUN

In RUN, the RUN- and BA-LEDs are on. Now data can be exchanged.

In case of an error, like e.g. slave failure, the ERR-LED at the CAN master is on and an alarm is send to the CPU.

IM 208CAN - CANopen master - Process image

The process image is build of the following parts:

- Process image for input data (PI) for RPDOs
- Process image for output data (PO) for TPDOs

Every part consists of 64Byte "Digital-Data"- and 320Byte "Network Variables".

Input data

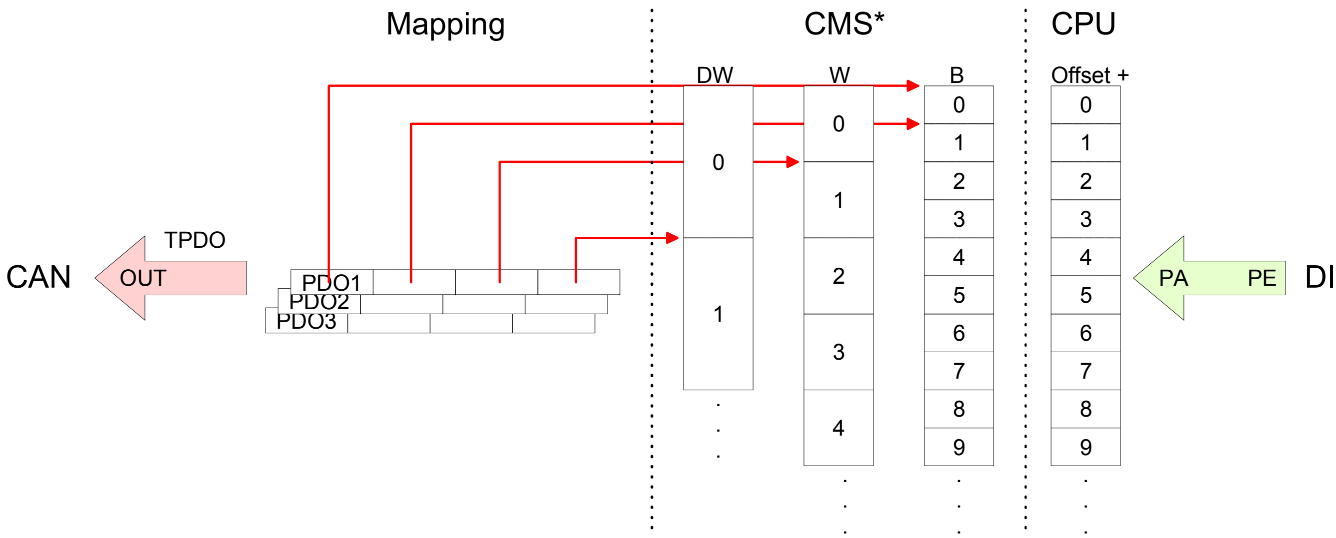
For input data, the following objects are available:

- 8 Bit digital input (Object 0x6000)
- 16 Bit digital input (Object 0x6100)
- 32 Bit digital input (Object 0x6120)
- 8 Bit input network variables (Object 0xA040)
- 16 Bit input network variables (Object 0xA100)
- 32 Bit input network variables (Object 0xA200)
- 64 Bit input network variables (Object 0xA440)

Like to see in the following illustration, the objects of the digital input data use the same memory area of the CPU.

For example, an access to Index 0x6000 with Subindex 2 corresponds an access to Index 0x6100 with Subindex 1. Both objects occupy the same memory cell in the CPU.

Please regard that the input network variables also use the same memory area.



*) CMS = **C**ANopen **m**aster/**s**lave

Output-data

For the digital output data, the assignment is similar.

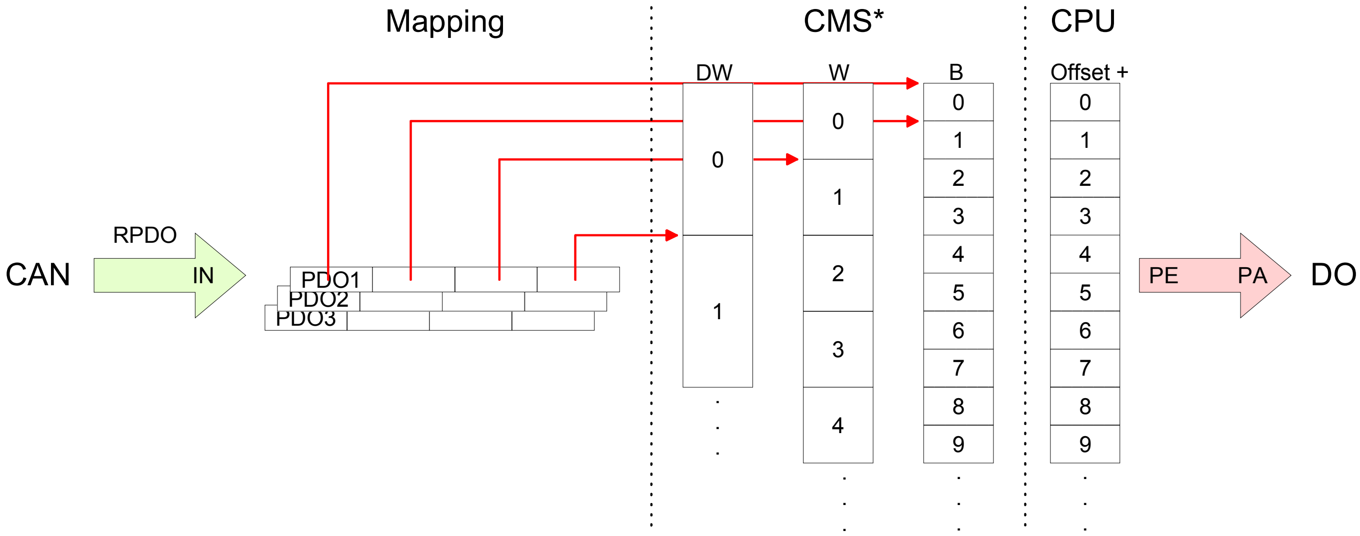
For output data, the following objects are available:

- 8 Bit digital output (Object 0x6200)
- 16 Bit digital output (Object 0x6300)
- 32 Bit digital output (Object 0x6320)
- 8 Bit output network variables (Object 0xA400)
- 16 Bit output network variables (Object 0xA580)
- 32 Bit output network variables (Object 0xA680)
- 64 Bit output network variables (Object 0xA8C0)

Like to see in the following illustration, the objects of the digital output data use the same memory area of the CPU.

For example, an access to Index 0x6200 with Subindex 2 corresponds an access to Index 0x6300 with Subindex 1. Both objects occupy the same memory cell in the CPU.

Please regard that the output network variables also use the same memory area.



*) CMS = **C**ANopen **m**aster/**s**lave

IM 208CAN - CANopen master - Messages

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 master supports the following objects:

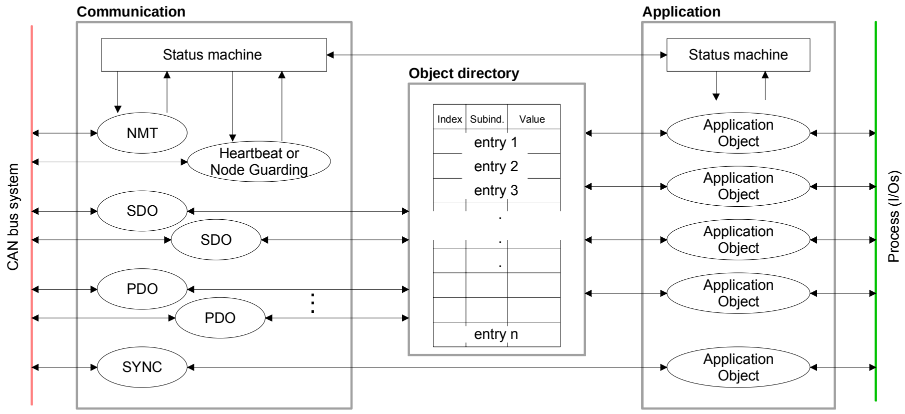
- 40 Transmit PDOs (PDO Linking, PDO Mapping)
- 40 Receive PDOs (PDO Linking, PDO Mapping)
- 2 Standard SDOs (1 Server, 127 Clients)
- 1 Emergency Object
- 1 Network management Object NMT
- Node Guarding
- Heartbeat

Note!

The exact structure and data content of all objects is described in the CIA-Profiles DS-301, DS-302, DS-401 and DS-405.

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 addres-sed 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.

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-Master supports 80 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 40 Tx transmit PDOs for input data and 40 Rx receive PDOs for output data. The PDOs are named seen from the CAN-Master:

Receive PDOs (RxPDOs) are received by the CAN-Master and contain input data.

Transmit PDOs (TxPDOs) are send by the CAN-Master and contain output data.

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

SDO

For access to the object directory, the **S**ervice-**D**ata-**O**bject (SDO) is used. The SDO allows you a read or write access to the object directory. In the CAL-Layer-7-Protocol you find the specification of the Multiplexed-Domain-Transfer-Protocol that is used by the SDOs. This protocol allows you to transfer data with any length. At need, the messages are divided into several CAN messages with identical identifier (segmentation). A SDO is transferred acknowledged, i.e. every reception of a message is acknowledged.

Note!

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

In the following only the error messages are described that may occur at a wrong parameter communication.

SFC 219 CAN_TLGR SDO request to CAN master

Every CPU has the SFC 219 integrated. This allows you to start a SDO read or write access from your PLC program to the CAN master.

You address your master via the plug-in location and the destination slave via its CAN address. The process data is defined by index and subindex. Via SDO every access transfers max. one data word process data. The SFC 219 contains the following parameters:

Parameter	Declaration	Type	Description
REQUEST	IN	BOOL	
SLOT_MASTER	IN	BYTE	
NODEID	IN	BYTE	
TRANSFERTYP	IN	BYTE	
INDEX	IN	DWORD	
SUBINDEX	IN	DWORD	
CANOPENERROR	OUT	DWORD	
RETVAL	OUT	WORD	
BUSY	OUT	BOOL	
DATABUFFER	IN_OUT	ANY	

REQUEST

Control parameter: 1: Start the order

SLOT_MASTER

0: VIPA 21x-2CM01

1...32: VIPA 208-1CA00, depending on plug-in location no.

NODELD

Address of the CANopen node (1...127)

TRANSFER TYPE

40h, 60h: Read SDO

61h: Write SDO (undefined length)
23h: Write SDO (1 DWORD)
2Bh: Write SDO (1 WORD)
2Fh: Write SDO (1 BYTE)

INDEX

CANopen Index

SUBINDEX

CANopen Subindex

CANOPENERROR If no error occurs *CANOPENERROR* returns value 0.

In case of error the *CANOPENERROR* contains one of the following error messages which are generated in the CAN master:

Code	Description
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)

RETVAL

When the function has been executed successfully, the return value contains the valid length of the respond data: 1: BYTE, 2: WORD, 4: DWORD.

If an error occurs during function processing, the return value contains an error code.

Value	Description
F021h	Invalid slave address (Call parameter equal 0 or above 127)
F022h	Invalid Transfer type (Value unequal 60h, 61h)
F023h	Invalid data length (data buffer to small, at SDO read access it should be at least 4Byte, at SDO write access 1Byte, 2Byte or 4Byte).
F024h	The SFC is not supported
F025h	Write buffer in the CANopen master full, service can not be processed at this time.
F026h	Read buffer in the CANopen master full, service can not be processed at this time.
F027h	The SDO read or write access returned wrong answer, see CANopen Error Codes.
F028h	SDO-Timeout (no CANopen participant with this Node-Id has been found).

BUSY

Busy = 1: The read/write job is not yet completed.

DATABUFFER

SFC data communication area.

Read SDO: Destination area for the SDO data that were read.

Write SDO: Source area for the SDO data that were write.

Note

Unless a SDO demand was processed error free, *RETVAL* contains the length of the valid response data in 1, 2 or 4 byte and the *CANOPENERROR* the value 0.

IM 208CAN - CANopen master - 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 – 0x1011	General communication specific parameters (e.g. device name)
0x1400 – 0x1427	Communication parameters (e.g. identifier) of the receive PDOs
0x1600 – 0x1627	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 – 0x1827 0x1A00 – 0x1A27	Communication and mapping parameters of the transmit PDOs

Manufacturer specific profile area (0x2000 – 0x5FFF)

Here you find the manufacturer specific entries. The CAN master from VIPA has no manufacturer specific entries.

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
1000h	Device type
1001h	Error register
1005h	COB-ID SYNC
1006h	Communication Cycle Period
1007h	Synchronous Window Length
1008h	Manufacturer Hardware Version
1009h	Hardware Version
100Ah	Software Version
100Ch	Guard Time
100Dh	Life Time Factor
1016h	Consumer Heartbeat Time
1017h	Producer Heartbeat Time
1018h	Identity Object
1400h to 1427h	Receive PDO Communication Parameter
1600h to 1627h	Receive PDO Mapping Parameter
1800h to 1827h	Transmit PDO Communication Parameter
1A00h to 1A27h	Transmit PDO Mapping Parameter
1F22h	Concise DCF
1F25h	Post Configuration
1F80h	NMT StartUp
1F81h	Slave Assignment
1F82h	Request NMT
1F83h	Request Guarding
6000h	Digital-Input-8-Bit Array (see DS 401)
6100h	Digital-Input-16-Bit Array (see DS 401)
6120h	Digital-Input-32Bit Array (see DS 401)
6200h	Digital-Output-8-Bit Array (see DS 401)
6300h	Digital-Output-16-Bit Array (see DS 401)
6320h	Digital-Output-32-Bit Array (see DS 401)
A040h	Dynamic Unsigned8 Input
A100h	Dynamic Unsigned16 Input
A200h	Dynamic Unsigned32 Input
A4400h	Dynamic Unsigned64 Input
A4C0h	Dynamic Unsigned8 Output
A580h	Dynamic Unsigned16 Output
A680h	Dynamic Unsigned32 Output
A8C0h	Dynamic Unsigned64 Output

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)	405dec=0x0195

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

Bit 7							Bit 0
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)

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	Communi-cation cycle period	Unsigned32	rw	N	0x00000000	Maximum length of the SYNC interval in µs.

If a value other than zero is entered here, the master 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 Master / Slave 208-1CA00

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	1.00	Hardware version number of bus coupler

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	1.xx	Software version number CANopen software

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

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, but can itself be monitored by the master (node guarding).

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...127		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	0x2081CA00	Product Code
	3	Revision Number	Unsigned32	ro	N		Revision Number
	4	Serial Number	Unsigned32	ro	N		Serial Number

Communication parameter RxPDO

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1400 ... 0x1427	0	Number of Elements	Unsigned8	ro	N	0x02	Communication parameter for the first receive PDOs, Subindex 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 (0) or not (1), Bit 30 shows if a RTR access to this PDO is permitted (0) or not (1).

The sub-index 2 contains the transmission type.

Mapping RxPDO

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1600 ... 0x1627	0	Number of Elements	Unsigned8	rw	N	0x01	Mapping parameter of the first receive PDO; subindex 0: number of mapped objects
	1	1st mapped object	Unsigned32	rw	N	0x62000108	(2 byte index, 1 byte subindex, 1 byte bit-width)
	2	2nd mapped object	Unsigned32	rw	N	0x62000208	(2 byte index, 1 byte subindex, 1 byte bit-width)
	...	...	...	...	...	...	...
	8	8th mapped	Unsigned32	rw	N	0x62000808	(2 byte index, 1 byte subindex, 1 byte bit-width)

The reception PDOs get a default mapping automatically from the master depending on the connected modules.

Communication parameter TxPDO1

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1800 ... 0x1827	0	Number of Elements	Unsigned8	ro	N	0x05	Communication parameter of the first transmit PDO, subindex 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 (0) or not (1), 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.

Mapping TxPDO1

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1A00 ... 0x1A27	0	Number of Elements	Unsigned8	rw	N	depending on the components fitted	Mapping parameter of the first transmit PDO; subindex 0: number of mapped objects
	1	1st mapped object	Unsigned32	rw	N	0x60000108	(2 byte index, 1 byte subindex, 1 byte bit-width)
	2	2nd mapped object	Unsigned32	rw	N	0x60000208	(2 byte index, 1 byte subindex, 1 byte bit-width)
	...	...	...	...	...	...	...
	8	8th mapped object	Unsigned32	rw	N	0x60000808	(2 byte index, 1 byte subindex, 1 byte bit-width)

The send PDOs get a default mapping automatically from the coupler depending on the connected modules.

Concise DCF

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1F22	Array	Concise DCF	Domain	rw	N		

This object is required for the Configuration Manager. The Concise-DCF is the short form of the DCF (**D**evice **C**onfiguration **F**ile).

Post Configuration

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1F25	Array	ConfigureSlave	Unsigned32	rw	N	0x00000000	

Via this entry, the Configuration Manager can be forced to transfer a stored configuration into the net.

The configuration can be initiated for a defined node at any time via the index 0x1F25.

Subindex 0 has the value 128.

Subindex x (with x = 1..127): Starts the reconfiguration for nodes with the node ID x.

Subindex 128: reconfiguration of all nodes.

For example: If you want to initiate the configuration for node 2 and there are configuration data for this node available, you have to write the value 0x666E6F63 (ASCII = "conf") to the object 1F25h Subindex 2.

NMT Start-up

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1F80	0x00	NMTStartup	Unsigned32	rw	N	0x00000000	

Define the device as NMT master.

Bit	Meaning
Bit 0	0 : Device is NOT the NMT Master. All other bits have to be ignored. The objects of the Network List have to be ignored. 1 : Device is the NMT Master.
Bit 1	0 : Start only explicitly assigned slaves. 1 : After boot-up perform the service NMT Start Remote Node All Nodes
Bit 2..31	Reserved by CiA, always 0

Slave Assignment

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1F81	0x00	SlaveAssignment	Unsigned32	rw	N	0x00000000	

Enter the nodes that are controlled by the master. For every assigned node you need one entry.

Subindex 0 has the value 127. Every other Subindex corresponds with the Node-ID of the node.

Byte	Bit	Description
Byte 0	Bit 0	0: Node with this ID is not a slave 1: Node with this ID is a slave. After configuration (with Configuration Manager) the Node will be set to state Operational.
	Bit 1	0: On Error Control Event or other detection of a booting slave inform the application. 1: On Error Control Event or other detection of a booting slave inform the application and automatically start Error Control service.
	Bit 2	0: On Error Control Event or other detection of a booting slave do NOT automatically configure and start the slave. 1: On Error Control Event or other detection of a booting slave do start the process Start Boot Slave.
	Bit 3..7	Reserved by CiA, always 0
Byte 1		8 Bit Value for the RetryFactor
Byte 2,3		16 Bit Value for the GuardTime

Request NMT

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x1F82	0x00	RequestNMT	Unsigned32	rw	N	0x00000000	

If a totally automatic start of the stack is not wanted, the functionalities:

- Status change
- Start of the guarding
- Configuration via CMT

can be also executed at request for every node. The request always happens via objects in the object directory.

The switch of the communication state of all nodes in the network (including the local slaves) happens via the entry 1F82h in the local object directory:

Subindex 0 has the value 128.

Subindex x (with x=1..127): Initiates the NMT service for nodes with Node ID x.

Subindex 128: Initiates NMT service for all nodes.

At write access, the wanted state is given as value.

State	Value
Prepared	4
Operational	5
ResetNode	6
ResetCommunication	7
PreOperational	127

Request Guarding

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1F83	0x00	RequestGuarding	Unsigned32	rw	N	0x00000000	

Subindex 0 has the value 128.

Subindex x (with x=1..127): Initiates guarding for the slave with Node ID x.

Value	Write Access	Read Access
1	Start Guarding	Slave actually is guarded
0	Stop Guarding	Slave actually is not guarded

Subindex 128: Request Start/Stop Guarding for all nodes.

8bit Digital inputs

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
	...	...	...	...	...	...	...
	0x40	64th input block	Unsigned8	ro	Y		64th digital input block

16bit Digital inputs

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
	...	...	...	...	...	...	...
	0x20	32nd input block	Unsigned16	ro	N		32nd digital input block

32bit Digital inputs

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
	...	...	...	...	...	...	...
	0x10	16th input block	Unsigned32	ro	N		16th digital input block

8bit Digital outputs

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
	...	...	...	...	...	...	...
	0x40	64th output block	Unsigned8	rw	Y		64th digital output block

16bit Digital outputs

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
	...	...	...	...	...	...	...
	0x20	32nd output block	Unsigned16	rw	N		32nd digital output block

32bit Digital outputs

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
	...	...	...	...	...	...	...
	0x10	16th output block	Unsigned32	rw	N		16th digital output block

8bit Network input variables

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0xA040	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
	...	...	...	...	...	...	...
	0x140	320th input block	Unsigned8	ro	Y		320th digital input block

16bit Network input variables

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0xA100	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
	...	...	...	...	...	...	...
	0xA0	160th input block	Unsigned16	ro	N		160th digital input block

32bit Network input variables

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0xA200	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
	...	...	...	...	...	...	...
	0x50	80th input block	Unsigned32	ro	N		80th digital input block

64bit Network input variables

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0xA440	0x00	64bit digital input block	Unsigned8	ro	N	depending on the compo-nents fitted	Number of available digital 64bit input blocks
	0x01	1st input block	Unsigned32	ro	N		1st digital input block
	...	...	...	...	...	...	...
	0x28	40th input block	Unsigned32	ro	N		40th digital input block

8bit Network output variables

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0xA400	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
	...	...	...	...	...	...	...
	0x140	320th output block	Unsigned8	rw	Y		320th digital output block

16bit Network output variables

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0xA580	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
	...	...	...	...	...	...	...
	0xA0	160th output block	Unsigned16	rw	N		160th digital output block

32bit Network output variables

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0xA680	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
	...	...	...	...	...	...	...
	0x50	80th output block	Unsigned32	rw	N		80th digital output block

64bit Network output variables

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0xA8C0	0x00	64bit digital input block	Unsigned8	ro	N	Depending on the compo-nents fitted	Number of available digital 64bit output blocks
	0x01	1st output block	Unsigned32	rw	N		1st digital output block
	...	...	...	...	...	...	...
	0x50	40th output block	Unsigned32	rw	N		40th digital output block

IM 253CAN - CANopen slave - Structure

Properties

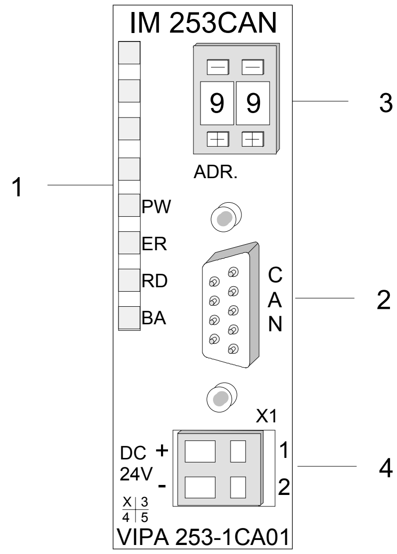
- 10 Rx and 10 Tx PDOs
- 2 SDOs
- Support of all baudrates
- PDO linking
- PDO mapping

Restrictions 253-1CA30 - ECO

The IM 253-1CA30 - ECO is functionally identical to the IM 253-1CA01 and has the following restrictions:

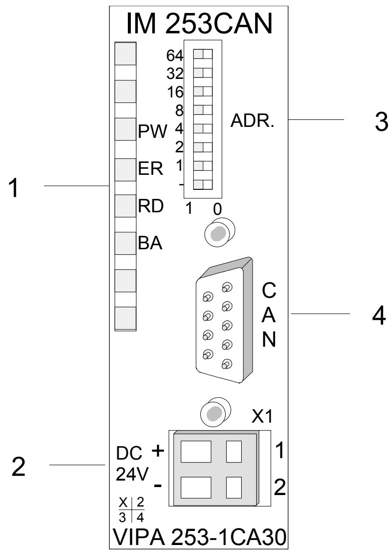
- CANopen slave for max. 8 peripheral modules
- Integrated DC 24V power supply for the peripheral modules 0.8A max.
- The CAN-Bus address can be adjusted by DIP switch.

Front 253-1CA01



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

Front 253-1CA30 - ECO



[1] LED status indicators
[2] Connector for an external 24V supply
[3] Address or baudrate selector (DIP switch)
[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	green	Indicates that the supply voltage is available.
ER	red	Blinks at overflow of the error counters (e.g. there is no further CAN station at the bus or wrong CAN transfer rate)
		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

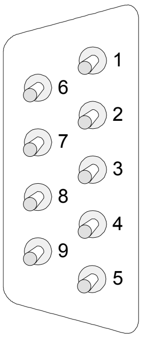
Various combinations of the LEDs indicate the different operating states:

LEDs	State
PW on ER on RD on BA on	Error during RAM or EEPROM initialization
PW on ER blinks 1Hz RD blinks 1Hz BA blinks 1Hz	Baudrate setting activated
PW on ER blinks 10Hz RD blinks 10Hz BA blinks 10Hz	Error in the CAN baudrate setting
PW on ER off RD blinks 1Hz BA off	Module-ID setting activated

9pin D-type socket

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

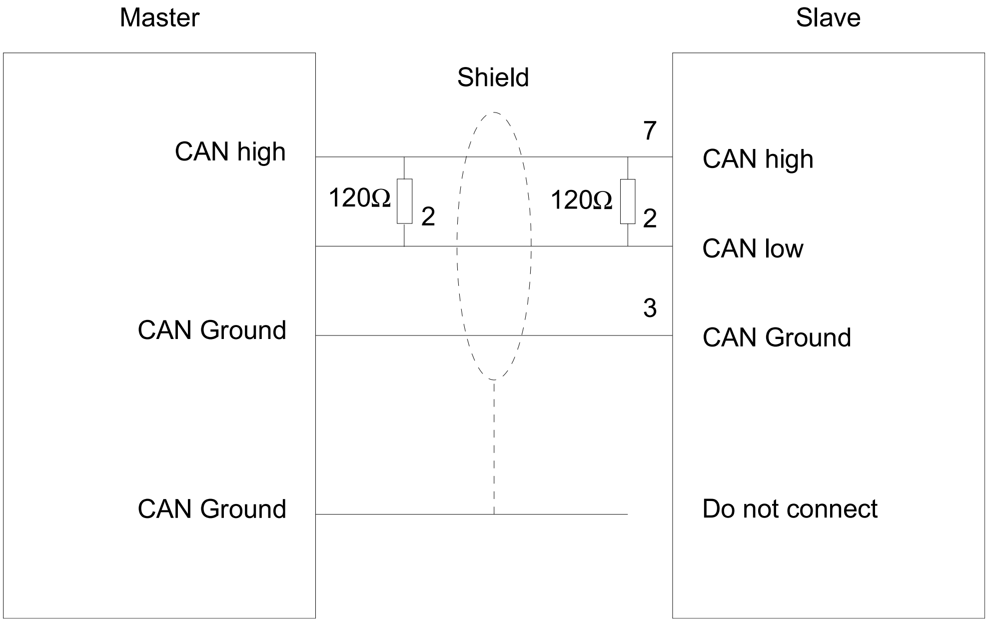
The following diagram shows the pin assignment for the interface.



