



Handbücher/Manuals



**VIPA**  
**Gesellschaft für Visualisierung**  
**und Prozessautomatisierung mbH**

Ohmstraße 4  
D-91074 Herzogenaurach  
Tel.: +49-9132-744-0  
Fax: +49-9132-744-144  
Internet: [www.vipa.de](http://www.vipa.de)  
E-Mail: [Info@vipa.de](mailto:Info@vipa.de)

# Manual

## VIPA System 200V

Order No.: VIPA HB97E  
Rev. 03/51



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

© Copyright 2003 VIPA, Gesellschaft für Visualisierung und Prozessautomatisierung mbH  
Ohmstraße 4, D-91074 Herzogenaurach,  
Tel.: +49 (91 32) 744 -0  
Fax.: +49 (91 32) 744-144  
EMail: info@vipa.de  
<http://www.vipa.de>

**Hotline: +49 (91 32) 744-114**

All rights reserved

#### **Disclaimer of liability**

The contents of this manual were verified with respect to the hard- and software.

However, we assume no responsibility for any discrepancies or errors. The information in this manual is verified on a regular basis and any required corrections will be included in subsequent editions.

Suggestions for improvement are always welcome.

#### **Trademarks**

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

STEP und S7-300 are registered trademarks of Siemens AG.

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

## About this manual

This manual describes all System 200V components that are available from VIPA with the exception of the CPUs. 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 different System 200V components. Every chapter is concluded with the technical data of the respective module.

A separate set of manuals is available for the CPUs.

### Overview

#### **Chapter 1: Introduction**

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: 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 3: 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 4: 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 5: 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 6: 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 7: 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 8: PC 288 - CPU**

This chapter describes the PC-CPU PC 288 and applications in System 200V. The configuration of a PC-based system is described in detail. The chapter ends with an overview of the BIOS setup and the registers.

**Chapter 9: Communication processor CP 240**

This chapter contains information on the construction, the interfacing and the communication protocols of the communication processor CP 240. It also contains an explanation of the standard handler blocks for the VIPA CPU 21x and the CPU 24x.

**Chapter 10: Counter modules**

This chapter deals with VIPA counter modules. The chapter also contains information on the SSI module as well as the construction, configuration and the different counter modes along with the respective interfaces.

**Chapter 11: MotionControl modules FM 253 und FM 254**

The chapter describes the VIPA MotionControl stepper and the MotionControl servo module. It contains information on the assembly, operating modes, data transfer and applications in conjunction with a shaft encoder at FM 253 or FM 254.

**Chapter 12: Power supplies**

This chapter deals with external power supplies for the System 200V. Here you find a comprehensive set of safety related hints and information as well as details on the construction, the installation and commissioning of the module.

**Chapter 13-15: Digital input/output modules**

These chapters describe the digital remote I/O that is available from VIPA. It provides all the information that is required for applications using these modules. Chapter 13 contains information on the input modules, chapter 14 the information on the output modules and chapter 15 provides details on input/output modules.

**Chapter 16-18: Analog input/output modules**

These chapters contain a description of the analog remote I/O. The chapter also provides all the information that is required for applications using each module. Chapter 16 describes the input modules, chapter 17 the output modules and chapter 18 the analog input/output modules that are available from VIPA.

**Chapter 19: System expansion modules**

This chapter deals with the system expansion modules that are available for the System 200V. These include amongst others the bus expansion modules IM 26x that provide for the expansion of a single bus row to cater for several rows, the mini switch CM 240 and terminal modules required for the expansion of the available number of connections.

**Chapter 20: Installation 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.

# Contents

<b>User considerations .....</b>	<b>1</b>
<b>Safety information .....</b>	<b>2</b>
<b>Chapter 1 Introduction.....</b>	<b>1-1</b>
Safety information for Users .....	1-2
Hints for the deployment of the Green Cable from VIPA.....	1-3
Overview.....	1-4
Components .....	1-5
Overview over GSD-files from VIPA .....	1-6
General description System 200V.....	1-7
ISO/OSI reference model .....	1-8
Communication layers employed by automation systems .....	1-11
<b>Chapter 2 Profibus-DP .....</b>	<b>2-1</b>
System overview.....	2-2
Principles .....	2-5
IM 208DP - Master with RS485 - Construction.....	2-10
IM 208DP - Master with RS485 - Deployment at CPU 21x.....	2-13
IM 208DP - Master with RS485 - Project engineering .....	2-14
IM 208DP - Master with RS485 - Overall-Reset .....	2-22
IM 208DPO - Master with FO link - Construction .....	2-23
IM 208DPO - Master with FO link - Deployment at CPU 21x .....	2-26
IM 208DP - Master with FO link - Project engineering .....	2-27
IM 253DP - Slave (Standard) - Construction .....	2-33
IM 253DPR - Slave (redundant) - Construction.....	2-36
IM 253DP, DO 24xDC 24V - Construction .....	2-39
IM 253DP - Slave - Block diagram .....	2-43
IM 253DP - Slave - Project engineering .....	2-44
IM 253DP - Slave - Parameters .....	2-46
IM 253DP - Slave - Diagnostic functions.....	2-47
Installation guidelines.....	2-54
Commissioning .....	2-64
Using the diagnostic LEDs.....	2