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

CAN-Bus wiring

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



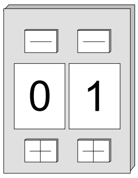
Line termination

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!

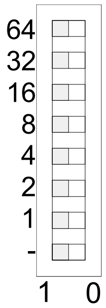
Address selector for Baudrate and module-ID



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

For details please refer to "IM 253CAN - CANopen slave - Baudrate and module-" in this chapter.

Address selector
IM 253-1CA30 - ECO



Contrary to the coding switch described above the IM 253-1CA30 - ECO is equipped by a DIL switch for addressing.

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 back plane current of the IM 253-1CA30 - ECO is limited to 0.8A.

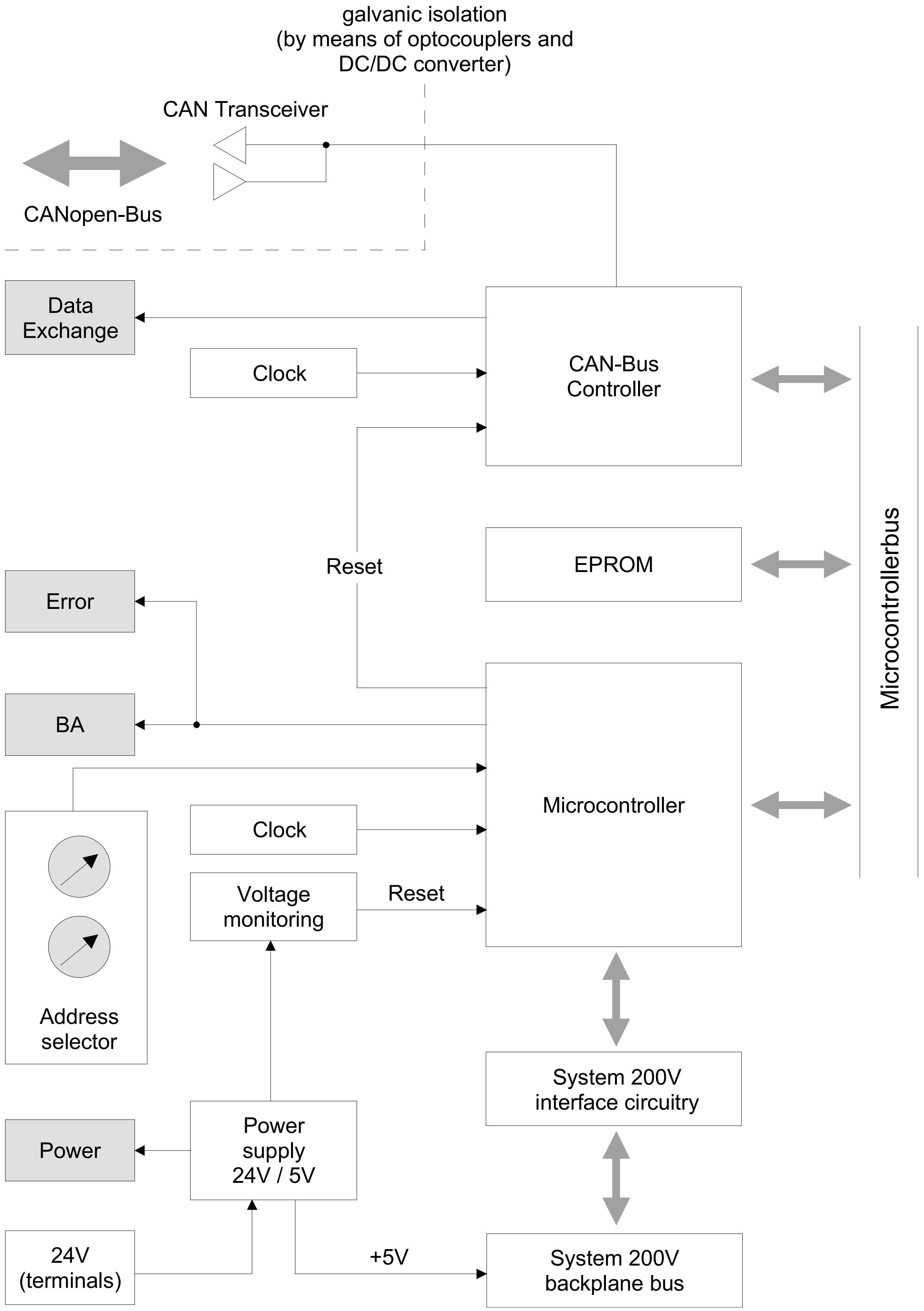
The power supply is protected against reverse polarity.

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

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

Block diagram

The following block diagram shows the hardware structure of the bus coupler and the internal communication:

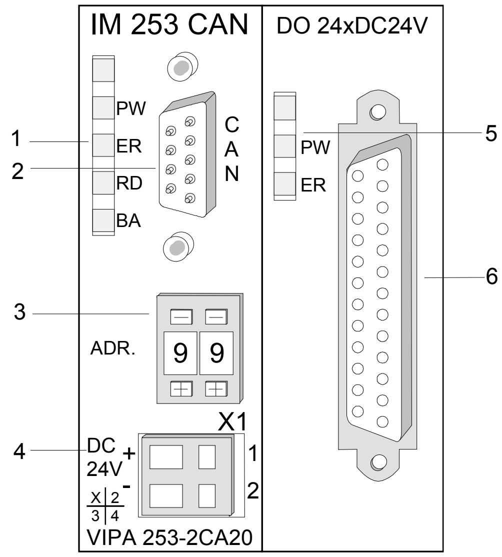


IM 253CAN, DO 24xDC 24V - Structure

Properties

- CANopen slave with 24 digital outputs on-board
- Project engineering via standard tools (e.g. SyCon from Hilscher)
- 1 Rx PDO
- 2 SDOs
- Support of all baudrates
- PDO linking
- PDO mapping: fix

Structure



[1] LED status indicators
[2] CAN-Bus socket
[3] Address or baudrate selector
[4] Connector for an external 24V supply
[5] LEDs status indicator output unit
[6] 25pin jack for digital output

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	green	Indicates that the supply voltage is available.
ER	red	Blinks at overflow of the error counters (e.g. there is no further CAN station at the bus or wrong CAN transfer rate).
		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
ER on
RD on
BA on

Error during RAM or EEPROM initialization

PW on
ER blinks 1Hz
RD blinks 1Hz
BA blinks 1Hz

Baudrate setting activated

PW on
ER blinks 10Hz
RD blinks 10Hz
BA blinks 10Hz

Error in the CAN baudrate setting

PW on
ER off
RD blinks 1Hz
BA off

Module-ID setting activated

LEDs digital output unit

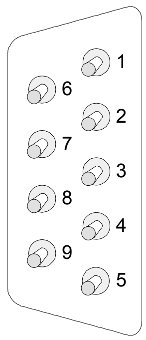
The digital output unit provides 2 LEDs with the following function:

Label	Color	Description
PW	green	Signalizes applying voltage via Profibus unit (Power).
ER	red	On at short circuit, overload and overheat

9pin D-type socket

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

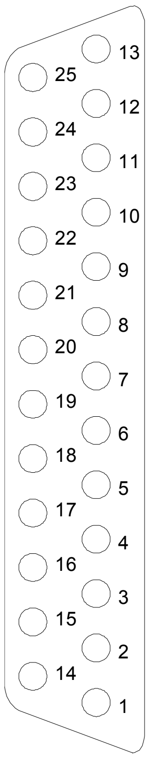
The following diagram shows the pin assignment for the interface.

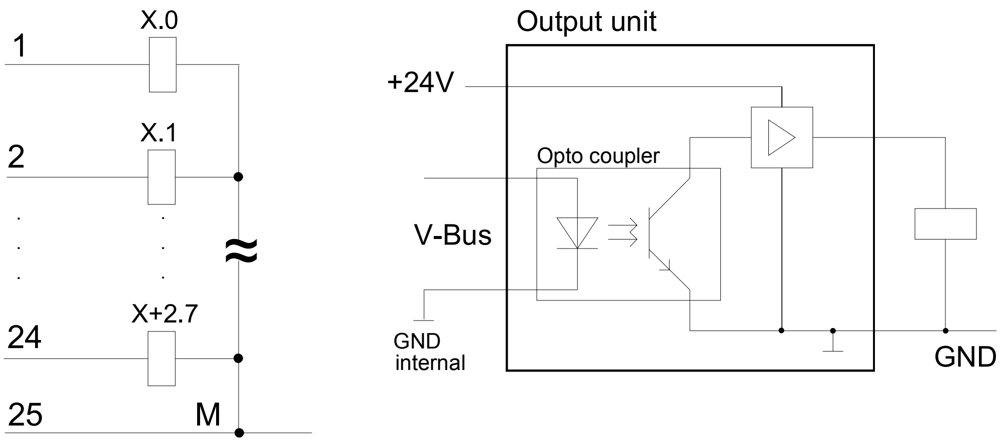


Pin	Assignment
1	n.c.
2	CAN low
3	CAN ground
4	n.c.
5	n.c.
6	n.c.
7	CAN high
8	n.c.
9	n.c.

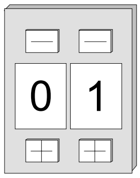
Output unit: Connection and schematic diagram

The DC 24V voltage supply of the output section happens via the power supply of the slave unit.





Address selector for baudrate and module-ID



The address selector is used to specify the module-ID as well as the CAN baudrate.

For details please refer to the section under the heading "Adjusting baudrate and module-ID" in this chapter.

Power supply

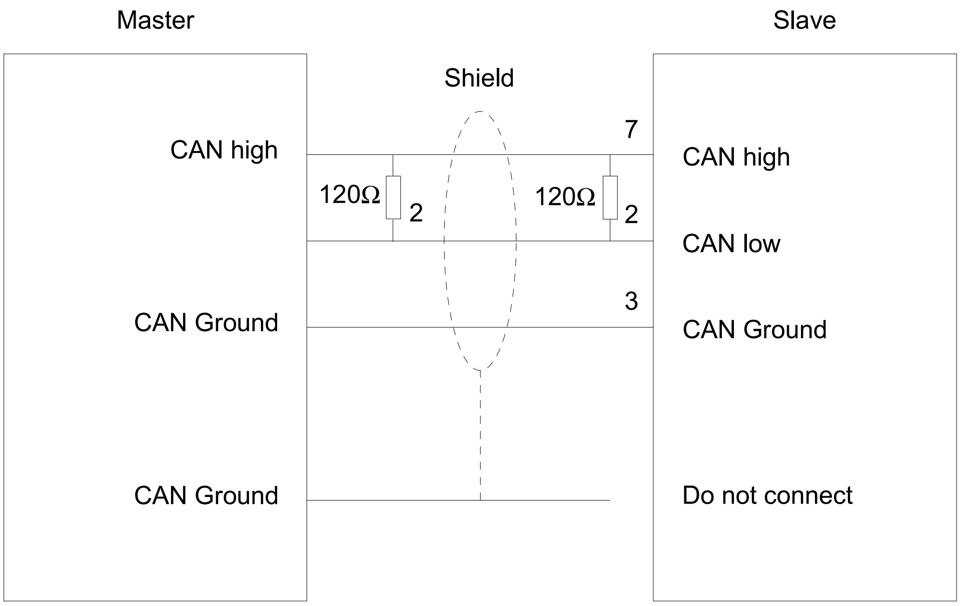
The CAN-bus coupler is equipped with an internal power supply. This power supply requires an external supply of DC 24V. In addition to the internal circuitry of the bus coupler the supply voltage is also used to power any devices connected to the backplane bus. Please note that the maximum current available for the backplane bus from the internal power supply is limited to 3.5A.

The power supply is protected against reverse polarity.

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

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!

IM 253CAN - 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 200V 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 baudrate and module-ID

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

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

For details please refer to the section under the heading "IM 253CAN - CANopen slave - Baudrate and module-" in this chapter.

CAN identifier

The CAN identifier for the in-/output data of the System 200V 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 200V 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 200V 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 200V 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**	**Info0**	**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 200V, read inputs, write outputs and control the modules.

In the following, the functions are described in detail.

IM 253CAN - CANopen slave - Baudrate and module-ID

Outline

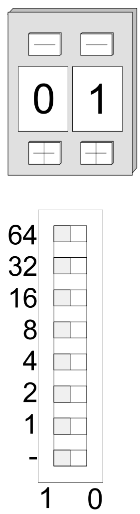
You have the option to specify the baudrate and the module-ID by setting the address selector to 00 within a period of 10s after you have turned the power on.

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 baudrate 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 baudrate by means of the address selector:



Address selector	CAN baudrate	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 baudrate 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").

Baudrate selection by an SDO-write operation

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

IM 253CAN - 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 253 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

The VIPA CAN-Bus coupler IM 253 CAN with DO 24xDC 24V supports the following objects:

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

CANopen function codes

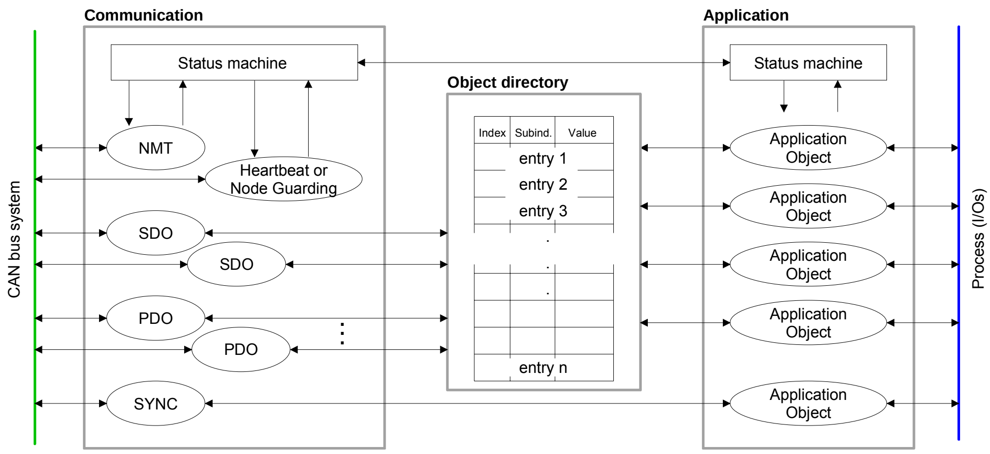
Every object is associated with a function code. You can obtain the required function code from the following table:

Object	Function code (4 bits)	Receiver	Definition	Function
NMT	0000	Broadcast	CiA DS-301	Network managem.
EMERGENCY	0001	Master	CiA DS-301	Error message
PDO1S2M	0011	Master, Slave (RTR)	CiA DS-301	Digital input data 1
PDO1M2S	0100	Slave	CiA DS-301	Digital output data 1
SDO1S2M	1011	Master	CiA DS-301	Configuration data
SDO1M2S	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 253CAN - 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 253CAN 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.

Note!

The IM 253CAN with DO 24xDC 24V provides only 1 receive PDO, the PDO mapping is fix.

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 200V 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 "nth" 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 253CAN - 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 253CAN - 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 – 0x1018	General communication specific parameters (e.g. device name)
0x1400 – 0x140F	Communication parameters (e.g. identifier) of the receive PDOs
0x1600 – 0x160F	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 – 0x180F 0x1A00 – 0x1A0F	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 baudrate (baudrate 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
0x1008		Device name
0x1009		Hardware version
0x100A		Software version
0x100B		Node number
0x100C		Guard time
0x100D		Life time factor
0x100E		Node Guarding Identifier
0x1010	X	Save parameter
0x1011	X	Load parameter
0x1014		Emergency COB-ID
0x1016	X	Heartbeat consumer time
0x1017	X	Heartbeat producer time
0x1018		Device identification
0x1027		Module list
0x1029		Error behavior
0x1400 - 0x1409	X	Communication parameter for receive PDOs (RxPDO, Master to Slave)
0x1600 - 0x1609	X	Mapping parameter for receive PDOs (RxPDO)
0x1800 - 0x1809	X	Communication parameter for transmit PDOs (TxPDO, Slave to Master)
0x1A00 - 0x1A09	X	Mapping parameter for transmit PDOs (TxPDO)
0x2001		CAN-Baudrate
0x2100		Kill EEPROM
0x2101		SJA1000
0x2400	X	PDO Control
0x3001 - 0x3010	X	Module Parameterization
0x3401	X	Module Parameterization
0x6000		Digital-Input-8-Bit Array (see DS 401)
0x6002	X	Polarity Digital-Input-8-Bit Array (see DS 401)
0x6100		Digital-Input-16-Bit Array (see DS 401)
0x6102		Polarity Digital-Input-16-Bit Array (v DS 401)
0x6120		Digital-Input-32Bit Array (see DS 401)
0x6122		Polarity Digital-Input-32-Bit Array (see DS 401)
0x6200		Digital-Output-8-Bit Array (see DS 401)
0x6202	X	Polarity Digital-Output-8-Bit Array (see DS 401)
0x6206	X	Fault Mode Digital-Output-8-Bit Array (see DS 401)
0x6207	X	Fault State Digital-Output-8-Bit Array (see DS 401)
0x6300		Digital-Output-16-Bit Array (see DS 401)

continue ...

... continued object directory overview

Index		Content of Object
0x6302		Polarity Digital-Output-16-Bit Array (see DS 401)
0x6306		Fault Mode Digital-Output-16-Bit Array (see DS 401)
0x6307		Fault State Digital-Output-16-Bit Array (see DS 401)
0x6320		Digital-Output-32-Bit Array (see DS 401)
0x6322		Polarity Digital-Output-32-Bit Array (see DS 401)
0x6326		Fault Mode Digital-Output-32-Bit Array (see DS 401)
0x6327		Fault State Digital-Output-32-Bit Array (see DS 401)
0x6401		Analog-Input Array (see DS 401)
0x6411		Analog-Output Array (see DS 401)
0x6421	X	Analog-Input Interrupt Trigger Array (see DS 401)
0x6422		Analog-Input Interrupt Source Array (see DS 401)
0x6423	X	Analog-Input Interrupt Enable (see DS 401)
0x6424	X	Analog-Input Interrupt Upper Limit Array (see DS 401)
0x6425	X	Analog-Input Interrupt Lower Limit Array (see DS 401)
0x6426	X	Analog-Input Interrupt Delta Limit Array (see DS 401)
0x6443	X	Fault Mode Analog-Output Array (see DS 401)
0x6444	X	Fault State Analog-Output Array (see DS 401)

X = save into EEPROM

Device Type

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1000	0	Device Type	Unsigned32	ro	N	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 253 1CA01 = VIPA CANopen slave IM 253-1CA01

VIPA IM 253 1CA30 = VIPA CANopen slave IM 253-1CA30 - ECO

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 253 1CA01 = 1.00
VIPA IM 253 1CA30 = 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 253 1CA01 = 3.xx
VIPA IM 253 1CA30 = 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, but can itself be monitored by the master (node guarding).

Guarding identifier

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

Save parameters

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

By writing the 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 default values

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

By writing the 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 253-1CA01: 0x2531CA01
at 253-1CA30: 0x2531CA30

Modular Devices

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 Output-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
SM 238	45DCh 38C4h	8 16	4 16
CP 240	1CC1h	16	16
FM 250	B5F4h	10	10
FM 250-SSI	B5DBh	4	4
FM 253, FM 254	18CBh	16	16

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 (0) or not (1), 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 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 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 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 (0) or not (1), 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 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 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 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 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 baudrate

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

This index entry writes a new baudrate into the EEPROM.

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

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

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 to ID.3) ≡ (AC.7 to AC.0)] ∨ (AM.7 to AM.0) ≡ 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.

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.

8bit digital inputs

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

8bit polarity digital inputs

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

16bit digital inputs

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	36nd input block	Unsigned16	ro	N		36nd digital input block

16bit polarity digital inputs

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

32bit digital inputs

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

32bit polarity digital inputs

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

8bit digital outputs

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

8bit change polarity digital outputs

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

8bit error mode digital outputs

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

8bit error value digital outputs

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.

16bit digital outputs

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

16bit change polarity digital outputs

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

16bit error mode digital outputs

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

16bit error value digital outputs

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.

32bit digital outputs

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

32bit change polarity digital outputs

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6322	0x00	32bit digital input block	Unsigned8	ro	N	Depending on the components 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

32bit error mode digital outputs

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6326	0x00	32bit digital input block	Unsigned8	ro	N	Depending on the components 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

32bit error value digital outputs

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 inputs

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 outputs

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 selection

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

Event driven analog inputs

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)

Upper limit value analog inputs

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.

Lower limit value analog inputs

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6425	0x00	Number of Inputs	Unsigned8	ro	N	depending on the components 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.

Delta function

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6426	0x00	Number of Inputs	Unsigned8	ro	N	depending on the components 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).

Analog output error mode

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

Analog output error value

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 253CAN - 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 253CAN - CANopen slave - NMT - network management

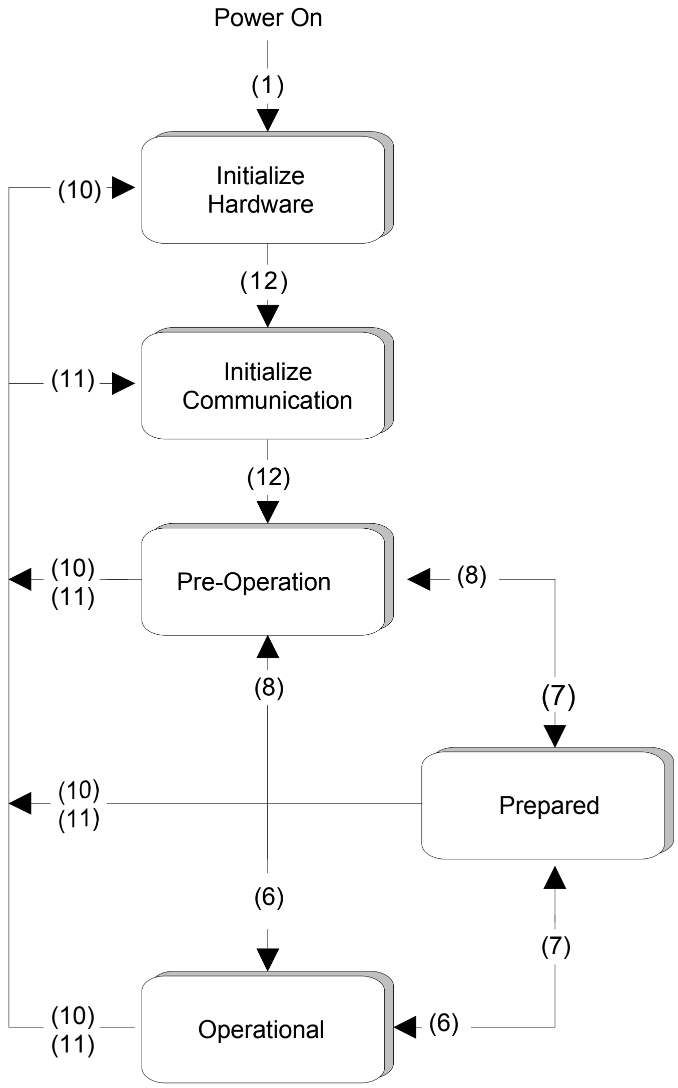
Network management (NMT) provides the global services specifications for network supervision and management. This includes the login and logout of the different network devices, the supervision of these devices as well as the processing of exceptions.

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 master
IM 208 CAN

Electrical data	VIPA 208-1CA00
Power supply	via backplane bus
Current consumption (rated value)	1A
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
Max. no. of stations	127 stations (depending on the master interface)
Combination with peripheral modules	
Max. number of slaves	125
Max. number of TxPDOs	40
Max. number of RxPDOs	40
Max. number of input bytes	384
Max. number of output bytes	384
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x78
Weight	110g

CANopen slave
IM 253CAN

Electrical data	VIPA 253-1CA01	VIPA 253-1CA30 - ECO
Power supply	DC 24V (20.4 ... 28.8) via front from ext. power supply	
Current consumption (in no-load operation)	50mA	50mA
Current consumption (rated value)	max. 0.8A	max. 0.3A
Output current backplane bus	max. 3.5A	max. 0.8A
Power loss	2W	1.5W
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.	Any combination of max. of 8 I/O modules per coupler.
Max. no. of stations	127 stations (depending on the master interface)	
Combination with peripheral modules		
max. no. of modules	32 (depending on current consumption)	8
max. inputs/outputs	80Byte each (80Byte = 10 PDOs à 8Byte)	
Dimensions and weight		
Dimensions (WxHxD) in mm	25.4x76x78	
Weight	80g	

CANopen slave
IM 253CAN,
DO 24xDC 24V

Electrical data	VIPA 253-2CA20
Power supply Current consumption at L+ Output current backplane bus Isolation Status indicator Connectors/interfaces	DC 24V (20.4 ... 28.8) via front from ext. power supply max. 5A 3.5A ≥ AC 500V by means of LEDs located on the front 9pin D-type (socket) CAN-Bus connection
CAN-Bus interface	
Connection Network topology Medium Data transfer rate Max. overall length Max. no. of stations	9pin D-type plug Linear bus, active bus termination at one end, tap lines permitted. Screened three-core cable, unscreened cable permitted - depending on environment. 10kBaud to 1MBaud 1000m at 50kBaud without repeaters 127 stations (depending on the master interface)
Output unit	
Number of outputs Nominal load voltage Output current per channel Status monitor	24 DC 24V (20.4...28.8V) internal via CAN coupler 1A (Total current max. 4A) Power (PW) fuse ok, Error (ER) short circuit, overload
Programming data	
Output data	3Byte
Dimensions and weight	
Dimensions (WxHxD) in mm Weight	50.8x76x78 150g

Chapter 6 DeviceNet

Overview

This chapter contains the description of the VIPA DeviceNet slave. The introduction to the system is followed by the description of the module. Another section of this chapter concerns the configuration by means of the *DeviceNet-Manager* of Allen - Bradley This section describes the configuration of the DeviceNet coupler and the System 200V modules.

A summary of the diagnostic messages and the technical data conclude the chapter.

Below follows a description of:

- DeviceNet basics
- Hardware description of the VIPA DeviceNet coupler IM 253DN
- Configuration by means of the *DeviceNet-Manager* inc. examples
- Diagnostics
- Technical data

Content

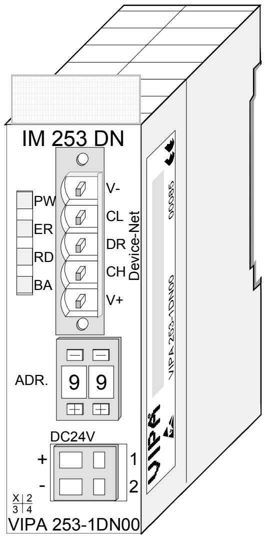
Topic **Page**

Chapter 6 DeviceNet ... 6-1
System overview ... 6-2
Basics ... 6-3
IM 253DN - DeviceNet coupler - Structure ... 6-5
Configuration by means of the DeviceNet-Manager ... 6-8
Specifying baudrate and DeviceNet address ... 6-9
Test in conjunction with the DeviceNet ... 6-10
Module configuration in the DeviceNet-Manager ... 6-11
I/O addressing of the DeviceNet scanner ... 6-16
Diagnostics ... 6-17
Technical data ... 6-22

System overview

You can use the VIPA DeviceNet coupler to link-up up to 32 modules of your System 200V periphery by means of DeviceNet.

The following DeviceNet components are currently available from VIPA.



Order data DeviceNet

Type	Order number	Description
IM 253DN	VIPA 253-1DN00	DeviceNet coupler

Basics

General

DeviceNet is an open low-end network that is based upon the physical properties of CAN-Bus. The bus is also used to supply the devices with the required DC 24V power.

You can use DeviceNet to install direct connections between your control system and simple industrial devices like sensors and switches as well as technologically advanced devices like frequency converters and barcode readers.

Direct interfacing improves communications between the different devices and provides important diagnostic facilities at the device level.

DeviceNet

DeviceNet is an open device net standard that satisfies the user profile for industrial real-time system applications.

The DeviceNet protocol has an open specification that is the property of and administered by the independent vendor organization "Open DeviceNet Vendor Association" ODVA.

This is where standardized device profiles are created to provide compatibility and exchangeability on logical level for simple devices of the same type.

In contrast to the classical source–destination model, DeviceNet uses a modern producer/consumer model that requires data packets with identifier fields for the identification of the data.

This approach caters for multiple priority levels, more efficient transfers of I/O data and multiple consumers for the data.

A device that has data to send *produces* the data on the network together with an identifier. All devices requiring data listen for messages. When a device recognizes a suitable identifier, they act and *consume* the respective data.

DeviceNet carries two types of messages:

- *I/O messages*
  Messages that are subject to critical timing constraints and that are contain data for control purposes that can be exchanged by means of a single or multiple connections and that employ identifiers with a high priority.
- *explicit messages*
  These are used to establish multi-purpose point-to-point communication paths between two devices which are used for the configuration of network couplers and for diagnostic purposes. These functions usually employ identifiers of a low priority.

Messages that are longer than 8Byte are subject to the fragmentation service. A set of rules for master/slave, peer-to-peer- and multi-master connections is also available.

Communication medium

DeviceNet employs a master line/tap line topology with up to 64 network nodes. The maximum distance is either 500m at a rate of 125kBaud, 250m at a rate of 250kBaud or 100m at a rate of 500kBaud.

The length of the tap lines can be up to 6m while the total length of all spur lines depends on the baudrate.

Network nodes can be removed from or inserted into the network without interruption of the network operation. New stations and failed stations are detected automatically.

DeviceNet employs a screened five-core cable as data communication medium.

DeviceNet uses voltage differences and for this reason it exhibits less sensitivity to interference than a voltage or current based interface.

Signals and power supply conductors are included in the same network cable. It is therefore possible to connect devices that obtain the operating voltage via the network as well as devices with an integrated power supply. Furthermore it is possible to connect redundant power supplies to the network that guarantees the power supply when required.

Bus access method

DeviceNet operates according to the Carrier-Sense Multiple Access (CSMA) principle, i.e. every station on the network may access the bus when it is not occupied (random access).

The exchange of messages is message orientated and not station orientated. Each message is provided with a unique and priorizing identifier. At any time only one station is able to occupy the bus with its messages.

The DeviceNet bus access control is subject to non-destructive, bit-wise arbitration. In this case non-destructive means that the successful station participating in the arbitration doesn't need to re-send its message. The most important station is selected automatically when multiple stations access the bus simultaneously. If a station that is ready to send recognizes that the bus is occupied, its send request is delayed until the current transfer has been completed.

Addressing

All stations on the bus must be uniquely identified by means of an ID address. Every DeviceNet device has addressing facilities.

EDS file

The properties of the DeviceNet units are supplied in the form of an EDS file (**E**lectronic **D**ata **S**heet) to configure a slave interface by means of your configuration tool.

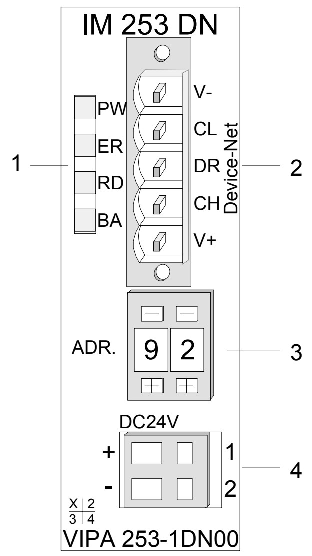
IM 253DN - DeviceNet coupler - Structure

Properties

The DeviceNet coupler IM 253DN provides a simple method of interfacing any decentral peripheral modules by means of the DeviceNet protocol.

- Group 2 only Device
  - employs the predefined connection set
- Poll only Device
  - no BIT STROBE mode support
  - no CHANGE OF STATE support
- supports all baudrates: 125, 250 and 500kBaud
- address selection by means of switches
- definition of the data rate by means of a special POWER ON procedure (start from address 90...92)
- LED status indicators
- a max. of 32 peripheral modules can be installed
- of these a max. of 8 may be configurable modules
- module configuration by means of the *DeviceNet-Manager*

Front view 253-1DN00



[1] LED status indicator
[2] DeviceNet connector
[3] Address selector
[4] DC 24V power supply connector

Components

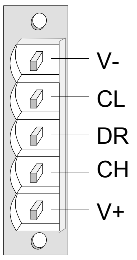
LEDs

4 LEDs on the front show the current status of the module for the quick troubleshooting. A detailed description of the troubleshooting procedure by means of the LEDs and the backplane is available in a section of the chapter "diagnostics".

Label	Color	Description
PW	green	Power-LED: supply voltage available
ER	red	DeviceNet or backplane bus bus error
RD	green	Backplane bus status
BA	yellow	DeviceNet status

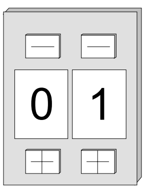
DeviceNet interfacing

The DeviceNet connection is provided by a 5pin Open Style connector. The pin assignment is imprinted on the front of the module.



[V-] GND operating voltage
[CL] CAN low
[DR] DRAIN
[CH] CAN HIGH
[V+] DC 24V operating voltage

Address selector



The address selector is used for:

- the definition of the unique DeviceNet address
- programming of the baudrate

Addresses:

0...63: DeviceNet address

90, 91, 92: set communication rate to 125, 250, 500kBaud

Power supply

Every DeviceNet 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.

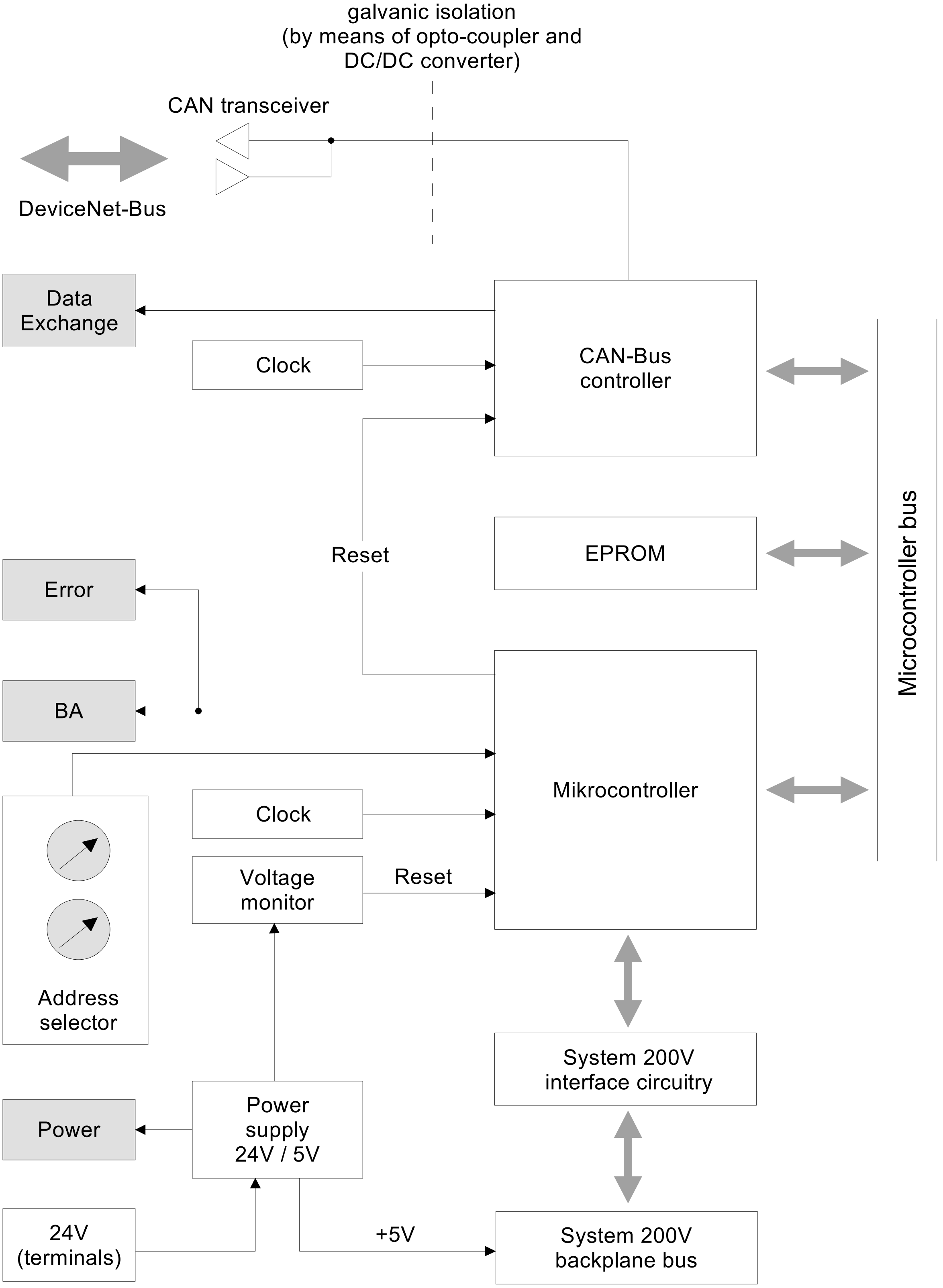
The power supply is protected against reverse polarity.

DeviceNet and backplane bus are galvanically isolated from each other.

Note!

The DeviceNet coupler does not require any current from the power that is available via the DeviceNet.

Block diagram The following block diagram shows the hardware structure of the bus coupler in principle as well as the internal communication:



Configuration by means of the DeviceNet-Manager

Overview

The DeviceNet is configured by means of the *DeviceNet-Manager* software from Allen - Bradley.

The following steps are necessary for the configuration:

- Configuration of the *DeviceNet-Manager*
- Set baudrate and DeviceNet address of the module
- Test the DeviceNet
- Module configuration
- I/O addressing of the DeviceNet scanner (master)

Configuration of the DeviceNet-Manager

During the configuration the module specific data of the VIPA DeviceNet coupler are defined and supplied to the *DeviceNet-Manager*.

The following steps are required:

- Insert the supplied disc into your PC.
- Copy the file **IM253DN.BMP** to your PC into the directory **/DNETMGR/RES** of the *DeviceNet-Manager*
- The EDS file is located in a sub-directory of 501.VND on the disc. Copy the file **1.EDS** into the directory **/DNETMGR/EDS/501.VND/0.TYP/-1.COD**

You can also copy the entire tree

```
501.vnd
    |-- 0.typ
        |--1.cod
            |-- 1.eds
            |-- device.bmp
```

into the directory DNETMGR/EDS.

Specifying baudrate and DeviceNet address

You may set the baudrate as well as the DeviceNet address when the power has been turned off. These will be transferred into the module when you turn the respective power supply on.

Setting the baudrate

All stations connected to the bus communicate at the same baudrate. You may define the required rate by means of the address selector.

- Turn off the power supply
- Set the address selector to the wanted baudrate

Setting	baudrate in kBaud
90	125
91	250
92	500

- Turn on the power supply
  The selected transmission rate is saved to the EEPROM.
  At this point your DeviceNet coupler is set to the correct baudrate.

LED-indicator RD-LED ER-LED

When the baudrate has been saved successfully, the RD-LED (green) will be turned on.

When the baudrate was selected incorrectly, the ER-LED will be turned on.

Setting the DeviceNet address

All stations connected to the bus must have a unique DeviceNet address. The address can be defined by means of the address selector when the supply has been turned off.

- Turn off the power supply
- Set the address selector to the required address.
  Please ensure that the address is unique in the system and that it is located between 0 and 63.
- Turn on the power supply
  The selected addressis saved to the RAM.

Note!

Any changes to the addressing will only become effective after a POWER ON or an automatic reset. Changes to settings are not recognized during normal operations.

LED indicator ER-LED

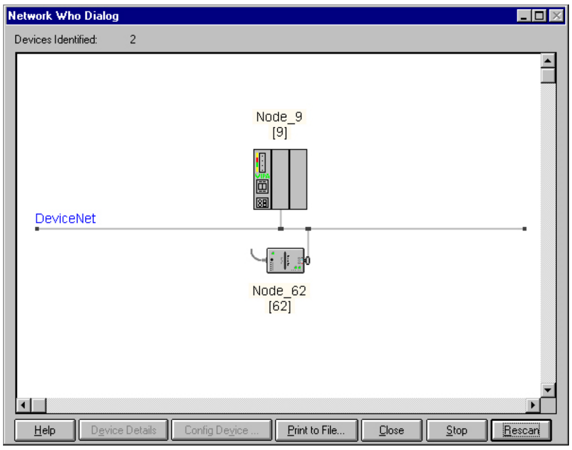
When the address is not valid or if it already exists the ER-LED (red) will be turned on after power on.

Test in conjunction with the DeviceNet

Approach

- Connect the PC containing the *DeviceNet-Manager* and the VIPA DeviceNet coupler to the DeviceNet.
- Define the baudrate and the node address at the coupler
- Turn on the power supply of the bus coupler
- Start the *DeviceNet-Manager*.
- Enter the same data rate into the manager that was selected at the bus coupler
- Start the function NETWORK WHO in the manager

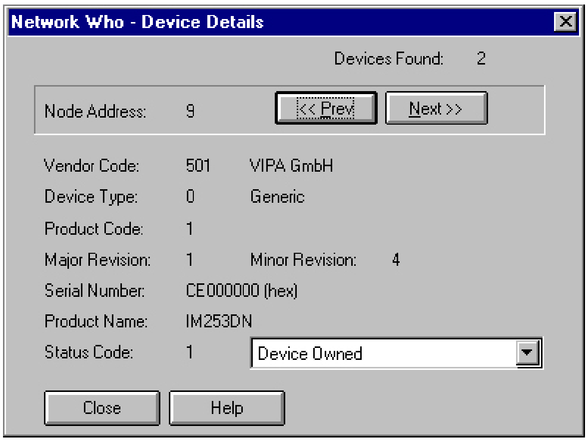
The following network windows is displayed:



Device Details

- Right-click the bus coupler.
- Select the function DEVICE DETAILS in the context menu.

The DEVICE DETAILS box is displayed on screen



Here you may display DeviceNet address (node address), the Vendor Code (in this case this is 501 for VIPA GmbH) and other internal information about every module on the bus.

Module configuration in the DeviceNet-Manager

The System 200V includes configurable modules like analog modules. When you are using these modules in conjunction with a DeviceNet coupler the respective parameters have to be saved in the DeviceNet coupler.

Configuration in groups

The following conditions apply to the configuration:

- DeviceNet manages the parameter data in groups.
- Every DeviceNet coupler is able to process and store a maximum of 144Byte of parameter data.
- These 144Byte are divided into 8 groups of 18Byte each.
- Every group can contain the parameter data of 1 module.
- Groups are identified by a prefix-No. (1...8) in the parameter name.
- The number of parameter bytes is defined in the parameter "Len" (1^{st} parameter) of a group. The number of parameter bytes is available from the technical data contained in the documentation on the peripheral modules.
- The group allocation for a module does not depend on the location or the installation sequence.
- The allocation of the plug-in location is defined by means of the "Slot"-parameter of a group (2^{nd} parameter)
- The values may be entered as bit patterns by double-clicking a parameter
- Unused groups are identified by a "Value" 0000 0000.

Approach

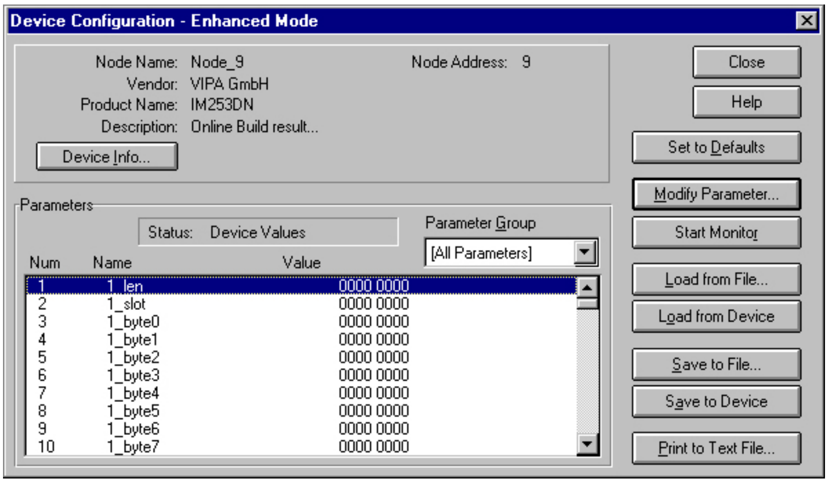
Precondition: The IM 253DN coupler is active on the bus.

Below follows a description of how the parameter settings are defined in the *DeviceNet-Manager*.

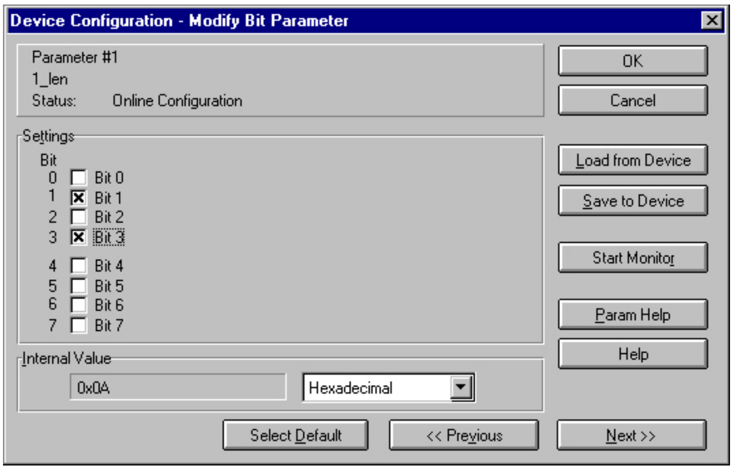
- Execute the function WHO in the *DeviceNet-Manager*.

  This will open a network window that includes your coupler.
- Double-click the icon of the bus coupler where you want to modify the parameter data.

  The parameters are read from the coupler and displayed in the following window:



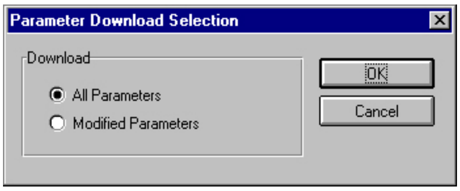
- Locate an unused group in the list of parameters (Value=0000 0000)
  You may display all 8 groups in the parameter list by choosing "All Parameters" in the selection field *Parameter Group*.
- Double click the "Len"-parameter
  The following dialog box is displayed:



- Enter the number of parameter bytes (bit coded) of the module that you are configuring. You can obtain the number from the documentation for the peripheral module. Set or reset the respective bits by clicking the checkbox.
- Click [OK] to close the mask. The next parameter (slot) of the same group is displayed when you click the button [Next>>].
- Now you have to enter the plug-in location number of the module you are configuring as a bit-code in the same manner.
  You can retrieve the input range by means of the button [Param Help].
- At this point you can enter the parameter bytes for your module one after the other by clicking [Next >>].
- If you wish to configure other modules you have to select another unused group and proceed in the same manner.

- When you have entered all parameters into the different groups you transfer and save the parameters in the DeviceNet coupler by clicking the [Save to Device] button.

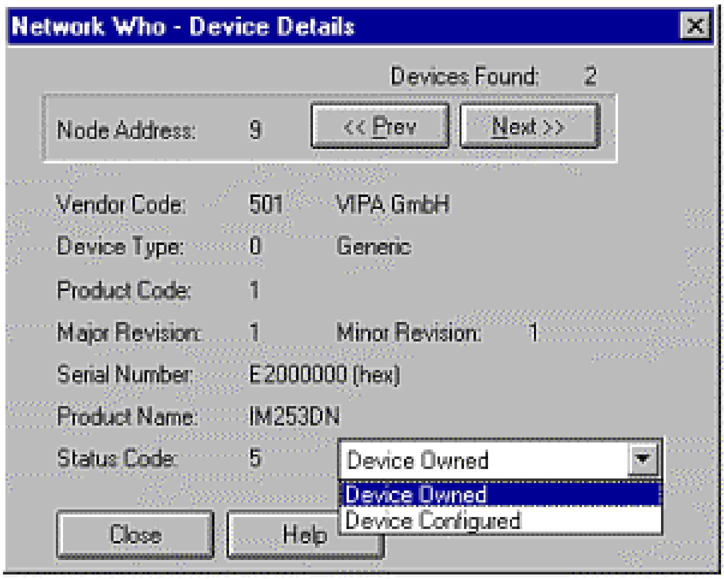
  The following selection window is opened:



Here you may decide whether you want to transfer all the parameters or only the parameters that were modified.

- During the transfer the status text "Status: downloading" is displayed. When the transfer has completed, the status text changes to "Status: Device Values"

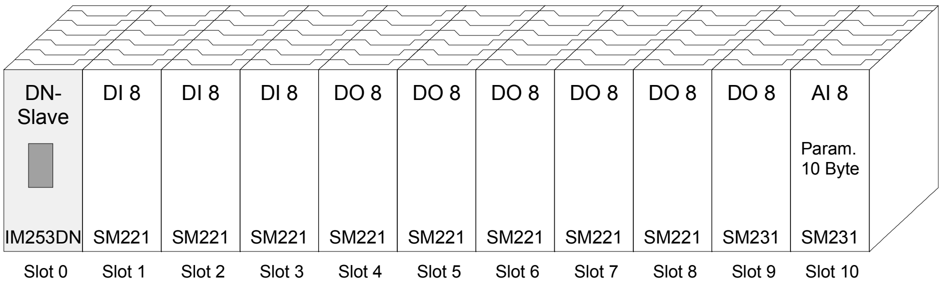
- If you were to request the "Device Details", you may see that the bit CONFIGURED is now also included in the status.



When you have entered the parameter values and downloaded them into the DeviceNet coupler, the peripheral modules connected via the backplane bus have been configured accordingly.

Example

The following example is intended to show the configuration of the System 200V. Let us assume that the system has the following structure:



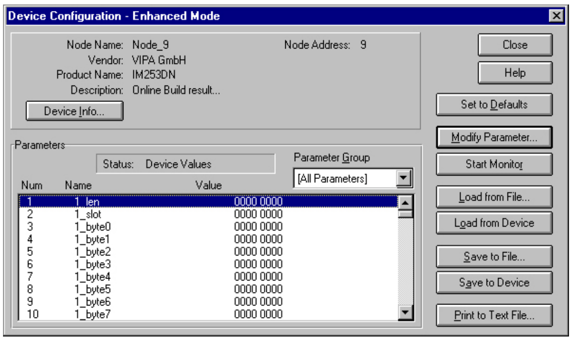
The example shows a DeviceNet coupler with 10 modules; however, the modules installed in plug-in locations 1 to 9 can not be configured.

Below follows the description of the configuration of the analog-module in location 10:

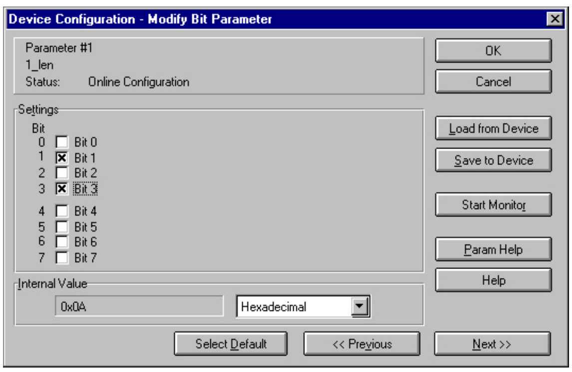
Precondition: - the hardware was assembled and is active on the bus.

- the Allen - Bradley *DeviceNet-Manager* was installed.

- Execute the function WHO in the *DeviceNet-Manager* and open the parameter window by double-clicking the DeviceNet coupler.



- Locate an unused group in the parameter list (Value=0000 0000)
- Double-click the "Len"-parameter.



The analog module has 10Byte of parameter data. Enter this value as a bit-coded value.

- Click [Next>>] and enter the location 10 as the "slot".
- You may now enter the parameter bytes of your module by clicking [Next >>] repeatedly.

The analog input module has the following parameters:

Byte	Bit 7 ... Bit 0	Default
0	Diagnostic alarm byte: Bit 5 ... 0: reserved Bit 6: 0: Diagnostic alarm inhibited 1: Diagnostic alarm enabled Bit 7: reserved	00h
1	reserved	00h
2	Function no. channel 0 (see module description)	2Dh
3	Function no. channel 1 (see module description)	2Dh
4	Function no. channel 2 (see module description)	2Dh
5	Function no. channel 3 (see module description)	2Dh
6	Option byte channel 0	00h
7	Option byte channel 1	00h
8	Option byte channel 2	00h
9	Option-byte channel 3	00h

- When all parameters have been entered into the group you transfer and save the parameters in the DeviceNet coupler by means of [Save to Device].
- During the transfer the status text is displayed as "Status: downloading". When the transfer has been completed the status text changes to "Status: Device Values"

Note!

Parameters may be changed at any time. For this purpose you have to click [Load from Device], then enter the required changes and save them by means of [Save to Device].

I/O addressing of the DeviceNet scanner

The DeviceNet coupler determines the modules installed on the backplane bus automatically and uses the result to generate the number of input and output bytes.

You have to determine these two values when you configure the input/output modules and enter them in the DeviceNet scanner (master):

- produced connection size (number of input bytes)
- consumed connection size (number of output bytes)

The addressing results from the sequence of the modules (plug-in location 1 to 32) and the base address that was defined in the DeviceNet scanner for the bus coupler.

DeviceNet scanner configuration

- Set the DeviceNet scanner to connection type POLL IO.
- Define the parameters:
  "Receive data size" = number of input bytes
  "Transmit data size" = number of output bytes
- Define the base address (mapping) of receive data and transmit data as required.
- Activate the DeviceNet coupler IM 253DN in the scan list.
- Start the DeviceNet scanner.

When the DeviceNet scanners have been configured, the input and output modules are accessible via the defined addresses.

Example

The following 6 modules have been installed into the backplane bus:

Plug-in location	Installed module	Input data	Output data
Slot 0	DeviceNet coupler	-	-
Slot 1	Digital Out SM 222		1Byte
Slot 2	Digital Out SM 222		1Byte
Slot 3	Digital In SM 221	1Byte	
Slot 4	Analog In SM 231	4Words	
Slot 5	Analog Out SM 232		4Words
Total:		1+4*2=9Byte	1+1+4*2=10Byte

The result is:

- produced connection size: 9Byte (sum of input bytes)
- consumed connection size: 10Byte (sum of output bytes)

Diagnostics

Overview

The LEDs installed to display the status allow extensive diagnostics during the POWER ON - procedure as well as during operation. The result of the diagnosis is determined by the combination of the different LEDs and the current operating mode.

Explanation:

LED	Description
☐ off	LED turned off
☐ on	LED is permanently on
☒ blinks	LED blinks

The following operating modes are available depending on the position of the address selector:

- DeviceNet mode (address selector in position 0...63)
- Configuration mode (address selector in position 90...92)

DeviceNet mode

POWER ON without DeviceNet

LED	Description
☐ PW on ☐ ER off ☒ RD blinks ☐ BA off	After POWER ON the PW-LED is turned on and indicates a properly operating power supply. The RD-LED blinks since the configuration data, stored in the EEPROM, was transferred successfully into the peripheral modules
☐ PW on ☐ ER on ☐ RD off ☐ BA off	After POWER ON the PW-LED is turned on. The ER-LED is on due to errors on the backplane bus or when the configuration data could not be transferred into the peripheral modules.

POWER ON with DeviceNet without master

LED	Description
PW on ER off RD blinks BA blinks	After POWER ON the PW-LED is turned on. The RD-LED blinks because: • the backplane bus is operating properly • the configuration data was transferred successfully from the EEPROM into the configurable peripheral modules. The BA-LED blinks because: • at least one additional device is active on the DeviceNet, • and the address set up on the coupler is unique.
PW on ER on RD off BA off	After POWER ON the PW-LED is turned on. The ER-LED is on due to one of the following conditions on the DeviceNet coupler: • bad address or address occupied by another device • data transfer rate is bad.
PW on ER on RD blinks BA blinks	After POWER ON the PW-LED is on. The ER-LED is turned on when the configuration data could not be transferred into the configurable peripheral module. The RD-LED blinks because • the backplane bus is operating properly • the configuration data was not transferred into the configurable peripheral modules. The BA-LED blinks because • at least one other device is active on the DeviceNet, • the address set up on the coupler is unique.

POWER ON with DeviceNet and master

LED	Description
■ PW on ■ ER on ☒ RD blinks □ BA on	After POWER ON the PW-LED is on. The ER-LED is turned on since the configuration data was not transferred into the configurable peripheral modules. The RD-LED blinks because • the backplane bus operates properly • the configuration data was not transferred into the configurable peripheral modules. The BA-LED is turned on • because the coupler IM 253DN has established a DeviceNet-connection to a master. Note! The IM 253DN coupler executes a reset after 30s. An error that occurs during POWER ON with DeviceNet and master displays the same combination of LEDs as a hardware error. It is possible to distinguish between these cases: • by interruption of the DeviceNet connection → ER-LED and RD are blinking! • with a network WHO in the *DeviceNet-Manager* → in case of a hardware error the IM253DN will not appear on the network Please call the VIPA hotline if a hardware error occurs!

Proper operation with DeviceNet and master

LED	Description
■ PW on □ ER off ■ RD on □ BA on	After POWER ON the PW-LED is on. The RD-LED is turned on because the connection to the peripheral modules could be established via the backplane bus. The BA-LED is turned on because the coupler IM 253DN established a DeviceNet connection with a master.

Errors during the operation with DeviceNet and master

LED	Description
PW on ER on RD off BA on	After POWER ON the PW-LED is on. The ER-LED is turned on because an error was detected on the backplane bus. The BA-LED is turned on because the IM 253DN coupler established a DeviceNet connection with a master. **Note!** The IM 253DN coupler will execute a reset after 30s.

Change of state from operational to module error status

LED	Description
PW on ER on RD off BA off	The ER-LED is turned on for 1 second because a module error was detected. Subsequently the coupler IM 253DN will execute a reset. After the reset the coupler is re-started and it indicates the error by means of the respective LED combination.

Indicators after a re-start and a reset

LED	Description
PW on ER on RD blinks BA on	The ER-LED is turned on permanently and the RD-LED blinks because the quantity of I/O data was changed by the failure of the module. The configuration data could not be transferred. All Allen - Bradley scanners will display message #77.
PW on ER off RD on BA on	The ER-LED is not turned on and the RD-LED is permanently on because the quantity of I/O data was modified by the failure of the module. The connection with the I/O modules was established. All Allen - Bradley scanners will display message #77.

Change of state from operational to connection error status

LED	Description
PW on ER blinks RD blinks BA on	The ER-LED blinks because the timer of the I/O connection detected an error. The RD-LED blinks because the I/O-connection does not exist any longer. All inputs and outputs are set to zero. The BA-LED is turned on because the connection with the master is still established.

Configuration mode

POWER ON in configuration mode

LED	Description
PW on ER off RD on BA off	After POWER ON the PW-LED is turned on and indicates that the power supply operates properly. The RD-LED is turned on after a short delay since the baudrate was transferred into the EEPROM.

Device error

LED	Description
PW on ER on RD off BA off	The address that was set up on the coupler is not valid. Change the address to a valid setting: • 0...63 as DeviceNet address • 90...92 for the definition of the baudrate
PW on ER on RD on BA on	When the coupler is not connected to the DeviceNet, an error was detected in the internal EEPROM or in RAM. When a DeviceNet connection exists, it is also possible that an error has occurred during the transfer of the configuration data into the peripheral modules. Note! Errors that occur during POWER ON with DeviceNet and master display the same combination of LEDs as a hardware error. It is possible to distinguish between these cases: • by interruption of the DeviceNet connection → ER-LED and RD are blinking! • with a network WHO in the *DeviceNet-Manager* → in case of a hardware error the IM 253DN will not appear on the network Please call the VIPA hotline if a hardware error occurs!

Technical data

DeviceNet coupler
IM 253DN

Electrical data	VIPA 253-1DN00
Power supply Current consumption Output current backplane bus Isolation between DeviceNet and backplane bus	DC 24V (20.4 ... 28.8) via front from ext. power supply Bus coupler: 50mA incl. supply to the peripheral modules: 800mA max. 3.5A 500V rms
Function specific data	
Status indicator Physical connection to DeviceNet Network topology Communication medium Communication rate Overall length of the bus Number of stations	by means of LEDs on the front 5pin Open Style connector Linear bus, tap lines up to 6m length Screened 5core cable 125, 250, 500kBaud up to 500m max. 64
Combination with peripheral modules	
Number of modules Inputs Outputs	max. 32 max. 256Byte max. 256Byte
Dimensions and Weight	
Dimensions (BxHxT) Weight	25.4x76x78mm 80g

Chapter 7 SERCOS - Spare part

Outline

Content of this chapter is the description of the SERCOS coupler from VIPA. A system overview is followed by a description of the module. Another part of this chapter is the project engineering. With the help of examples we will explain the project engineering of the SERCOS coupler and the parameterization of the System 200V modules.

The description closes with an overview of diagnostic messages and the technical data.

The following text describes:

- SERCOS basics
- Hardware description of the SERCOS coupler IM 253SC from VIPA
- Description of the identifiers with assignment sample
- Example for the parameterization
- Technical data

Content

Topic **Page**

Chapter 7 SERCOS - Spare part ... 7-1

System overview ... 7-2

Basics ... 7-3

IM 253Sercos - SERCOS coupler - Structure ... 7-5

Basic parameterization via address adjuster ... 7-8

SERCOS Identifier ... 7-10

Example for the automatic ID assignment ... 7-13

Technical Data ... 7-22

Note!

For the deployment of the SERCOS coupler in this chapter, a thorough knowledge of SERCOS is required.

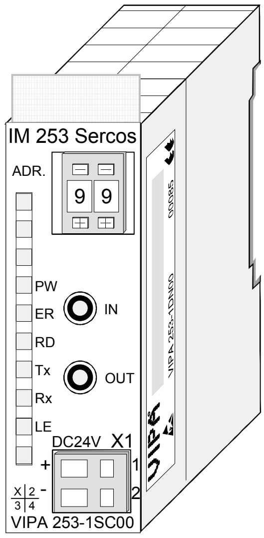
This manual describes exclusively the VIPA specific properties.

The description of the properties included in the SERCOS standard, like e.g. the identifiers S-0 and S-1 are to find, for example, in the SERCOS specification of the SERCOS Interface Committee.

System overview

With the SERCOS coupler from VIPA you may connect up to 32 modules of your 200V periphery to SERCOS.

The following SERCOS components are available from VIPA at this time.



Order data SERCOS

Type	Order number	Description
IM 253SC	VIPA 253-1SC00	SERCOS coupler

Basics

SERCOS

SERCOS means **Se**rial **R**eal Time **C**ommunication **S**ystem and has been established for numeric controls all over the world. Beyond the classic CNC machines this technique has proofed its worth as fast and precise motion control in the whole automation branch.

SERCOS, also called "SERCOS Interface", is a standardized digital drive interface based on fiber optic transmitter technology.

The high real-time demands and the interference secure fiber optic technology are distinguishing features of this bus system.

With the SERCOS coupler IM 253SC from VIPA, the SERCOS connection to the sensor/actuator level is now possible.

The SERCOS coupler is anticipated for the fast data exchange at the sensor/actuator level. Here, central controls, like e.g. a PLC, communicate with decentral in- and output modules via a fast serial connection. The data exchange with this decentral devices is executed cyclically.

The master reads the input information from the slaves (drive telegram) and sends the output information to the slaves (master data telegram).

A maximum of 254 slaves may be connected to one bus.

Communication

SERCOS knows three kinds of telegrams for the communication:

- *Master-Sync telegram*
  The Master-Sync telegram is received simultaneously by all drives and serves the synchronization of all time related commands of the numeric control (NC) and drives.
- *Master-Data telegram*
  The Master-Data telegram is also received by all drives simultaneously. It contains the cyclical data and service data for all drives.
- *Configurable data field*
  The real-time data is completely transferred in every communication cycle in the so called configurable data field. The drives are sending their telegrams in sequence during assigned time windows. With the help of an ident no. system, the real-time data to send may be fixed during initialization. You may transfer numeric data like set point and effective values as well as bit lists with in-/output commands.

The exchange of service data needs a request from the master. Service data is transferred with a handshake procedure in 2, 4, 6 or 8Byte portions in the service data field "Info" and assembled again at the recipient.

FO as transfer medium

SERCOS uses a closed fiber optic ring (FO) as transfer medium. FO has a high immunity against electromagnetic interference. The ring structure needs the less number of FO and doesn't require T-connectors.

Using plastic FO, the length of each transfer section may be up to 50m, with glass fiber FO up to 250m. The maximum number of participants per ring is 254.

The exact number depends on the following factors:

- Required communication cycle time
- Operating data amount
- Data rate

Bus access procedure

The communication happens cyclically during the operation as a master slave communication. The cyclic time is defined during initialization and may range between 62µs and 65ms.

This cycle times are specified in a way that the required synchronization with fixed working cycle times in control and drives is met.

The communication master in a SERCOS ring is always the NC control.

Addressing

All participants at the bus must be identified by an unique address. Every SERCOS device has the option to fix the address.

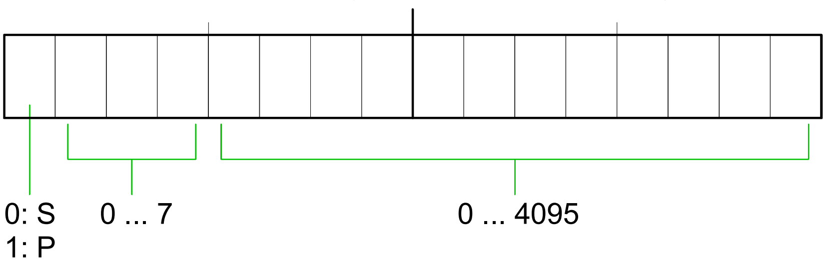
ID number for data exchange

The addressing of the data at the demand control data exchange and the definition of the real-time data happens with SERCOS via ident numbers. For the ID numbers, are value range of 2^{16} is fixed, divided into two areas:

1 ... 32767: for data (S-0 ... S-7)

32768 ... 65535: for parameters (P-0 ... P-7)

An identifier consists of 2Byte and has the following structure:



Example: Coding of S-2-1200

0	0	1	0	0	1	0	0	1	0	1	1	0	0	0	0

IM 253Sercos - SERCOS coupler - Structure

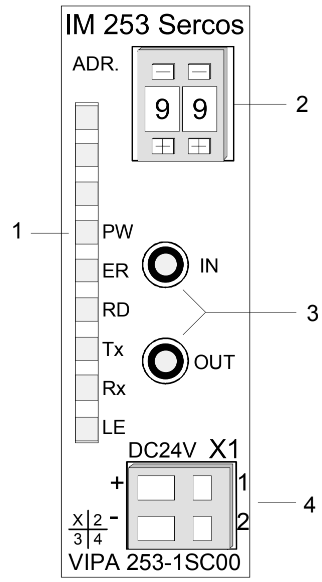
Properties

The SERCOS coupler IM 253SC supports the easy connection of decentral peripheral modules of the System 200V to SERCOS.

The SERCOS coupler is distinguished by the following properties:

- Fiber Optic (FO) Transmitters for use with 1mm Plastic Optical Fiber and 200µm Hard Clad Silica HCS®
- Support of all SERCOS baudrates (2, 4, 8, 16MBaud)
- Support of all System 200V modules from VIPA
- max. 32 peripheral modules, the number of analog modules is limited to 16 modules (please regard the assembly guidelines)
- max. 256Byte input and 256Byte output data
- Minimal SERCOS cycle: 1ms
- Address adjuster addresses (1 ... 89) and parameterization (90 ... 99)
- Integrated DC 24V power supply for voltage supply from coupler peripheral modules.
- LED status indicator

Front view 253-1SC00



[1] LED status indicator
[2] Address adjuster
[3] FO connection to SERCOS
[4] DC 24V connection supply voltage

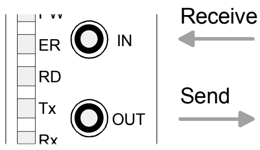
Components

LEDs

For the fast diagnosis of the recent module status there are 6 LEDs at the frontside.

Label	Color	Description
PW	yellow	Power LED: operating voltage on
ER	red	Error and the backplane bus or SERCOS
RD	green	Blinks at System OK and boot-up is in Phase 4. Is on when Phase 4 has been reached.
Tx	yellow	Is on at send activity via SERCOS
Rx	yellow	Is on at receive activity via SERCOS
LE	red	Error in the FO communication (line interruption res. hardware defect)

FO connection SERCOS

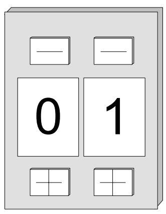


Via this jack you include the SERCOS coupler via FO transmitters into your SERCOS.

The connection to SERCOS takes place via 2 jacks. The direction of the 2 jacks is shown at the left side.

The jacks are for use with 1mm Plastic Optical Fiber and 200µm Hard Clad Silica HCS®.

Address selector



The address adjuster selector:

- the fixing of an unique SERCOS address (1 ... 89)
- the programming of the baudrate (90 ... 93)
- the adjustment of the light intensity (94 ... 97)
- the predefining of the time window calculation mode (98, 99)

Power supply

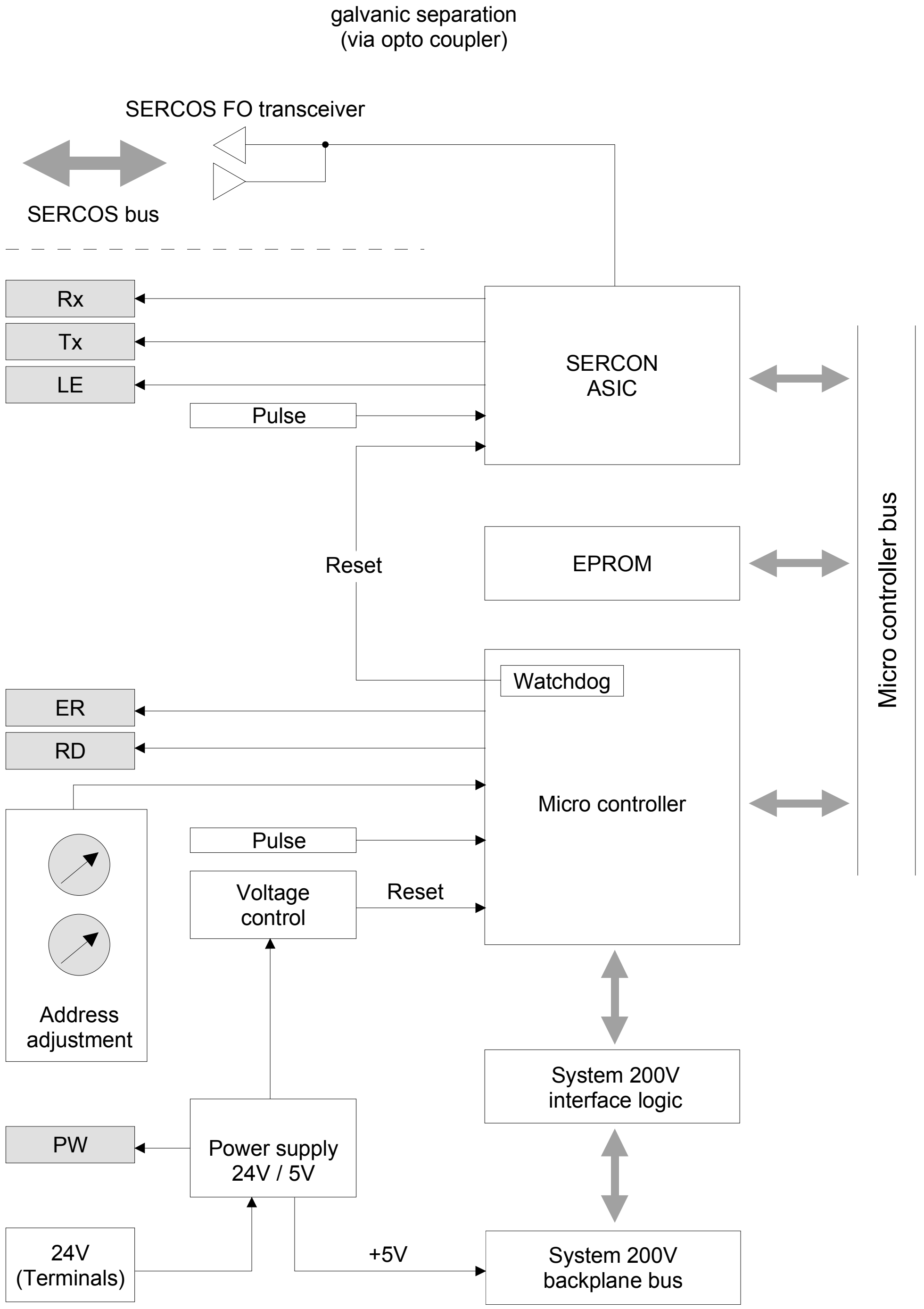
The SERCOS coupler has an integrated power supply, protected against inverse polarity and overcurrent.

This power supply also provides the connected peripheral modules with max. 3.5A via the back plane bus.

The connection of the supply voltage is at the frontside. The power supply has to be provided with DC 24V (20.4 ... 28.8V).

Block diagram

The following block diagram shows the principle of the hardware structure of the SERCOS coupler and the internal communication:



Basic parameterization via address adjuster

Overview

Via the address adjuster you may alter basic settings of the SERCOS coupler. Choose the according address code at the shut-down SERCOS coupler. At power on, this code is stored permanently in the SERCOS module.

The following basic settings may be altered in this way:

- Baudrate
- Light intensity
- Time window calculation

Note!

Please regard, that you may use the address adjuster only in off state. Otherwise malfunctions of the SERCOS couplers may occur!

Approach

Turn off the supply voltage of the SERCOS coupler.

Choose the according address code at the address adjuster.

Turn on the voltage supply.

→ The assigned parameter is permanently stored in the SERCOS coupler and this is shown via the green RD-LED.

Value range

00: reserved (may not be used)

01 ... 89: possible SERCOS station addresses

90 ... 99: VIPA additional functions for basic parameterization

Fix baudrate

All participants connected together at the bus are communicating at the same baudrate. You may fix the wanted baudrate via the address adjuster.

- Turn off the voltage supply.
- Choose the wanted baudrate at the address adjuster. Here means:

90: 2Mbaud
91: 4Mbaud
92: 8Mbaud
93: 16Mbaud

- Turn on the voltage supply.

→ The assigned baudrate is permanently stored in the SERCOS coupler and this is shown via the green RD-LED.

Fix light intensity

You may predefine the light intensity of the FO diode in 4 steps.

- Turn off the voltage supply.
- Choose the wanted light intensity at the address adjuster. You have following possibilities:

94: light intensity 0 (Minimum)
95: light intensity 1
96: light intensity 2
97: light intensity 3 (Maximum)

- Turn on the voltage supply.

→ The assigned light intensity is permanently stored in the SERCOS coupler and this is shown via the green RD-LED.

Time window calculation

Set here the operating mode for the time window calculation. The following 2 modes are possible:

98: Mode_All_Cyclic

The complete periphery is available in the cyclic SERCOS operation. Additionally you may also use the service channel. Depending on the number of modules you need SERCOS cycles of 2ms or more. The more periphery is connected, the higher you have to choose the SERCOS cycle time.

99: Mode_All_Service_Channel

In this mode, no periphery is available in the cyclic operation. For this you may operate the SERCOS ring with a cycle time of 1ms. Here you may address the peripheral modules exclusively via the service channel.

SERCOS Identifier

Overview

The read and write access to the System 200V under SERCOS takes place via ident numbers (short: IDN).

For the SERCOS coupler IM 253SC there are the following 3 ranges:

S-0-xxxx, S-1-xxxx: Standard IDNs, fixed by the SERCOS Interface Committee

S-2-xxxx, S-3-xxxx: IDNs from VIPA for transferring in- and output data.

P-0-xxxx: IDNs from VIPA for transferring parameter data

Standard IDNs S-0-xxxx, S-1-xxxx

The SERCOS coupler IM 253SC supports all Standard IDNs. More detailed information is to find in the SERCOS specification of the SERCOS Committee.

Depending on the operating mode the two Standard-ID lists are filled:

- Mode_All_Cyclic

  S-0-0187: points to all input identifier S-2-xxxx
  S-0-0188: points to all output identifier S-3-xxxx

- Mode_All_Service_Channel

  S-0-0187: List is empty
  S-0-0188: List is empty

VIPA specific IDNs S-2-xxxx, S-3-xxxx, P-0-xxxx

For the System 200V is are modular system, you may connect up to 32 modules in any sequence and assortment to the SERCOS coupler IM 253SC.

This builds dynamically very different configurations of in- and output channels. A module may occupy one or more of this channels. The maximum number of in-/output channels (I/O channels) is restricted to 256. The mapping of the modules and the I/O channels into the S-2- res. S-3-area and (at parameterizable modules additionally) into the P area happens automatically.

VIPA specific assignment of the IDN S-2-xxxx, S-3-xxxx and P-0-xxxx

The modules are scanned from the left to the right (Plug-in location 1 to 32) and separated after input and output the identifiers are created:

- Input channels are created in steps of 10s as S-2-ccc0 identifier. Here is ccc = 000 ... 255.

  Range: S-2-0000, S-2-0010, S-2-0020, ... S-2-2550

- Output channels are created in steps of 10s as S-3-ccc0 identifier. Here is ccc = 000 ... 255

  Range: S-3-0000, S-3-0010, S-3-0020, ... S-3-2550

- If you plug-in parameterizable modules, a P-0-ssxx identifier block is created for each module. Here is:

  Plug-in location: ss = 01 ... 32, Parameter: xx = 00 ... 17

  Example: P-0-0100 (module in plug-in location 1), P-0-0200 (module in plug-in location 2), ... P-0-3200 (module in plug-in location 32)

VIPA specific S-Identifier

For the S-Identifier there are the following information:

Name (consists of max. 32 characters)

Format: S.I.T_W.D

with

S = plug-in location (1..32)
I = module internal Byte offset at multi channel modules (0..15)
T = Type: (DIGITAL, ANALOG)
W = Data width: (BYTE, WORD, DOUBLE =1,2,4Byte)
D = Direction: (IN,OUT)

Example: Name: “1.0.DIGITAL_BYTE.IN” means:

The module in plug-in location 1 provides one Byte digital input data starting at the internal address 0 .

Attribute

in accordance to the SERCOS specification, the attribute fixes if the operating date is readable res. writeable. More detailed information is to find in the SERCOS specification of the SERCOS Committee.

Operating date

Here the in- res. output date with the according data width.

VIPA specific P-Identifier (always)

The SERCOS coupler always contains the two identifiers P-0-0000 and P-0-0001.

P-0-0000

Name: WRITE_PARAMETER

Attribute: Read/Write in Phase 0..3, Read Only in Phase 4

Operating date: 1 = Init adopt all parameter into EEPROM.
2 = Init delete all parameter in EEPROM.
0 = Return value OK
65535 (FFFFhex) = Return value ERROR

P-0-0001

Name: Estimated SERCOS cycle time

Attribute: Read Only

Unit: Micro seconds

Operating date: The chosen SERCOS cycle time must be higher than this value! (e.g. 1460 means that the estimated cycle time for the present module structure is 1.46ms and therefore you have to choose a SERCOS cycle of at least 2ms.)

18 VIPA specific P-Identifier (at parameterizable modules)

If you deploy parameterizable modules, for each parameterizable module a 18 P-0-ssxx identifier block is created dynamically. Here ss means plug-in location (1 ... 32) and xx for the parameter no. (0 ... 17).

In principle these additional P-0 identifiers have the following structure:

P-0-ss00

Name: ss.SLOT

Attribute: Read Only

Operating date: Indicates that there is a parameterizable module at the plug-in location

The length value and a description of the parameters to transfer are to find in the according chapters of the modules in this manual.

P-0-ss01

Name: ss.LENGTH

Attribute: Read/Write in Phase 0 ... 3, Read Only in Phase 4

Operating date: number of the now following parameter bytes for this module (value: 0 ... 15).

P-0-ss02

Name: ss.PARAMETER.0

Attribute: Read/Write in Phase 0..3, Read Only in Phase 4

Operating date: Parameter byte 0 (value: 0..255)

...

P-0-ss17

Name: ss.PARAMETER.15

Attribute: Read/Write in Phase 0..3, Read Only in Phase 4

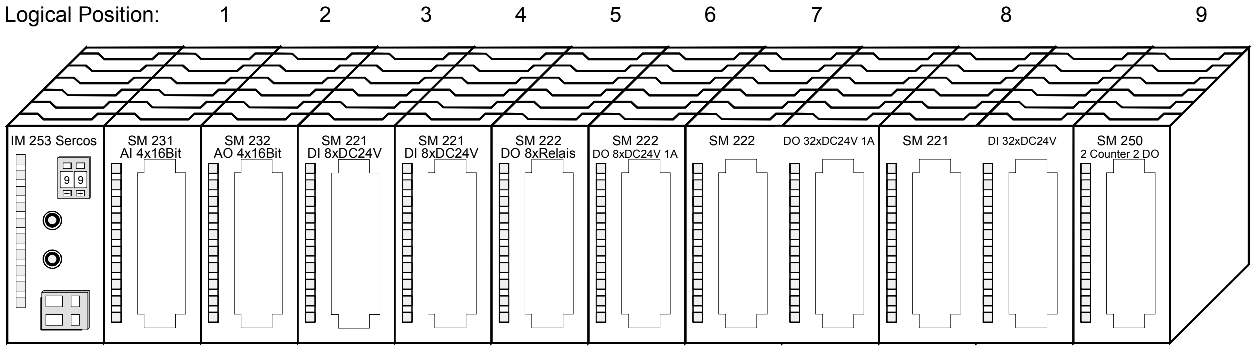
Operating date: Parameter byte 15 (value: 0..255)

Example for the automatic ID assignment

Structure

The following example describes how the automatic identifier assignment happens in the SERCOS coupler.

It has the following structure:



Logical position	Module	Input	Output	Parameter
1	VIPA 231-1BD52 (4 channel multi Analog Input)	ANALOG_WORD ANALOG_WORD ANALOG_WORD ANALOG_WORD		10 Byte
2	VIPA 232-1BD51 (4 channel multi Analog Output)		ANALOG_WORD ANALOG_WORD ANALOG_WORD ANALOG_WORD	6 Byte
3	VIPA 221-1BF00 (8bit digital Input)	DIGITAL_BYTE		-
4	VIPA 221-1BF00 (8bit digital Input)	DIGITAL_BYTE		-
5	VIPA 222-1HF00 (8bit digital Output, Relay)		DIGITAL_BYTE	-
6	VIPA 222-1BF00 (8bit digital Output, Transistor)		DIGITAL_BYTE	-
7	VIPA 222-2BL10 (32bit digital Output, Transistor)		DIGITAL_DOUBLE	-
8	VIPA 221-2BL10 (32bit digital Input)	DIGITAL_DOUBLE		-
9	VIPA 250-1BA00 (Counter Module with 2x32Bit Counter and control register)	DIGITAL_DOUBLE DIGITAL_DOUBLE DIGITAL_BYTE DIGITAL_BYTE	DIGITAL_DOUBLE DIGITAL_DOUBLE DIGITAL_BYTE DIGITAL_BYTE	2 Byte

Automatically created identifiers

For this structure, the following identifiers are created automatically:

S-2-Identifier (Input)

Identifier	Name	Comment
S-2-0000	1.0.ANALOG_WORD.IN	Module in position 1 Inside the module at Byte offset 0 An analog word Input
S-2-0010	1.2.ANALOG_WORD.IN	Module in position 1 Inside the module at Byte offset 2 An analog word Input
S-2-0020	1.4.ANALOG_WORD.IN	Module in position 1 Inside the module at Byte offset 4 An analog word Input
S-2-0030	1.6.ANALOG_WORD.IN	Module in position 1 Inside the module at Byte offset 6 An analog word Input
S-2-0040	3.0.DIGITAL_BYTE.IN	Module in position 3 Inside the module at Byte offset 0 An digital Byte Input
S-2-0050	4.0.DIGITAL_BYTE.IN	Module in position 4 Inside the module at Byte offset 0 An digital Byte Input
S-2-0060	8.0.DIGITAL_DOUBLE.IN	Module in position 8 Inside the module at Byte offset 0 An digital double word Input

continue ...

... continue

S-2-0070	9.0.DIGITAL_DOUBLE.IN	Module in position 9 Inside the module at Byte offset 0 An digital double word Input
S-2-0080	9.4.DIGITAL_DOUBLE.IN	Module in position 9 Inside the module at Byte offset 4 An digital double word Input
S-2-0090	9.8.DIGITAL_BYTE.IN	Module in position 9 Inside the module at Byte offset 8 An digital Byte Input
S-2-0100	9.9.DIGITAL_BYTE.IN	Module in position 9 Inside the module at Byte offset 9 An digital Byte Input

S-3-Identifier (Output)

S-3-0000	2.0.ANALOG_WORD.OUT	Module in position 2 Inside the module at Byte offset 0 An analog word Output
S-3-0010	2.2.ANALOG_WORD.OUT	Module in position 2 Inside the module at Byte offset 2 An analog word Output
S-3-0020	2.4.ANALOG_WORD.OUT	Module in position 2 Inside the module at Byte offset 4 An analog word Output
S-3-0030	2.6.ANALOG_WORD.OUT	Module in position 2 Inside the module at Byte offset 6 An analog word Output
S-3-0040	5.0.DIGITAL_BYTE.OUT	Module in position 5 Inside the module at Byte offset 0 A digital Byte Output

continue ...

... continue

S-3-0050	6.0.DIGITAL_BYTE.OUT	Module in position 6 Inside the module at Byte offset 0 A digital Byte Output
S-3-0060	7.0.DIGITAL_DOUBLE.OUT	Module in position 7 Inside the module at Byte offset 0 A digital double word Output
S-3-0070	9.0.DIGITAL_DOUBLE.OUT	Module in position 9 Inside the module at Byte offset 0 A digital double word Output
S-3-0080	9.4.DIGITAL_DOUBLE.OUT	Module in position 9 Inside the module at Byte offset 4 A digital double word Output
S-3-0090	9.8.DIGITAL_BYTE.OUT	Module in position 9 Inside the module at Byte offset 8 A digital Byte Output
S-3-0100	9.9.DIGITAL_BYTE.OUT	Module in position 9 Inside the module at Byte offset 9 A digital Byte Output

P-0-Identifier (Parameter) always present

P-0-0000	WRITE_PARAMETER	Set here the init for read/write all parameters: 1=Write, 2=Clear
P-0-0001	Estimated SERCOS cycle time	Value here: 1460 Micro seconds i.e. you may run this assembly with 2ms SERCOS cycle.

P-0-Identifier (Parameter) at parameterizable modules

P-0-0100	1.SLOT	At position 1 is a parameterizable module
P-0-0101	1.LENGTH	(operating date) Bytes shall be transferred to the module at position 1.
P-0-0102	1.PARAMETER.0	Parameter byte 0 for module at position 1
P-0-0103	1.PARAMETER.1	Parameter byte 1 for module at position 1
P-0-0104	1.PARAMETER.2	Parameter byte 2 for module at position 1
P-0-0105	1.PARAMETER.3	Parameter byte 3 for module at position 1
P-0-0106	1.PARAMETER.4	Parameter byte 4 for module at position 1
P-0-0107	1.PARAMETER.5	Parameter byte 5 for module at position 1
P-0-0108	1.PARAMETER.6	Parameter byte 6 for module at position 1
P-0-0109	1.PARAMETER.7	Parameter byte 7 for module at position 1
P-0-0110	1.PARAMETER.8	Parameter byte 8 for module at position 1
P-0-0111	1.PARAMETER.9	Parameter byte 9 for module at position 1
P-0-0112	1.PARAMETER.10	Parameter byte 10 for module at position 1
P-0-0113	1.PARAMETER.11	Parameter byte 11 for module at position 1
P-0-0114	1.PARAMETER.12	Parameter byte 12 for module at position 1
P-0-0115	1.PARAMETER.13	Parameter byte 13 for module at position 1
P-0-0116	1.PARAMETER.14	Parameter byte 14 for module at position 1
P-0-0117	1.PARAMETER.15	Parameter byte 15 for module at position 1
P-0-0200	2.SLOT	At position 2 is a parameterizable module
P-0-0201	2.LENGTH	(operating date) Bytes shall be transferred to the module at position 2.
P-0-0202	2.PARAMETER.0	Parameter byte 0 for module at position 2
P-0-0203	2.PARAMETER.1	Parameter byte 1 for module at position 2
P-0-0204	2.PARAMETER.2	Parameter byte 2 for module at position 2
P-0-0205	2.PARAMETER.3	Parameter byte 3 for module at position 2
P-0-0206	2.PARAMETER.4	Parameter byte 4 for module at position 2
P-0-0207	2.PARAMETER.5	Parameter byte 5 for module at position 2
P-0-0208	2.PARAMETER.6	Parameter byte 6 for module at position 2
P-0-0209	2.PARAMETER.7	Parameter byte 7 for module at position 2
P-0-0210	2.PARAMETER.8	Parameter byte 8 for module at position 2
P-0-0211	2.PARAMETER.9	Parameter byte 9 for module at position 2
P-0-0212	2.PARAMETER.10	Parameter byte 10 for module at position 2
P-0-0213	2.PARAMETER.11	Parameter byte 11 for module at position 2
P-0-0214	2.PARAMETER.12	Parameter byte 12 for module at position 2
P-0-0215	2.PARAMETER.13	Parameter byte 13 for module at position 2
P-0-0216	2.PARAMETER.14	Parameter byte 14 for module at position 2
P-0-0217	2.PARAMETER.15	Parameter byte 15 for module at position 2

continue ...

... continue

P-0-0900	9.SLOT	At position 9 is a parameterizable module
P-0-0901	9.LENGTH	(operating date) Bytes shall be transferred to the module at position 9.
P-0-0902	9.PARAMETER.0	Parameter byte 0 for module at position 9
P-0-0903	9.PARAMETER.1	Parameter byte 1 for module at position 9
P-0-0904	9.PARAMETER.2	Parameter byte 2 for module at position 9
P-0-0905	9.PARAMETER.3	Parameter byte 3 for module at position 9
P-0-0906	9.PARAMETER.4	Parameter byte 4 for module at position 9
P-0-0907	9.PARAMETER.5	Parameter byte 5 for module at position 9
P-0-0908	9.PARAMETER.6	Parameter byte 6 for module at position 9
P-0-0909	9.PARAMETER.7	Parameter byte 7 for module at position 9
P-0-0910	9.PARAMETER.8	Parameter byte 8 for module at position 9
P-0-0911	9.PARAMETER.9	Parameter byte 9 for module at position 9
P-0-0912	9.PARAMETER.10	Parameter byte 10 for module at position 9
P-0-0913	9.PARAMETER.11	Parameter byte 11 for module at position 9
P-0-0914	9.PARAMETER.12	Parameter byte 12 for module at position 9
P-0-0915	9.PARAMETER.13	Parameter byte 13 for module at position 9
P-0-0916	9.PARAMETER.14	Parameter byte 14 for module at position 9
P-0-0917	9.PARAMETER.15	Parameter byte 15 for module at position 9

Example parameterization

For example, the following values shall be set:

AI 4x16Bit (231-1BD52) at position 1

Length: 10Byte

Parameter:

Byte	Description	Set property	Handling value
0	Diagnostic alarm Byte:	deactivated	00h = 0dez
1	reserved	00h	00h = 0dez
2	Function no. channel 0	Voltage ±10V in the S7 format from Siemens	28h = 40dez
3	Function no. channel 1	Voltage ±10V in the S7 format from Siemens	28h = 40dez
4	Function no. channel 2	Current 4...20mA in S7 format from Siemens	2Dh = 45dez
5	Function no. channel 3	Current 4...20mA in S7 format from Siemens	2Dh =45dez
6	Option Byte channel 0	default	00h = 0dez
7	Option Byte channel 1	default	00h = 0dez
8	Option Byte channel 2	default	00h = 0dez
9	Option Byte channel 3	default	00h = 0dez

Herefore the table has the following entries:

P-0-0100	1.SLOT	At position 1 is a parameterizable module
P-0-0101	1.LENGTH	10dez
P-0-0102	1.PARAMETER.0	0dez
P-0-0103	1.PARAMETER.1	0dez
P-0-0104	1.PARAMETER.2	40dez
P-0-0105	1.PARAMETER.3	40dez
P-0-0106	1.PARAMETER.4	45dez
P-0-0107	1.PARAMETER.5	45dez
P-0-0108	1.PARAMETER.6	0dez
P-0-0109	1.PARAMETER.7	0dez
P-0-0110	1.PARAMETER.8	0dez
P-0-0111	1.PARAMETER.9	0dez
P-0-0112 ... P-0-0117	1.PARAMETER.10 ... 1.PARAMETER.15	are created but not used

Set the value in **P-0-0000** to 1 and the parameters are stored in the EEPROM of the SERCOS coupler.

At successful transfer, you get the return value 0 and at the analog input module the LEDs F2 and F3 for wirebreak recognition are illuminated due to the current measuring range.

AO 4x16Bit (232-1BD51) at position 2

Length: 6Byte

Parameter:

Byte	Description	Set property	Handling value
0	Diagnostic alarm Byte:	deactivated	00h = 0dez
1	reserved	00h	00h = 0dez
2	Function no. channel 0	Voltage ±10V in the S7 format from Siemens	09h = 9dez
3	Function no. channel 1	Voltage ±10V in the S7 format from Siemens	09h = 9dez
4	Function no. channel 2	Current 4...20mA in S7 format from Siemens	0Ch = 12dez
5	Function no. channel 3	Current 4...20mA in S7 format from Siemens	0Ch =12dez

Herefore the table has the following entries:

P-0-0200	2.SLOT	At position 2 is a parameterizable module
P-0-0201	2.LENGTH	6dez
P-0-0202	2.PARAMETER.0	0dez
P-0-0203	2.PARAMETER.1	0dez
P-0-0204	2.PARAMETER.2	9dez
P-0-0205	2.PARAMETER.3	9dez
P-0-0206	2.PARAMETER.4	12dez
P-0-0207	2.PARAMETER.5	12dez
P-0-0208 ... P-0-0217	2.PARAMETER.6 ... 2.PARAMETER.15	are created but not used

Set the value in **P-0-0000** to 1 and the parameters are stored in the EEPROM of the SERCOS coupler.

At successful transfer, you get the return value 0 and at the analog output module the LED for wirebreak recognition are illuminated due to the current measuring range.

SM 250 2 Counter 2 DO (250-1BA00) at position 2

Length: 2Byte

Parameter:

Byte	Description	Set property	Handling value
0	Mode Counter 0	Frequency measurement	16dez
1	Mode Counter 1		16dez

Herefore the table has the following entries::

P-0-0900	9.SLOT	At position 9 is a parameterizable module
P-0-0901	9.LENGTH	2dez
P-0-0902	9.PARAMETER.0	16dez
P-0-0903	9.PARAMETER.1	16dez
P-0-0904 ... P-0-0917	9.PARAMETER.2 ... 9.PARAMETER.15	are created but not used

Set the value in **P-0-0000** to 1 and the parameters are stored in the EEPROM of the SERCOS coupler.

At successful transfer, you get the return value 0.

Technical Data

SERCOS coupler
IM 253SC

Electrical Data	VIPA 253-1SC00
Voltage supply Current consumption Output current backplane bus Potential separ. to the backplane bus	DC 24V (20.4 ... 28.8V) via front by ext. pow. supply Bus coupler: 50mA incl. supply of the peripheral modules: max. 3.5A (5V) max. 3.5A 500V eff.
Function specific data	
Status indicator Physical connection SERCOS Network topology Transfer medium Transfer rate Number of participants	via LED at the frontside FO jacks Ring Fiber optic transmitter, for use with 1mm Plastic Optical Fiber and 200µm Hard Clad Silica HCS® 2, 4, 8, 16MBaud max. 89
Combination with peripheral modules	
Module number Inputs Outputs	max. 32 max. 256Byte max. 256Byte
Dimensions and Weight	
Dimensions (WxHxD) Weight	25.4x76x78mm 75g

Chapter 8 Ethernet coupler

Outline

Content of this chapter is the description of the Ethernet coupler IM 253NET from VIPA. It contains all information for installation and commissioning of the Ethernet coupler.

The chapter starts with the basics. Here the basic expressions of the Ethernet communication are explained together with the guidelines for building up a network.

Another part describes the hardware components and the access to the Ethernet coupler.

The chapter ends with the used protocols, a sample for socket programming and the technical data.

The following text contains:

- System overview
- Basics of the Ethernet communication
- Structure of the Ethernet coupler
- Principles of the automatic address allocation
- (Online-)access to the Ethernet coupler
- Programming sample
- Technical data

Content

Topic **Page**

Chapter 8 Ethernet coupler 8-1
System overview 8-2
Basics 8-3
Planning a network 8-7
IM 253NET - Ethernet coupler - Structure 8-9
Access to the Ethernet coupler 8-11
Principle of the automatic address allocation 8-14
Project engineering under WinNCS 8-15
Diagnosis and test via Internet Browser 8-16
ModbusTCP 8-20
Modbus function codes 8-21
Siemens S5 Header Protocol 8-26
Principle of Alarm handling 8-28
Programming sample 8-32
Technical data 8-33

System overview

Typical fieldbus systems are divided into master and slave systems.

Master system are CPs, coupled to a CPU, allowing remote programming res. visualization of the according CPU as well as the data transfer between several TCP/IP participants.

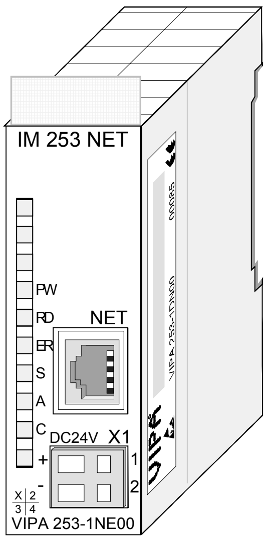
Slave systems on the other hand are "data collectors" that deliver the I/O data of the connected modules to the requesting master.

The Ethernet coupler described in this chapter is a slave system.

For the communication happens via TCP/IP, the slave system is referred to as server and a master as client.

The Ethernet coupler from VIPA allows you to connect up to 32 modules of your System 200V periphery via Ethernet. With each protocol up to 8 clients may communicate simultaneously with the Ethernet coupler.

At this time, VIPA offers the following Ethernet coupler:



Ordering data Ethernet coupler

Type	Order number	Description
IM 253NET	VIPA 253-1NE00	Ethernet coupler

Basics

Ethernet

Originally, Ethernet has been developed from DEC, Intel and Xerox (as DIX standard) for the data transfer between office devices. Nowadays it normally means the specification *IEEE 802.3 CSMA/CD*, first published in 1985. Due to the worldwide deployment and the high lot sizes, this technology is commonly available and reasonably priced. This allows the easy link-up to existing networks.

Ethernet transports Ethernet packages from one sender to one ore more receivers. This transfer happens without acknowledgement and without repetition of lost packages. For a secure data transfer, protocols like TCP/IP are used that are accompanying Ethernet.

Twisted Pair

In the early days of networking the Triaxial- (yellow cable) or thin Ethernet cable (Cheapernet) was used as communication medium. This has been superseded by the twisted pair network cable due to its immunity to interference. The IM 253NET Ethernet coupler has a twisted-pair connector.

Where the coaxial Ethernet networks are based on a bus topology the twisted pair network is based on a point-to-point scheme.

The network that may be established by means of this cable has a star topology. Every station is connected to the hub/switch by means of a separate cable. The hub/switch provides the interface to the Ethernet.

Hub

The hub is the central element that is required to implement a twisted pair Ethernet network. It is the job of the hub to regenerate and to amplify the signals in both directions. At the same time it must have the facility to detect and process segment wide collisions and to relay this information. The hub is not accessible by means of a separate network address since it is not visible to the stations on the network. A hub has provisions to interface with Ethernet or another hub.

Switch

A switch also is a central element for implementing a twisted pair Ethernet network. Several station res. hubs are connected together via a switch. These then may communicate with each other via the switch without causing network load. An intelligent hardware analyses the incoming telegrams for every port of the switch and passes them collision free on to the destination stations at the switch. A switch optimizes the band width of every connected segment of a network. Switches allow changing exclusive connections between the connected segment of a network.

Access control

Ethernet supports the principle of random bus access: every station on the network accesses the bus independently as and when required. These accesses are coordinated by a CSMA/CD (Carrier Sense Multiple Access/Collision Detection) scheme: every station "listens" on the bus cable and receives communication messages that are addressed to it.

Stations only initiate a transmission when the line is unoccupied. In the event that two participants should start transmitting simultaneously, they will detect this and stop transmitting to restart after a random delay time has expired.

Communication

The Ethernet coupler is connected with the modules via the backplane bus. It collects their data and places this as "server" (slave) at the disposal of the superordinated "client" (master system).

The communication happens via TCP/IP with leading ModbusTCP or Siemens S5 header protocol.

Vice versa, the Ethernet coupler receives the data, addressed to it by IP address and port, and transfers it to its output periphery. For project engineering, VIPA offers the configuration tool WinNCS that allows you to configure the Ethernet coupler online.

For test and diagnostic purposes the Ethernet slave provides a web server that allows the read and write access to the I/O periphery as well as the parameterization of the modules.

Overview Protocols

Protocols define rules or standards that enables different computers to establish communication connections and exchange data as error free as possible.

The so called ISO/OSI layer model is generally accepted for the standardization of computer communication. The layer model is based upon seven layers with guidelines for the deployment of hard- and software.

Layer	Function	Protocol
Layer 7	Application Layer (Application)	Siemens S5 Header, ModbusTCP
Layer 6	Presentation Layer (Presentation)	
Layer 5	Session Layer (Session)	
Layer 4	Transport Layer (Transport)	TCP
Layer 3	Network Layer (Network)	IP
Layer 2	Data Link Layer (Security)	
Layer 1	Physical Layer (Bit transfer)	

Telegram structure

Layer 2	Layer 3	Layer 4	Layer 7	
MAC/DLL	IP	TCP	API	...
14 Byte	20 Byte	20 Byte	Length depends on protocol	

MAC/DLL

While the Ethernet physics covers with its normed signal levels Layer 1, MAC/DLL covers the conditions of the security layer (Layer 2). With MAC (**M**edium **A**ccess **C**ontrol) / DLL (**D**ata **L**ink **L**ayer) the communication happens at the lowest Ethernet level using MAC addresses. Every Ethernet communication participant has a MAC address that must be unique at the network.

The deployment of MAC addresses specifies source and destination unambiguously.

IP

The Internet Protocol covers the network layer (layer 3) of the ISO/OSI layer model.

The main purpose of IP is to send data packages from one station to another, passing several other stations. This data packages are referred to as datagrams. The IP does neither serve the according sequence nor the deliverance at the receiver.

For the unambiguous distinction between sender and receiver, 32Bit addresses are used (IP addresses) that are normally written in four octets of each 8Bit, e.g. 172.16.192.11. One octet may represent numbers between 0 and 255.

A part of the address specifies the network, the rest identifies the single stations in the network. The proportions of network part and station part is floating and depends on the network size.

TCP

The TCP (**T**ransmission **C**ontrol **P**rotocol) puts directly upon the IP and covers therefore the transport layer (layer 4) of the ISO/OSI layer model. TCP is a connection orientated end-to-end protocol and serves the logical connection between two partners.

TCP ensures the sequential correct and reliable data transfer.

Every datagram is preceded by a header of at least 20 octets that contains, among others, the serial number for the according sequence. This causes that within a network, the single datagrams may reach their destination on different ways.

API

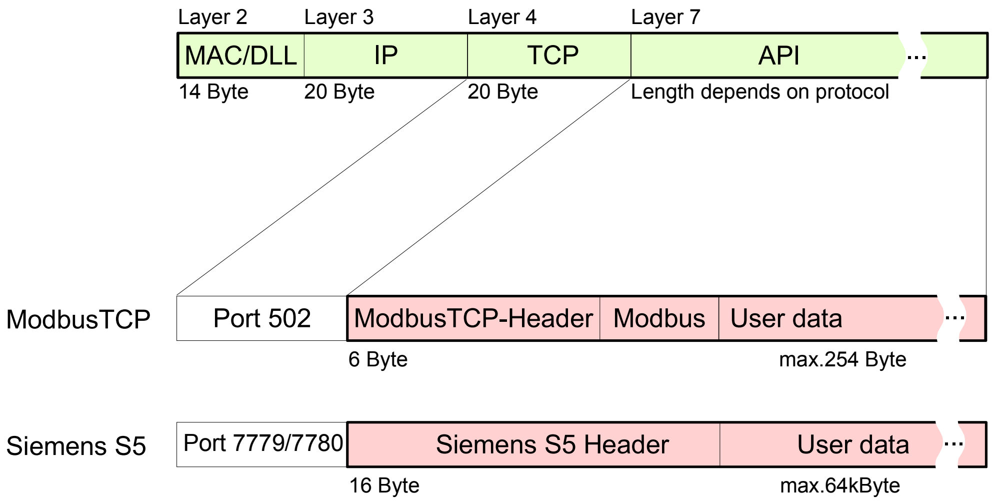
API means **A**pplication **P**rogramming **I**nterface. API covers the conditions of the Application Layer (Layer 7).

Here, the header and user data of the according protocols are stored.

The Ethernet coupler IM 253NET from VIPA uses the following protocols, described further below:

- ModbusTCP
- Siemens S5 Header

API structure



ModbusTCP

ModbusTCP is a Modbus-RTU protocol, put upon TCP/IP.

The Modbus protocol is a communication protocol supporting a hierarchic structure with one master and several slaves. ModbusTCP extends Modbus to a client server communication where several client may access a server.

For the addressing happens by means of the IP addresses, the address integrated in the Modbus telegram irrelevant. Furthermore, the check sum is not required because the sequence insurance happens via TCP/IP.

After the request of a client, this awaits the answer of the server for a configurable time.

ModbusTCP exclusively uses the RTU format.

Every Byte is transferred as one sign. This enables a higher data pass-through than the Modbus-ASCII format. The RTU time supervision is omitted for the header contains the size of the telegram length to be received.

Data that are transferred via ModbusTCP may contain bit and word information. At bit chains, the highest bit is send first, i.e. in a word it is at the most left position. At words, the highest Byte is send first.

The access to a Modbus slave happens via function codes that are described in detail in this chapter further below.

Siemens S5 Header

The Siemens S5 Header protocol serves the data transfer between PLC systems. Deploying the organization format (short ORG) integrated in the Siemens S5 Header protocol, a short description of a data source res. data destination in PLC environment is possible.

The possible ORG formats are corresponding to Siemens.

Planning a network

General

The main characteristic of a bus structure is the existence of a single physical transfer line. As physical transfer mediums are used:

- one or more electrical cables (drilled cable)
- coaxial cable (Triaxial cable)
- fiber optic transmitter.

To enable the communication between the single stations, rules and instructions have to be arranged and kept.

The appointments cover the form of the data protocol, the access procedure to the bus and more basics for communication. The Ethernet coupler IM 253NET from VIPA has been developed upon the ISO standards and norms.

Standards and norms

The following standards and norms about network technologies have been fixed by international and national committees:

ANSI
American National Standards Institute
The ANSI X3T9.5 standard currently defines the provisions for high speed LAN's (100 MB/s) based on fiber optic technology.
(FDDI) Fiber Distributed Data Interface.

CCITT
Committee Consultative Internationale de Telephone et Telegraph.
Amongst others, this advisory committee has produced the provisions for the connection of industrial networks (MAP) to office networks (TOP) on Wide Area networks (WAN).

ECMA
European Computer Manufacturers Association.
Has produced various MAP and TOP standards.

EIA
Electrical Industries Association (USA)
This committee has issued standard definitions like RS-232 (V.24) and RS-511.

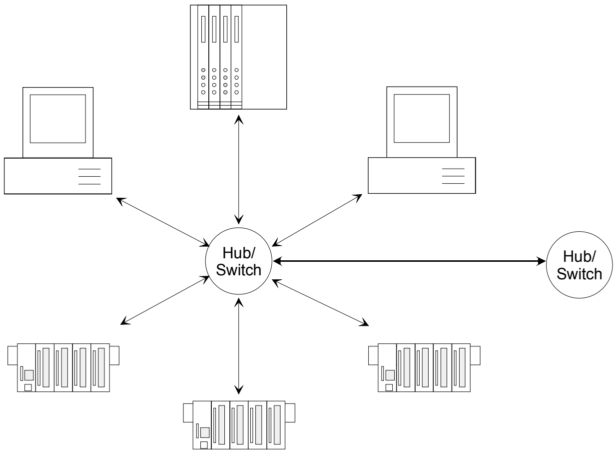
IEC
International Electrotechnical Commission.
Defines certain special standards, e.g. for the Field bus.

ISO
International Organization for Standardization.
This association of national standards organizations developed the OSI-model (ISO/TC97/SC16). It provides the framework for the standardization of data communications. ISO standards are included in different national standards like for example UL and DIN.

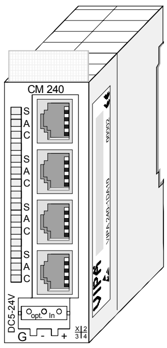
IEEE
Institute of Electrical and Electronic Engineers (USA).
The project-group 802 determines LAN-standards for transfer rates of 1 to 20 MB/s. IEEE standards often form the basis for ISO-standards, e.g. IEEE 802.3 = ISO 8802.3.

Overview components

A twisted pair network can only be constructed with a star topology.



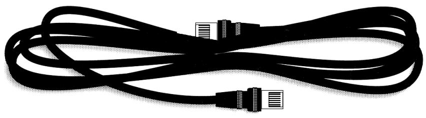
Mini-Switch CM 240



Twisted Pair Cable

A Twisted Pair cable is a cable with four cores drilled in pairs.

The single cores have a diameter of 0.4 to 0.6mm.



Restrictions

This is a summary of the restrictions and rules referring to Twisted Pair:

- Maximum number of coupler elements per segment 2
- Maximum length of a segment 100m

Analyzing the requirements

- What is the size of the area that must be served by the network?
- How many network segments provide the best solution for the physical (space, interference related) conditions encountered on site?
- How many network stations (SPS, IPC, PC, transceiver, bridges if required) must be connected to the cable?
- What is the distance between the different stations on the network?
- What is the expected ”growth rate” and the expected number of connections that must be catered for by the system?
- What is the expected data amount (Band width, accesses/sec.)?

Drawing a network diagram

- Draw a diagram of the network. Identify every hardware item (i.e. station cable, Hub, switch). Observe the applicable rules and restrictions.
- Measure the distance between all components to ensure that the maximum length is not exceeded.

IM 253NET - Ethernet coupler - Structure

Properties

- Ethernet coupler with ModbusTCP and Siemens S5 Header protocol
- max. 32 modules connectable with max. 256Byte input and 256Byte output data
- I/O access with both protocols via PC software like e.g. the OPC server from VIPA
- Online project engineering under WinNCS from VIPA with automatic coupler search and parameterization of modules in plain text. Here you may also fix IP address, subnet mask and coupler name and execute a firmware update.
- Extensive Alarm handling
- Integrated web server for test and diagnosis
- RJ45 jack 100BaseTX, 10BaseT
- Automatic polarity and speed recognition (auto negotiation)
- Automatic recognition of parallel or crossed cable (auto crossover)
- Network LEDs for link/activity, speed and collision
- Status-LEDs for Ready and Error

Delivery default

IP address: 10.0.0.1

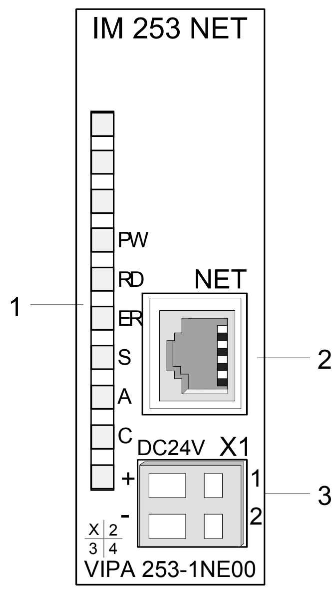
Password for alteration access via WinNCS: `00000000`

Attention!

For every Ethernet coupler is delivered with the IP address 10.0.0.1, you must not connect more than one new Ethernet coupler at one time.

First commissioning: Connect the new coupler with the network, assign a TCP/IP address. Now you may connect the next new coupler...

Front view IM 253NET



[1] LED Status monitor
[2] RJ45 jack for Twisted Pair
[3] DC 24V voltage supply

Components

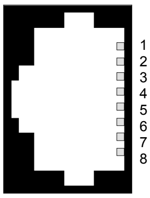
LEDs

The Ethernet coupler has different LEDs for diagnosis and monitoring the operational state. The usage and meaning of the colors are described in the following table.

Label	Color	Description
PW	Green	Power: DC 24V voltage supply is present
RD	Green	Ready: The Ethernet coupler has booted. I/O periphery, connected to the backplane bus can be accessed.
ER	Red	Error: Shows an error like e.g. module failure or parameterization error (Details: see coupler web site)
S	Green	Speed: on: 100MBit, off: 10Mbit
A	Green	Activity: on: physically connected off: no physical connection blinking: shows bus activity
C	Green	Collision: on: full duplex operation active off: half duplex operation active blinking: collision detected

RJ45 Ethernet connection

The RJ45 jack is the Twisted-Pair connection to Ethernet. The jack has the following pin assignment:



8pin RJ45 jack:

Pin	Signal
1	Transmit +
2	Transmit -
3	Receive +
4	-
5	-
6	Receive -
7	-
8	-

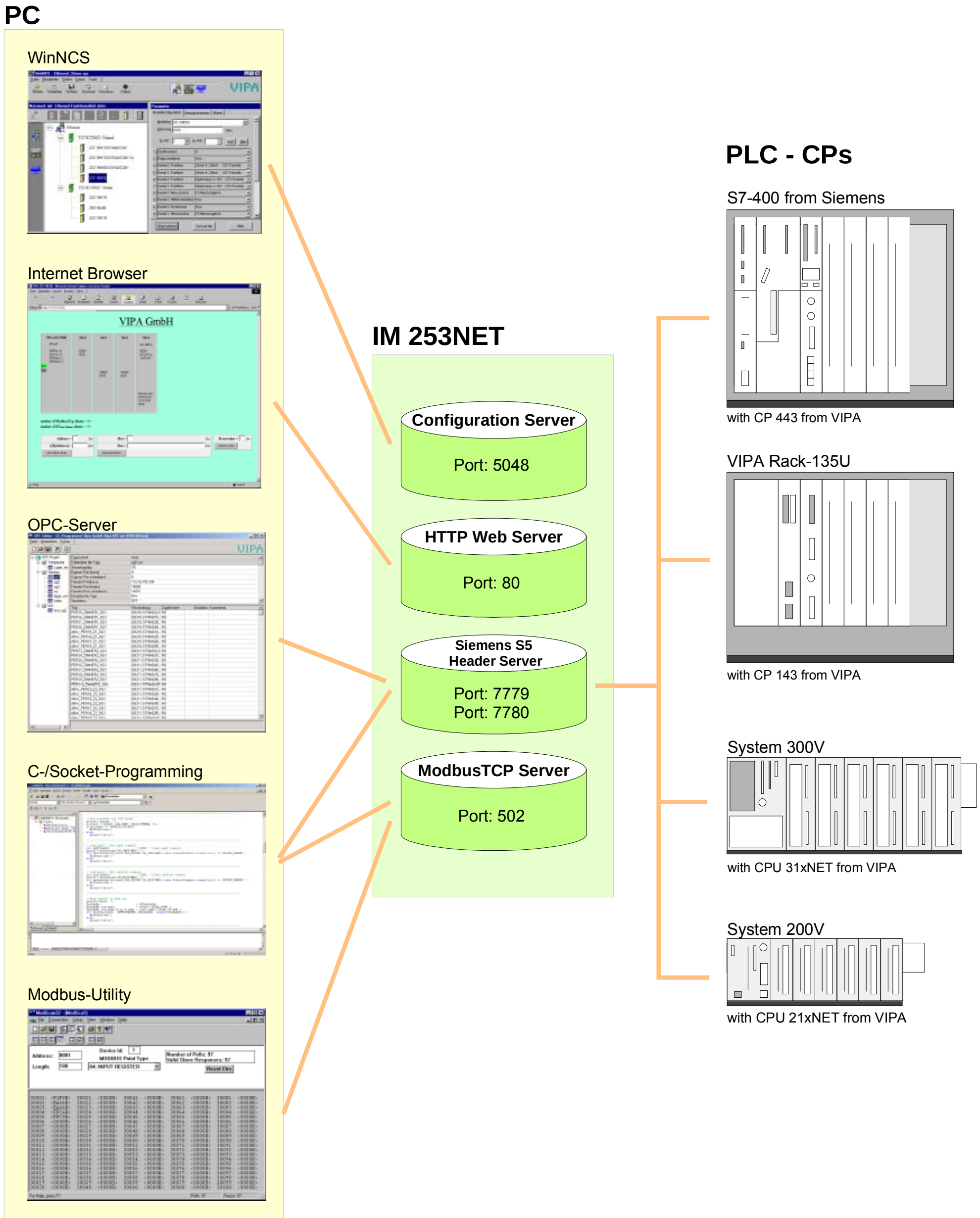
Power supply

The Ethernet coupler comes with an integrated power supply. The power supply has to be supplied with DC 24V (20.4 ... 28.8V) via the front. By means of the supply voltage, the bus coupler electronic is supplied as well as the connected modules via backplane bus. Please regard that the integrated power supply may supply the backplane bus with max. 3.5A. The power supply is protected against inverse polarity and overcurrent, Ethernet and backplane bus are galvanically isolated.

Access to the Ethernet coupler

Overview

The following illustration shows the Ethernet coupler IM 253NET access possibilities.



Access from PC

WinNCS for project engineering

The access happens via Port 5048 on the configuration server.

The configuration server calculates the number of plugged modules, their address and parameter ranges and puts the information under its IP address at the disposal of WinNCS.

WinNCS searches all couplers of the network via broadcast (slaves). The network to search is here until the gateway.

The collected data is used by WinNCS to model a symbolic network and is monitored in the network window.

Now you may assign real module types to the symbolic network and parameterize them.

Now you can assign an IP address to the Ethernet coupler online and update the firmware.

In WinNCS you also define the http web server properties of the Ethernet coupler.

All changing accesses are password protected. The password is requested once per session and slave.

In delivery state the password is `00000000`

Note!

Before you may access the Ethernet slave via internet browser, you have to assign an IP address according to your network. This may happen online via WinNCS.

Internet Browser for diagnosis and test

The access is via Port 80 at the HTTP web server.

The http server transfers a dynamically built web site that shows the recent configuration of the Ethernet coupler.

Besides of the firmware version and RDY/ERR-LED state, the I/O states and the parameters of the modules are shown.

The website also gives you the opportunity to send your alterations online, like accessing module outputs, change the parameters and initialize a re-boot of the Ethernet coupler.

OPC server for data transfer between coupler and PC

The access happens via the ports 7779 and 7780 on the Siemens S5 Header Server. Via these ports, fetch and write accesses via the VIPA OPC server are enabled.

The VIPA OPC server is a comfortable tool for visualization and data transfer.

C-/Socket programming for data transfer between coupler and PC

At ModbusTCP, the access is via port 502 at the ModbusTCP server and at Siemens S5 header via the ports 7779 and 7780 on the Siemens S5 Header Server.

This possibility of data transfer is for C program developers who want to create an open interface by means of socket programming.

Via simple C programs it is possible to transfer data between PC and Ethernet coupler. Depending on the program, the data is transferred via ModbusTCP or via Siemens S5 Header.

More detailed information about programming with sample sources is to find further below in this chapter.

Modbus utility

The access is via port 502 at the ModbusTCP Server. Modbus utility means all tools and programs that have a ModbusTCP interface.

For example, you may find the demo tool "ModbusScan32" from WinTech for download under www.win-tech.com.

Access from SPS res. CP

Data transfer between coupler and CP via Siemens S5 Header

The access happens via the ports 7779 and 7780 on the Siemens S5 Header Server. Via this ports, the VIPA CP, OPC server or other devices have fetch and write access.

For the communication, you need a PLC program in the CPU that serves the in-/output areas of the CP. Herefore, you have to configure fetch/write connections at the CP.

Principle of the automatic address allocation

Automatic addressing

To individually call the connected peripheral modules, certain addresses in the Ethernet coupler have to be assigned to them. For input and output area, the Ethernet coupler has an address range of each 256Byte.

The address allocation (also called Mapping) happens automatically and may not be influenced. The mapping may be seen via the website of the coupler.

After the 256Byte wide I/O image there follows the "alarm information image" with a size of 520Byte.

Rules

At boot-up, the Ethernet coupler assigns automatically addresses for its in-/output periphery following this rules:

- All modules are mapped from left (Ethernet coupler) to right in ascending sequence starting with address 0.
- It is separated between in- and output area (if a module has in- and output data, these are stored at different addresses).
- There is no separation between digital and analog data. The Ethernet coupler creates cohere areas for in- and output data.

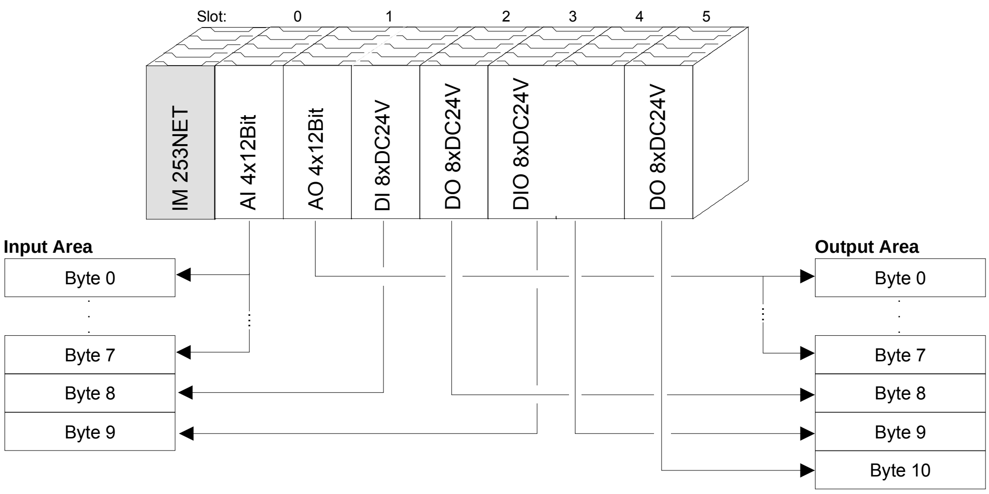
Note!

A description of the in- and output areas that are occupied by a module is to find in the concerning module description.

Please regard that modules that are occupying more than 1Byte like e.g. analog modules, are stored starting with an even address. Otherwise ModbusTCP has problems with word accesses.

Sample for the automatic address allocation

The following picture illustrates the automatic address allocation:



Project engineering under WinNCS

Preconditions

The project engineering happens via WinNCS starting with V3.09. For project engineering , the following preconditions should be met:

- Recent VIPA_ETH200V.GSD is stored in WinNCS/GSD/Englisch.

  For project engineering of the System 200V modules in WinNCS you receive the features of the VIPA components with a GSD-file.

  The GSD-file for the IM 253NET Ethernet coupler from VIPA is: VIPA_ETH200V.GSD

  Copy this GSD-file into WinNCS/GSD/Englisch.

  The latest version is to find under ftp.vipa.de/support.
- For online project engineering, the IM 253NET should be assembled with the according modules, connected to the Ethernet and supplied with voltage.

Attention!

For every Ethernet slave is delivered with the IP address 10.0.0.1, you must not install more than one new Ethernet slave at a time!

Approach online project engineering

- Start WinNCS and create a new "Ethernet" project via **File** > *Create/Open project*.
  → A parameter windows for online search of "Slaves" and "Stations" opens. [Slaves] lists all Ethernet coupler and [Stations] all CPs.
- Click at [Slaves]
  → All Ethernet coupler are searched and listed with IP address and where applicable with label.
- Via double-click at a listed slave, this is overtaken into the network window and listed with the concerning I/O periphery.
  → If there is no parameterization yet, the modules are listed as symbol (without label).
- Now you assign the according module type to the listed module symbol in the parameter window and adjust the parameters when needed. The address range that is occupied by the module in the TCP data stream is automatically preset by the Ethernet coupler.
- As soon as you click at [apply], you have to type the password. The password request happens once per session and coupler. In delivery state, the password is `00000000.` With correct password, the data is transferred online to the Ethernet coupler. Repeat this for all listed modules.
- Save your project.

Diagnosis and test via Internet Browser

Addressing

Type the configured IP address of your Ethernet coupler into your Internet Browser. Now you have access to a dynamically built-up website of the HTTP server.

Please regard that the website always contains the information of the last update.

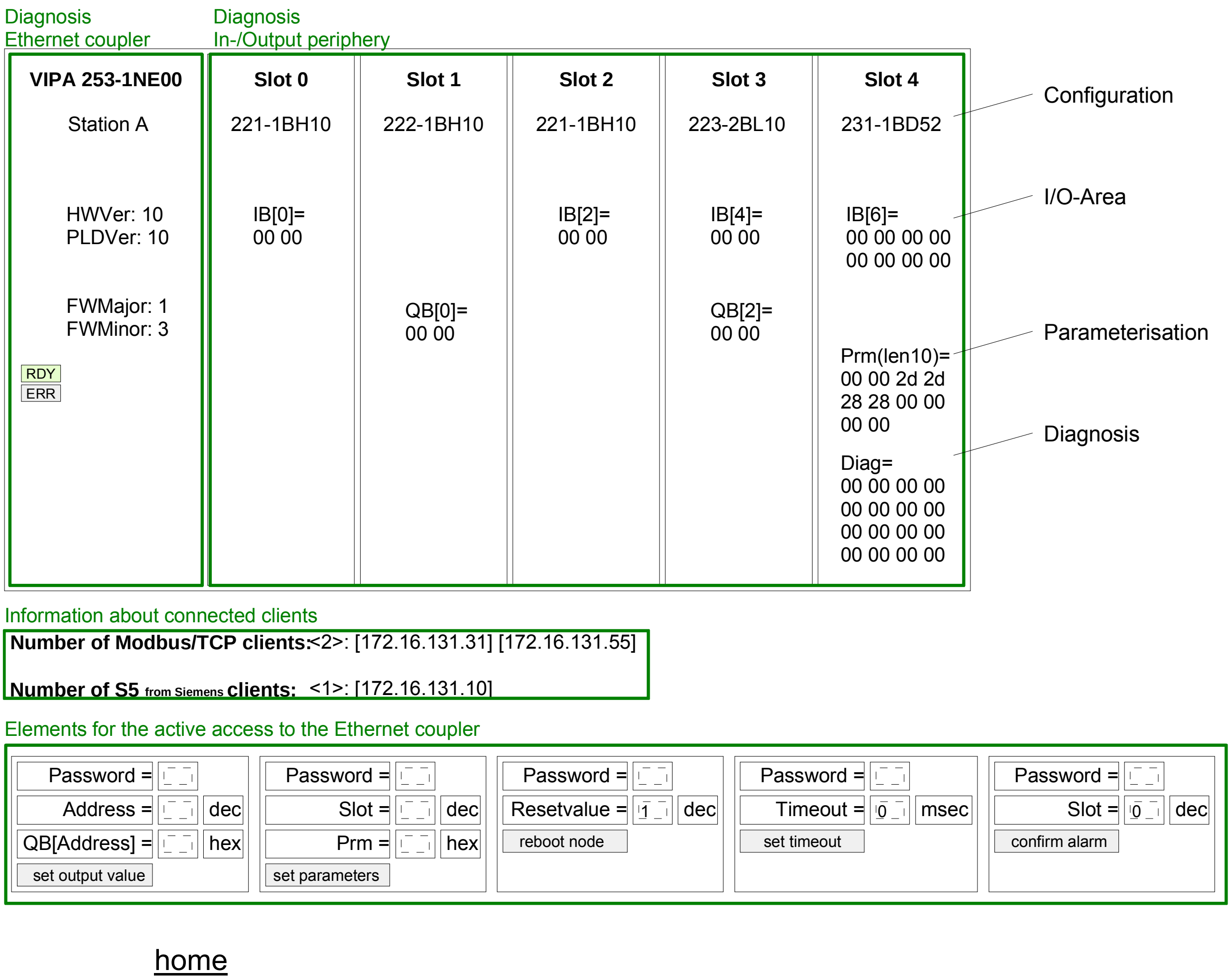
For an update, click at home in the lower left corner of the website.

Structure of Website

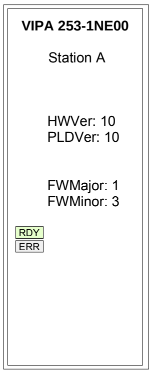
The website is dynamically built-up and depends on the number of the modules connected to the Ethernet coupler. The access rights to this website are in WinNCS freely configurable.

The following elements are to find on the website:

- Diagnosis Ethernet coupler
- Parameterization and diagnosis data in-/output periphery
- Information about connected clients
- Elements for active access to the Ethernet coupler



Diagnosis Ethernet coupler



This area shows all information about the Ethernet coupler like symbolic name, version and status monitors of the LEDs.

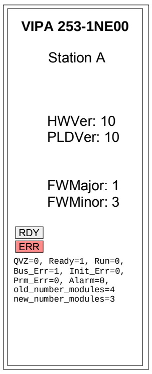
Symbolic name: Via WinNCS you may assign a symbolic name to the Ethernet coupler besides the IP address.

HWVer: This is the hardware version (electronics). The HW release (only number before comma) is also at the front side of the module.

PLDVer: The PLD (**P**rogrammable **L**ogic **D**evice) is a programmable logic block for control of the communication between backplane bus and processor.

FWMajor, FWMinor: The firmware version is divided into *FWMajor* (main version) and *FWMinor* (lower version). A lower version contains small alterations. When basic alterations are made, the main version number is increased.

Status monitor



RDY, ERR: Status monitor of the LEDs RD and ER

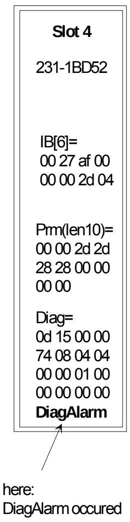
rdy (small letter): LED is blinking / RDY (capital letter): LED is on

As long as the Ethernet coupler communicates error free, the status monitor remains like shown above. In case of an error, e.g. the following message is displayed below ERR:

```
QVZ=0 Ready=1, Run=0, Bus_Err=1, Init_Err=0, Prm_Err=0, Alarm=0
old_number_modules=4, new_number_modules=3
```

This message shows that one module is defect.

Module area Slot 0 ... 31



This area shows all information about the in-/output periphery like module name, in-/output assignment, if existing parameter bytes and diagnosis data.

Module name: The order number of the module serves as module name. This allows an unambiguous identification of the module.

In-/output assignment: Here you find four informations:

- Type: input area (IB), output area (QB)
- The start address of the area is in brackets
- You see the number of bytes occupied by the module
- The content of the bytes corresponds to that of the Ethernet coupler at the last website update

Example:
```
Slot 4

IB[6]=
00 00 00 00
00 00 00 00
```

The `Prm()`= Parameter bytes contain the following information:

- The length of the parameter block is in brackets with a preceding `len`.
- The content of the bytes are the parameter bytes of the according module.

`DIAG` = showes 16Byte diagnosis data for the alarm handling.

Information about connected clients

This area gives you information about number and IP address of the clients that are communicating with the Ethernet coupler at the time via ModbusTCP res. Siemens S5 Header protocol. With every protocol, a max. of 8 clients may communicate simultaneously with the Ethernet slave.
The number is in <> followed by the IP addresses in [].

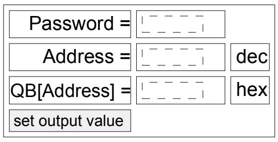
Example:
Number of ModbusTCP clients: <2>: [172.16.131.20] [172.16.140.63]
(At this time, 2 clients are communicating via ModbusTCP with the IP addresses 172.16.131.20 and 172.16.140.63.)

Elements for the active access...

Whereas the elements above are displaying information, the active access elements here allow to access the Ethernet coupler and its modules online.

Every control element is password protected. Use the password configured for your coupler (default = "00000000").
The following 5 control elements are available:

- Control outputs
- Parameterize module
- Reset the Ethernet coupler
- Configure Timeout
- Confirm alarm



Control outputs

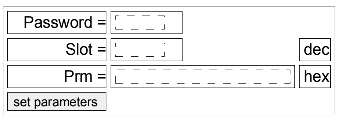
This control element allows you to set values into a wanted address area and transfer them via [set output value] to the Ethernet coupler.

Please regard that the address has to be a decimal number and the value a Hex number. You may transfer a max. of 4Byte to the address given in `Address`.

Please regard that the Bytes always have to be transferred with a leading zero. Space signs are serving as Byte separator.

Example: Address=0

```
QB[Address]= 12   →    QB[0]= 12 00
QB[Address]= 1 2  →    QB[0]= 01 02
QB[Address]= 1234 →    QB[0]= 12 34
QB[Address]= 123  →    QB[0]= 01 23
```



Parameterize module

This control element allows you to provide the module online with parameters by typing the parameter bytes into `Prm` and setting a plug-in location via `Slot`.

With [set parameters], the according parameters are transferred to the according module.

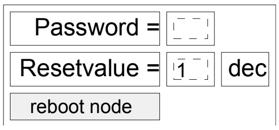
Please regard that the slot number has to be a decimal number and the parameter a Hex number.

Bytes are always transferred with a leading zero. A blank must be inserted as separator.

Note!

Always transfer the complete number of parameter bytes to a module, otherwise errors at the module may occur.

The number of parameters and their assignment is to find in the description of the concerning module.



Reset of the Ethernet coupler

Via [reboot node] a reset of the Ethernet coupler is initialized. After a re-boot, you have to update the website via <u>*home*</u>.

By presetting a r*eset value*, you may additionally to the re-boot of the Ethernet coupler delete the configuration or module parameters.

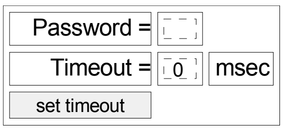
Permissible r*eset value* values are 1, 2 or 3. Other values are ignored!

Reset value= 1	Re-boot of the coupler (default setting)
Reset value= 2	Delete all module configurations (module names) and re-boot the coupler
Reset value= 3	Delete all module parameters and re-boot the coupler
Reset value= 4	Reset password (Default-Value "00000000")

Reset password

Resetting the password to its default value "00000000" is possible in a special operation mode.

- Power off the Ethernet coupler and pull it off from back plane bus.
- Power up the Ethernet coupler again
- Open the Web site of the Ethernet coupler by means of you Web browser and the IP address.
- Type in the default Password "00000000" at the parameter "reboot node"
- Set "Resetvalue =" 4 and click "reboot node". → Now the coupler resets its password to default value "00000000" and boots up again.



Configure Timeout

The coupler offers a connection timeout.

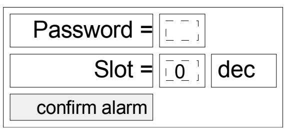
If the value 0 is transferred, this function is deactivated. (In the picture of the Ethernet coupler "Timout: off" is shown).

Note!

Choose "Timout: off" if you want to control outputs via internet browser otherwise all outputs are set to the secure state 0 after timeout.

With timeout values > 0msec, an I/O connection must read/write faster than the time value. If not, the connections are terminated and the outputs are set to the secure state 0.

The RD LED blinks and the website shows "rdy" in small letters.



Confirm alarm

Using "confirm alarm", it is possible to clear a slots alarm status bit. By setting a slot you need to set your couplers password and the slot (0 ... 31) where the alarm status bit shall be confirmed. Then clicking the button [confirm alarm] will clear the status bit and the "DiagAlarm" or "ProcAlarm" message will disappear.

ModbusTCP

General

ModbusTCP is a Modbus protocol put upon TCP/IP, where the IP address serves the addressing. The ModbusTCP allows a client-server-communication, several clients may be provided from one server.

Telegram structure incl. TCP/IP

The request telegrams sent by a master and the respond telegrams of the slave have the same structure:

ModbusTCP	Slave address	Function code	Data
6Byte-Header with number of following Bytes	1Byte data	1Byte data	max 254Byte

ModbusTCP-Header (6Byte)

For send and receive telegrams, ModbusTCP uses a header of 6Byte with the following structure:

ModbusTCP header

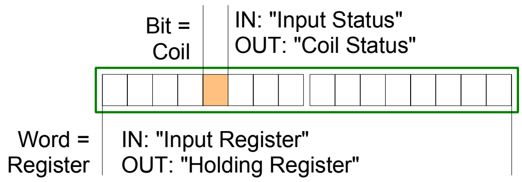
Byte	Name	Description
0	Transaction identifier (High-Byte)	Is sent back by the server (user-defined)
1	Transaction identifier (Low-Byte)	Is sent back by the server (user-defined)
2	Protocol identifier (High-Byte)	Always 0
3	Protocol identifier (Low-Byte)	Always 0
4	Length field (High-Byte)	Always 0 because messages < 256Byte
5	Length field (Low-Byte)	Number of following bytes

Normally, Byte 0 ... 4 have the value 0. You may also increase Byte 0 and 1 in the slave and thus establish an additional control.

Modbus function codes

Naming convention

Modbus has some naming conventions:



- Modbus differentiates between bit and word access;
  Bits = "Coils" and Words = "Register".
- Bit inputs are referred to as "Input-Status" and Bit outputs as "Coil-Status".
- Word inputs are referred to as "Input-Register" and Word outputs as "Holding-Register".

Range definitions

Normally the access under Modbus happens by means of the ranges 0x, 1x, 3x and 4x.

0x and 1x gives you access to *digital* Bit areas and 3x and 4x to *analog* word areas.

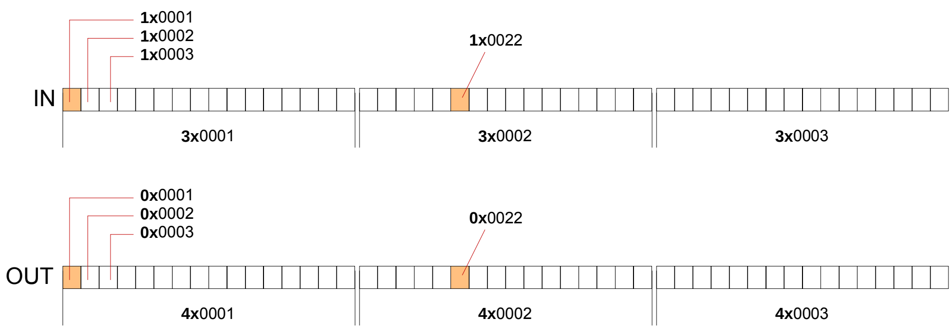
For the Ethernet coupler from VIPA is not differentiating digital and analog data, the following assignment is valid:

0x: Bit area for master output
Access via function code 01h, 05h, 0Fh

1x: Bit area for master input
Access via function code 02h

3x: Word area for master input
Access via function code 04h, 17h

4x: Word area for master output
Access via function code 03h, 06h, 10h, 17h



A description of the function codes follows below.

Overview

With the following Modbus function codes a Modbus master can access a Modbus slave. The description always takes place from the point of view of the master:

Code	Command	Description
01h	Read n Bits	Read n Bits of master output area 0x
02h	Read n Bits	Read n Bits of master input area 1x
03h	Read n Words	Read n Words of master output area 4x
04h	Read n Words	Read n Words master input area 3x
05h	Write one Bit	Write 1 Bit to master output area 0x
06h	Write one Word	Write 1 Word to master output area 4x
0Fh	Write n Bits	Write n Bits to master area 0x
10h	Write n Words	Write n Words to master area 4x
17h	Write n Words and Read m Words	Write n words into master output area 4x and the respond contains m read words of the master input area 3x

The Ethernet coupler from VIPA does not differentiate between digital and analog data!

Note!
The Byte sequence in a Word always is:

1 Word	
High Byte	Low Byte

Respond of the coupler

If the slave announces an error, the function code is send back with a "OR" and 80h. Without an error, the function code is sent back.

Coupler answer: Function code OR 80h → Error
Function code → OK

Read n Bits 01h, 02h

This function enables the reading from a slave bit by bit.

Command telegram

ModbusTCP-Header	Slave address	Function code	Address 1st Bit	Number of Bits
x x 0 0 0 6				
6Byte	1Byte	1Byte	1Word	1Word

Respond telegram

ModbusTCP-Header	Slave address	Function code	Number of read Bytes	Data 1st Byte	Data 2nd Byte	...
x x 0 0 0						
6Byte	1Byte	1Byte	1Byte	1Byte	1Byte max. 252Byte	

max. 255Byte

Read n Words 03h, 04h

This function enables the reading from a coupler word by word.

Command telegram

ModbusTCP-Header	Slave address	Function code	Address Word	Number of Words
x x 0 0 0 6				
6Byte	1Byte	1Byte	1Word	1Word

Respond telegram

ModbusTCP-Header	Slave address	Function code	Number of read Bytes	Data 1st Word	Data 2nd Word	...
x x 0 0 0						
6Byte	1Byte	1Byte	1Byte	1Word	1Word	

max. 255Byte

max. 126Words

Write a Bit 05h

This function allows to alter a Bit in your coupler. A status change happens via "Status Bit" with the following values:

"Status Bit" = 0000h → Bit = 0, " Status Bit" = FF00h → Bit = 1

Command telegram

ModbusTCP-Header	Slave address	Function code	Address Bit	Status Bit
x x 0 0 0 6				
6Byte	1Byte	1Byte	1Word	1Word

Respond telegram

ModbusTCP-Header	Slave address	Function code	Address Bit	Status Bit
x x 0 0 0 6				
6Byte	1Byte	1Byte	1Word	1Word

Write a word 06h

This function sends a word to the coupler. This allows to overwrite a register in the coupler.

Command telegram

ModbusTCP-Header	Slave address	Function code	Address Word	Value Word
x x 0 0 0 6				
6Byte	1Byte	1Byte	1Word	1Word

Respond telegram

ModbusTCP-Header	Slave address	Function code	Address Word	Value Word
x x 0 0 0 6				
6Byte	1Byte	1Byte	1Word	1Word

Write n Bits 0Fh

This function writes n Bits to the slave. Please regard that the number of Bits has additionally given in Byte.

Command telegram

ModbusTCP-Header	Slave address	Function code	Address 1st Bit	Number of Bits	Number of Bytes	Data 1st Byte	Data 2nd Byte	...
x x 0 0 0								
	1Byte	1Byte	1Word	1Word	1Byte	1Byte	1Byte	1Byte

max. 255Byte

max. 248Byte

Respond telegram

ModbusTCP-Header	Slave address	Function code	Address 1st Bit	Number of Bits
x x 0 0 0 6				
	1Byte	1Byte	1Word	1Word

Write n Words 10h

Via this function you may write n Words to the slave.

Command telegram

ModbusTCP-Header	Slave address	Function code	Address 1st Word	Number of Words	Number of Bytes	Data 1st word	Data 2nd word	...
x x 0 0 0								
	1Byte	1Byte	1Word	1Word	1Byte	1Word	1Word	1Word

max. 255Byte; Data: max. 124Words

Respond telegram

ModbusTCP-Header	Slave address	Function code	Address 1st Word	Number of Words
x x 0 0 0 6				
	1Byte	1Byte	1Word	1Word

Write n Words and Read m Words 17h

This function allows to write n words and read m words with a request.

Command telegram

ModbusTCP-Header	Slave address	Functions code	Read address	Read number of words	Write address	Write No. of words	Write No. of Bytes	Write Data 1st word	Write Data 2nd word	...
x x 0 0 0										
	1Byte	1Byte	1Word	1Word	1Word	1Word	1Byte	1Word	1Word	

max. 255Byte; Write Data: max. 122Words

Respond telegram

ModbusTCP-Header	Slave address	Functions code	Read number of Bytes	Read Data 1st word	Read Data 2nd word	...
x x 0 0 0						
6Byte	1Byte	1Byte	1Byte	1Word	1Word	

max. 255Byte; Read Data: max. 126Words

Siemens S5 Header Protocol

General

The Siemens S5 Header protocol serves the data exchange between PLC systems. Deploying the organization format (short ORG) that is included in the Siemens S5 Header protocol, a short description of a data source res. destination in PLC environment is possible.

ORG formats

The used ORG formats are corresponding to the Siemens specifications and are listed in the following table.

The ORG block is optional at READ and WRITE.

The ERW specification is irrelevant for the Ethernet coupler.

The start address and the number are addressing the memory area and are stored in HIGH-/LOW format (Motorola – Address format)

Description	Type	Area
ORG specification	BYTE	1..x
ERW specification	BYTE	irrelevant
Start address	HILOWORD	0..y
Number	HILOWORD	1..z

The following table lists the useable ORG formats. The "length" may not be specified as -1 (FFFFh).

ORG specification *02h-05h*

<table>
<tr><th>CPU area</th><th>MB</th><th>EB</th><th>AB</th><th>PB</th></tr>
<tr><td>ORG specification</td><td>02h</td><td>03h</td><td>04h</td><td>05h</td></tr>
<tr><td>Description</td><td rowspan="4">Only permitted:
Read MB0 with length 4.

The total length of the in- and output areas is calculated and stored in MB0 ... MB3 in this format:

MB0: Length In area
MB1: 00
MB2: Length Out area
MB3: 00</td><td>Source/destination data out/in
Process image inputs (PAE).</td><td>Source/destination data out/in
Process image outputs (PAA).</td><td>Source/destination data out/in peripheral module
At source data input modules, at destination data output modules.</td></tr>
<tr><td>DBNR</td><td>irrelevant</td><td>irrelevant</td><td>irrelevant</td></tr>
<tr><td>Start address
Meaning

Permitted range:</td><td>EB-No. from where on the data is fetched resp. written.

0...255</td><td>AB-No. from where on the data is fetched resp. written.

0...255</td><td>PB-No. from where on the data is fetched resp. written.

0... 65535</td></tr>
<tr><td>Number
Meaning

Permitted range:</td><td>Length of the source/destination data block in Bytes.

1...256</td><td>Length of the source/destination data block in Bytes.

1...256</td><td>Length of the source/destination data block in Bytes.

1...256</td></tr>
</table>

Structure PLC header

At READ and WRITE acknowledgement telegrams are created by the Ethernet coupler and request telegrams are expected with the format shown below. The headers have normally a length of 16Byte and have the following structure:

Client (PLC, PC) / **Server (Ethernet slave)**

at WRITE

Client (PLC, PC) – Request telegram

System spec.	="S"
	="5"
Length Header	=16d
Spec. OP-Code	=01
Length OP-Code	=03
OP-Code	**=03**
ORG-Block	=03
Length ORG-Block	=08
ORG specification	
DBNR	
Start address	H
	L
Length	H
	L
Empty block	=FFh
Length	=02
Data up to 64K but only if error no. =0	

Server (Ethernet slave) – Acknowledgement telegram

System spec.	="S"
	="5"
Length Header	=16d
Spec. OP-Code	=01
Length OP-Code	=03
OP-Code	**=04**
Ackn. block	=0Fh
Length ackn. Block	=03
Error No.	=Nr.
Empty block	=FFh
Length empty block	=07
free	

at READ

Client (PLC, PC) – Request telegram

System spec.	="S"
	="5"
Length Header	=16d
Spec. OP-Code	=01
Length OP-Code	=03
OP-Code	**=05**
ORG-Block	=03
Length ORG-Block	=08
ORG specification	
DBNR	
Start address	H
	L
Length	H
	L
Empty block	=FFh
Length	=02

Server (Ethernet slave) – Acknowledgement telegram

System spec.	="S"
	="5"
Length Header	=16d
Spec. OP-Code	=01
Length OP-Code	=03
OP-Code	**=06**
Ackn. block	=0Fh
Length ackn. Block	=03
Error No.	=Nr.
Empty block	=FFh
Length empty block	=07
free	
Data up to 64K but only if error no. =0	

Possible error numbers

The following error numbers may be included in the acknowledgement telegram:

0: no error

3: Address outside the defines area

6: No valid ORG format (Specification data source/destination is wrong).
Permitted: EB, AB, PB and MB

Principle of Alarm handling

Overview

Many of the System 200V modules are able to set an alarm (all non digital modules, e.g. analog modules, function modules, fieldbus masters).

As soon as one or more modules report an alarm, the alarm data of the appropriate slot location is received and acknowledged by the ethernet coupler.

After that the slot location assigned bit of the internal *alarm information image* is set and the diagnostic data with the length of 16Byte is stored.

In system 200V we distinguish between two types of alarms: the *process alarm* and the *diagnosis alarm.* A module will set only one of the alarm types at a time.

For differentiation, the alarm information image contains one 32bit wide field (bit 0 = slot 0 up to bit 31 = slot 31) called process alarm status and one 32bit wide field called diagnosis alarm status. After that, there follows for each slot 0 ... 31 a 16byte wide field called alarmdata.

For acknowledgement you can also access diagnostic and process alarm status writing. The 16byte alarmdata is read only.

alarm information image

The *alarm information image* with a size of 520Byte is placed behind the 256Byte I/O data and has the following structure:

32Bit process alarm status (little endian formatted): Bit 0 = slot 0 ... Bit 31 = slot 31
32Bit diagnosis alarm status (little endian formatted): Bit 0 = slot 0 ... Bit 31 = slot 31
16Byte alarmdata of slot 0
16Byte alarmdata of slot 1
... etc.
16Byte alarmdata of slot 31

Output Diagnosis

Web-Server

All alarm capable modules feature the entry "Diag=" with the latest 16byte of alarmdata. With an alarm set, the message "DiagAlarm" for diagnosis alarm resp. "ProcAlarm" for process alarm is displayed.

Slot 4
231-1BD52

IB[6]=
00 27 af 00
00 00 2d 04

Prm(len10)=
00 00 2d 2d
28 28 00 00
00 00

Diag=
0d 15 00 00
74 08 04 04
00 00 01 00
00 00 00 00
DiagAlarm

ModbusTCP
Read starting at register 3x0129:

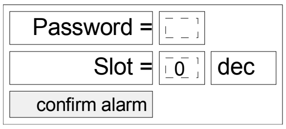
Register	Content
3x0129	process alarm status: Byte 0, Byte 1
3x0130	process alarm status: Byte 2, Byte 3
3x0131	diagnosis alarm status: Byte 0, Byte 1
3x0132	diagnosis alarm status: Byte 2, Byte 3
3x0133	Slot 0: alarmdata 16Byte
3x0141	Slot 1: alarmdata 16Byte
3x0149	Slot 2: alarmdata 16Byte
3x0157	Slot 3: alarmdata 16Byte
3x0165	Slot 4: alarmdata 16Byte
3x0173	Slot 5: alarmdata 16Byte
3x0181	Slot 6: alarmdata 16Byte
3x0189	Slot 7: alarmdata 16Byte
3x0197	Slot 8: alarmdata 16Byte
3x0205	Slot 9: alarmdata 16Byte
3x0213	Slot 10: alarmdata 16Byte
3x0221	Slot 11: alarmdata 16Byte
3x0229	Slot 12: alarmdata 16Byte
3x0237	Slot 13: alarmdata 16Byte
3x0245	Slot 14: alarmdata 16Byte
3x0253	Slot 15: alarmdata 16Byte
3x0261	Slot 16: alarmdata 16Byte
3x0269	Slot 17: alarmdata 16Byte
3x0277	Slot 18: alarmdata 16Byte
3x0285	Slot 19: alarmdata 16Byte
3x0293	Slot 20: alarmdata 16Byte
3x0301	Slot 21: alarmdata 16Byte
3x0309	Slot 22: alarmdata 16Byte
3x0317	Slot 23: alarmdata 16Byte
3x0325	Slot 24: alarmdata 16Byte
3x0333	Slot 25: alarmdata 16Byte
3x0341	Slot 26: alarmdata 16Byte
3x0349	Slot 27: alarmdata 16Byte
3x0357	Slot 28: alarmdata 16Byte
3x0365	Slot 29: alarmdata 16Byte
3x0373	Slot 30: alarmdata 16Byte
3x0381	Slot 31: alarmdata 16Byte

Siemens S5 Header

Write starting at periphery byte 256:

Byte address	Content
256	process alarm status: Byte 0, Byte 1
258	process alarm status: Byte 2, Byte 3
260	diagnosis alarm status: Byte 0, Byte 1
262	diagnosis alarm status: Byte 2, Byte 3
264	Slot 0: alarmdata 16Byte
280	Slot 1: alarmdata 16Byte
296	Slot 2: alarmdata 16Byte
312	Slot 3: alarmdata 16Byte
328	Slot 4: alarmdata 16Byte
344	Slot 5: alarmdata 16Byte
360	Slot 6: alarmdata 16Byte
376	Slot 7: alarmdata 16Byte
392	Slot 8: alarmdata 16Byte
408	Slot 9: alarmdata 16Byte
424	Slot 10: alarmdata 16Byte
440	Slot 11: alarmdata 16Byte
456	Slot 12: alarmdata 16Byte
472	Slot 13: alarmdata 16Byte
488	Slot 14: alarmdata 16Byte
504	Slot 15: alarmdata 16Byte
520	Slot 16: alarmdata 16Byte
536	Slot 17: alarmdata 16Byte
552	Slot 18: alarmdata 16Byte
568	Slot 19: alarmdata 16Byte
584	Slot 20: alarmdata 16Byte
600	Slot 21: alarmdata 16Byte
616	Slot 22: alarmdata 16Byte
632	Slot 23: alarmdata 16Byte
648	Slot 24: alarmdata 16Byte
664	Slot 25: alarmdata 16Byte
680	Slot 26: alarmdata 16Byte
696	Slot 27: alarmdata 16Byte
712	Slot 28: alarmdata 16Byte
728	Slot 29: alarmdata 16Byte
744	Slot 30: alarmdata 16Byte
760	Slot 31: alarmdata 16Byte

Confirm alarm



Web-Server

With WinNCS (Version > V320) it will be possible to activate the web control "confirm alarm". Using that, it is possible to clear a slots alarm status bit. You need to set your couplers password and the slot (0 ... 31) where the alarm status bit shall be confirmed. Then clicking the button [confirm alarm] will clear the status bit and the "DiagAlarm" resp. "ProcAlarm" message should be deleted.

ModbusTCP

Write starting at register 4x0129:

Register	Content
4x0129	process alarm status: Byte 0, Byte 1
4x0130	process alarm status: Byte 2, Byte 3
4x0131	diagnosis alarm status: Byte 0, Byte 1
4x0132	diagnosis alarm status: Byte 2, Byte 3

Siemens S5 Header

Write starting at periphery byte 256:

Byte address	Content
256	process alarm status: Byte 0, Byte 1
258	process alarm status: Byte 2, Byte 3
260	diagnosis alarm status: Byte 0, Byte 1
262	diagnosis alarm status: Byte 2, Byte 3

Typical application

A typical application watches the alarm status fields and checks their value. For "0" there is nothing to do, because no alarm has occurred. If process alarm status or diagnosis alarm status are <> "0", there have been found one or more alarms and there are updated alarmdata fields to read. Those should be evaluated (find out about modules/channels state, e.g. wire break) and then the alarm status field should be set to zero. We call that "to confirm alarms". Now continue with watching for alarms (polling).

More than one alarm from different slots:

If there was signalled an alarm from more than one slots at a time, the appropriate alarm status bit will be set to "1" and each corresponding alarmdata field will be updated. So there is no loss of information!

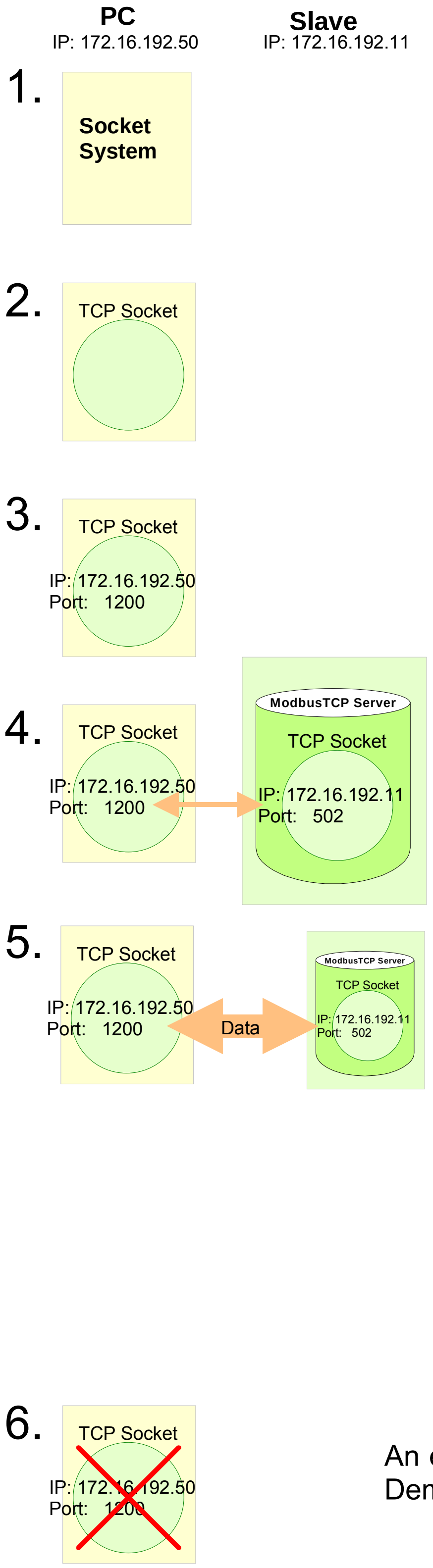
More than one alarm from one slot:

If there came more than one alarms from one slot, the slots alarm status bit will be set and keep on "1" (logical OR). In the corresponding alarmdata field the latest alarmdata may be read. The alarms history and how many alarms occurred is unknown! But at least it is assured that always the current alarm status and alarmdata is available.

Programming sample

Steps of Programming

For the deployment of the Ethernet couplers at a PC you should have a thorough knowledge in C programming, especially in socket programming. This section gives you a short overview about the programming.



to 1. Start Microsoft Socket System

```
WSAStartup (wVersionRequested, &wsaData);
```

to 2. Reserve Socket resources for TCP

```
m_lsock = socket (AF_INET, SOCK_STREAM, 0):
```

to 3. Link-up the socket to the local PC

```
SockAddr.sin_port = htons( 0 );
SockAddr.sin_addr.S_un.S_addr = inet_addr( "0.0.0.0" );
bind(m_lsock, (LPSOCKADDR) &SockAddr, sizeof(SockAddr));
```

By calling `bind` with the value 0 for port and IP address, the socket gets the PC-IP address and the next free Port.
(here: IP: 172.16.192.50, Port: 1200)

to 4. Establish connection to external device

```
SockAddr.sin_port = htons (m_wPort);
SockAddr.sin_addr.S_un.S_addr = inet_addr(m_szIpAddress);
connect(m_lsock, (LPSOCKADDR) &SockAddr, sizeof(SockAddr));
```

to 5. For write res. read access you have to build up telegrams according to the protocol and store them in sndBuf.

sndBufLen contains the number of Bytes to be sent.

Read access

Send sndBuf (Request)

```
send(m_lsock, (char *)sndBuf, sndBufLen, 0);
```

Receive telegram in rcvBuf (Response+data)

```
recv(m_lsock, (char *)rcvBuf, sizeof(rcvBuf), 0);
```

Write access

Send sndBuf (Request+data)

```
send(m_lsock, (char *)sndBuf, sndBufLen, 0);
```

Receive telegram in rcvBuf (Response)

```
recv(m_lsock, (char *)rcvBuf, sizeof(rcvBuf), 0);
```

to 6. Close socket again

```
closesocket(m_lsock);
```

An easy programming sample can be downloaded under ftp.vipa.de/support
Demo Client: Cx000059.

Technical data

IM 253NET

Electrical data	VIPA 253-1NE00
Voltage supply Current consumption Output current backplane bus Potential separation Status monitor Interfaces	DC 24V (20.4 ... 28.8V) via front from ext. power supply 120mA 3.5A ≥ AC 500V Via LEDs at the front side RJ45 for Twisted-Pair-Ethernet
Ethernet Interface	
Connection Network topology Medium Transfer rate Total length	RJ45 Star topology Twisted Pair 10/100MBit max. 100m per segment
Online access	
Test/Diagnosis Project engineering	http server integrated that graphically displays the configuration via website and supports parameterization and project engineering options for test purposes. Via WinNCS with online coupler search and engineering
Combination with peripheral modules	
max. number of clients max. number of input byte max. number of output byte	8 per ModbusTCP res. Siemens S5 protocol 256 256
Dimensions and Weight	
Dimensions (WxHxD) in mm Weight	25.4x76x78 70g

Chapter 9 Bus expansion modules IM 260 - IM 261

Overview

In this chapter follows the description of the bus expansion module that is used to split a single System 200V row over up to 4 rows. Here the maximum number of 32 modules may not be exceeded.

Below follows a description of:

- Field of application
- Proceeding with the wiring
- LEDs
- Technical data

Contents

Topic **Page**

Chapter 9 Bus expansion modules IM 260 - IM 261 9-1

Field of application 9-2

Wiring.. 9-3

Status indication 9-4

Technical data 9-5

Ordering data

Type	Order number	Description
IM 260	VIPA 260-1AA00	Basic interface IM 260
IM 261	VIPA 260-1AA00	Row interface IM 261
Cable 0.5m	VIPA 260-1XY05	Interconnecting cable, 0.5m length
Cable 1m	VIPA 260-1XY10	Interconnecting cable, 1m length
Cable 1.5m	VIPA 260-1XY15	Interconnecting cable, 1.5m length
Cable 2m	VIPA 260-1XY20	Interconnecting cable, 2m length
Cable 2.5m	VIPA 260-1XY25	Interconnecting cable, 2.5m length

Field of application

Overview

The system consisting of IM 260, IM 261 and interconnecting cables is an expansion option that you use to split the System 200V over up to 4 rows.

This system may only be installed in a centralized System 200V where a PC 288 or a CPU is employed as the master station!

For bus expansion purposes you always have to include the basic interface IM 260. The basic interface may then be connected to up to 3 additional System 200V rows by means of the appropriate interconnecting cables and the IM 261 interfacing module for rows.

Please note!

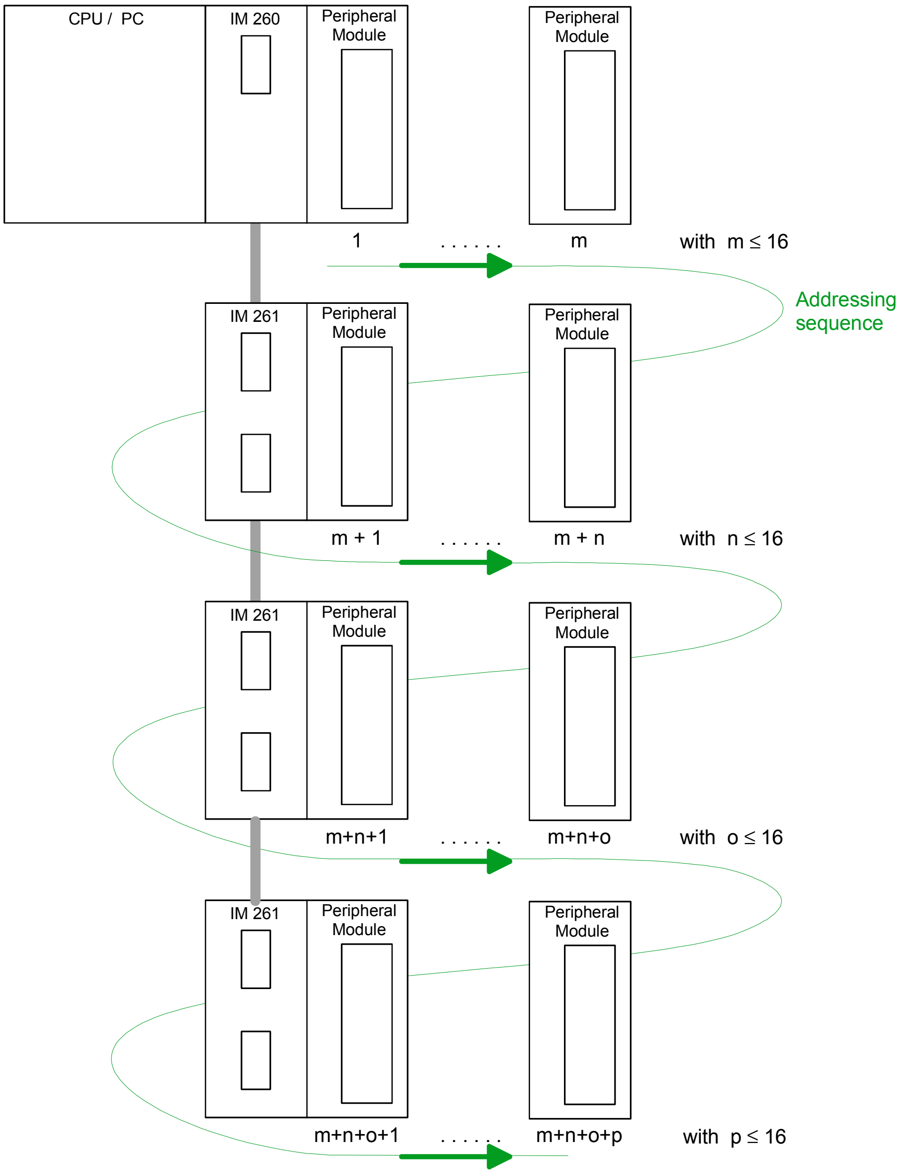
Certain rules and regulations have to be observed when the bus expansion modules are being employed:

- The bus expansion may only be used in conjunction with the PC 288 (VIPA 288-2BL10) or a CPU (combi-CPUs are also permitted). The system must never be employed in decentralized systems, e.g. behind a Profibus-DP slave!
- The system caters for a maximum of 4 rows.
- Every row can carry a maximum of 16 peripheral modules.
- The max. total quantity of 32 peripheral modules may not be exceeded.
- In critical environments the total length of interconnecting cables should not exceed a max. of 2m.
- Every row may derive a max. current of 1.5A from the backplane bus, while the total current is limited to 4A.
- At least one peripheral module must be installed next to the IM 260 basic interface!

Wiring

Configuration

The following figure shows the structure of a bus expansion under observance of the installation requirements and rules:



Where: $m + n + o + p \leq 32$

Note!

The bus expansion may only be used in conjunction with the PC 288 (VIPA 288-2BL10) or a CPU (combi-CPUs are also permitted)!

The bus expansion module is supported as of the following minimum firm-ware revision levels:

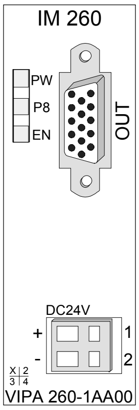
CPU compatible with Siemens STEP®5: from Version 2.07

CPU compatible with Siemens STEP®7: from Version 1.0

Status indication

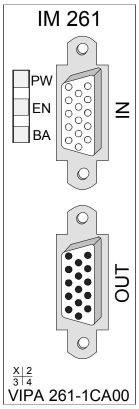
Status indication Basic interface IM 260

LED	Color	Description
PW	green	Supply voltage available
P8	yellow	Supply voltage for subsequent rows is active
EN	yellow	Backplane bus communications active



Status indication row interface IM 261

LED	Color	Description
PW	green	Supply voltage available via IM 260
EN	yellow	Backplane bus communication active
BA	red	Outputs inhibited (BASP) is active



Technical data

IM 260
Basic interface

Electrical data	VIPA 260-1AA00
Power supply	DC 24V (20.4...28.8) external via front
Current consumption	1.9A
Current consumption backplane bus	30mA
max. cable distance betw. 1st and last row	2.5m
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x78
Weight	80g

IM 261
Row interface

Electrical data	VIPA 261-1CA00
Power supply	by IM 260
Power supply backplane bus	max. 1.5A per row (max. total 4A)
max. cable distance betw. 1st and last row	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x78
Weight	50g

Appendix

A Index

A

Address selector
- CANopen slave.....5-41, 5-46, 5-51
  - ECO 5-41
- DeviceNet coupler 6-6
- Profibus DP slave 3-37, 3-59
  - ECO 3-59
- SERCOS.................................7-6

B

Basics
- CANopen 5-4
- DeviceNet 6-3
- Ethernet 8-3
- Interbus..................................4-3
- Profibus DP 3-5
- SERCOS.................................7-3

Baudrate
- CANopen slave.......................5-51
- DeviceNet coupler 6-9
- SERCOS.................................7-8

Bus expansion 9-1
- Notes 9-2
- Technical data 9-5
- Wiring 9-3

C

CANopen 5-1
- Basics 5-4
- Bus access 5-5
- Master......................................5-6
  - Device model 5-20
  - Error messages..................5-22
  - fast introduction....................5-8
  - Firmware update 5-15
  - include GSD 5-9
  - Message structure 5-19
  - Object directory..................5-24
  - Operating modes 5-16
  - PDO 5-20
  - Project engineering 5-8
  - SDO 5-21
  - SFC 219 5-21
  - Structure 5-6
  - Technical data 5-104
  - WinCoCT deployment 5-10
- Slave..................................... 5-38
  - Address selector 5-51
  - Baudrate 5-51
  - Block diagram 5-42
  - Communication types 5-56
  - DO 24xDC 24V 5-43
  - ECO 5-38, 5-41
  - Error messages 5-49, 5-59
  - fast introduction 5-47
  - Function codes 5-53
  - Heartbeat 5-103
  - Identifier 5-47, 5-55
  - Message structure 5-52
  - Module types...................... 5-70
  - Module-ID 5-51
  - NMT 5-102
  - Node Guarding 5-103
  - Object directory.................. 5-60
  - PDO 5-54
    - linking 5-56
  - SDO 5-58
  - Status indicator 5-39
  - Structure 5-38
  - Technical data 5-105
  - Wiring........................5-40, 5-46

COB-ID 5-55

Commissioning
- CANopen master....................... 5-8
- CANopen slave 5-47
- DeviceNet coupler 6-8
- Interbus 4-15
- Profibus 3-89

Cyclic data communication........... 3-8

D

DeviceNet 6-1
- Addressing 6-4
- Basics 6-3
- Bus access 6-4
- Coupler 6-5

Address 6-9
Baudrate 6-9
Block diagram 6-7
Configuration 6-8
Diagnostics 6-17
Example 6-14
Interfacing 6-6
Parameterization 6-11
Structure 6-5
Test 6-10
-Manager 6-8
-Scanner 6-16
Diagnostic functions
CANopen slave 5-101
DeviceNet 6-17
DP-V0 slave 3-49
DP-V1 slave 3-68
Ethernet coupler 8-16
DP cycle 3-9

E

EasyConn 3-81
EDS file 6-4
Emergency Object 5-49, 5-101
Error messages
CANopen master: 5-22
CANopen slave 5-49, 5-59
DeviceNet coupler 6-17
DP-V0 slave 3-52
DP-V1 slave 3-72
Interbus 4-16
Ethernet 8-1
Basics 8-3
Coupler 8-9
Access 8-11
Addressing 8-14
Alarm handling 8-28
Delivery default 8-9
Diagnosis 8-16
Include GSD 8-15
ModbusTCP 8-20
Function codes 8-21
Range 8-21
OPC server 8-12
ORG format 8-26
Password 8-19
PLC header 8-27
Project engineering ...8-11, 8-15
Siemens S5 Header 8-26
Socket programming 8-32
Structure 8-9
Technical data 8-33
Web server 8-12, 8-16
ModbusTCP 8-6
Network planning 8-7
Protocols 8-4
Siemens S5 Header 8-6

F

Fiber optic FO 3-12, 3-82
Interface 3-14

H

Heartbeat 5-49, 5-103
Hub 8-3

I

ID code 4-13
ID length 4-13
ID register 4-3
Intel format 3-64
Interbus 4-1
Basics 4-3
Configuration of the master 4-17
Coupler 4-7
Commissioning 4-15
Error messages 4-16
Process image 4-11
Technical data 4-18
Wiring 4-10
Data consistency 4-14
Data transfer 4-5
Master 4-3
Operating modes 4-4
Interface
DeviceNet coupler 6-6
Interbus coupler 4-8
RJ45 Ethernet coupler 8-10
RS485 CANopen master 5-7
RS485 CANopen slave 5-40
RS485 Profibus DP master 3-14
RS485 Profibus DP slave 3-36
SERCOS coupler 7-6

L

LEDs
Bus expansion 9-4
CANopen master 5-7

CANopen slave.....5-39, 5-43, 5-44
DeviceNet coupler 6-6, 6-17
Ethernet coupler 8-10
Interbus coupler........................4-7
Profibus DP master 3-14
Profibus DP slave 3-36
Profibus slave 3-58
SERCOS coupler......................7-6

M

min_slave_interval 3-9
MMC 3-15
ModbusTCP................................8-20
Motorola format...........................3-64

N

NMT 5-102
Node Guarding 5-48, 5-103
Norm diagnostic data..................3-69

O

Operating modes
CANopen master 5-16
Interbus....................................4-4
Profibus master 3-16

P

Parameterization
DeviceNet coupler 6-11
Profibus DP-V0 slave 3-48
SERCOS..................................7-8
Parametrerization
Profibus DP-V1 slave 3-63
PDO 5-54
Power supply
CANopen master 5-7
CANopen slave.......................5-41
ECO 5-41
DeviceNet coupler 6-6
Ethernet coupler 8-10
Interbus coupler........................4-8
Profibus DP master 3-15
Profibus DP slave 3-37
ECO 3-59
SERCOS..................................7-6
Process image
CANopen master 5-17
DeviceNet coupler 6-16
Ethernet coupler 8-14
Interbus.................................4-11
Profibus 3-17
Profibus DP 3-1
Addressing 3-12
Basics...................................... 3-5
Commissioning....................... 3-89
Data communication 3-7
Data consistency 3-7
Data transfer protocol............... 3-7
Diagnostic LEDs..................... 3-90
DP-V0 slave 3-35
Basics 3-5
Block diagram 3-45
Data communication 3-8
Diagnostics 3-49
DO 24xDC 24V 3-38
Error messages 3-52
include GSD....................... 3-46
Parameters 3-48
Project engineering............ 3-46
redundant........................... 3-42
Start-up 3-47
Structure 3-35
DP-V1 slave 3-56
Addressing......................... 3-65
Basics 3-5, 3-10
Block diagram 3-60
Diagnostic 3-68
ECO 3-56, 3-59
Error messages 3-72
Firmware update................ 3-76
I&M data............................ 3-77
include GSD....................... 3-61
Interrupts........................... 3-73
Module configuration 3-67
Parameters 3-63
Project engineering............ 3-61
Start-up 3-62
Structure 3-56
Examples............................... 3-91
Fiber optic FO......................... 3-82
Installation guidelines 3-79
Master 3-6, 3-13
2bf file 3-22
Deployment at CPU 21x 3-17
Firmware update................ 3-33
Flash ROM......................... 3-25
MSAC_C1/2 3-6

Operating modes 3-16
Overall-Reset 3-32
slave operating mode......... 3-28
Structure............................ 3-13
Transfer variants 3-24
WInNCS deployment 3-23
wld file 3-22
Multi master system................ 3-86
Networks................................ 3-85
Optical and electrical Profibus 3-88
Optical Profibus 3-87
Redundancy state................... 3-55
Restrictions 3-8
Slave 3-6
Technical data 3-99
Token-passing procedure......... 3-7
Transfer medium 3-12
Profibus-DP
Connectors 3-81
Protocols...................................... 8-4

S

SDO 5-58
Segment length at Profibus 3-79
SERCOS...................................... 7-1
Addressing............................... 7-4
Basics 7-3
Block diagram 7-7
Coupler 7-5
Baudrate.............................. 7-8
Connection 7-6
Example 7-13
Identifier 7-10
Parameterization 7-8
Structure.............................. 7-5
Technical data.................... 7-22
Shift register................................. 4-3
SIP tool 3-27
Switch 8-3
System 200V
Assembly 2-1, 2-5
dimensions 2-10
Basics 1-1
Bus connector.......................... 2-2
Centralized system 1-4
Components 1-4
Decentralized system 1-4
Disturbances 2-12
EMC 2-12
Basic rules 2-13
Environmental conditions 1-5
Installation guidelines 2-12
Overview............................1-3, 1-5
Peripheral modules 1-4
Profile rail 2-2
Project engineering 1-4
Reliability 1-5
Removal 2-7
Safety Information 1-2
Screening of cables................ 2-14
Wiring 2-8
System overview
CANopen................................. 5-2
DeviceNet................................ 6-2
Ethernet................................... 8-2
Interbus 4-2
Profibus DP 3-2
SERCOS 7-2

T

Technical data
Bus expansion.......................... 9-5
CANopen master.................. 5-104
CANopen slave 5-105
DeviceNet coupler 6-22
Ethernet coupler 8-33
Interbus coupler...................... 4-18
Profibus DP master 3-99
Profibus DP slave................. 3-101
SERCOS coupler 7-22
Termination 3-80
Twisted Pair................................. 8-3
Restrictions.............................. 8-8

V

V-bus cycle 3-9

W

WinCoCT.................................... 5-10
WinNCS
with Ethernet coupler.............. 8-15
with Profibus DP..................... 3-23

M.Stich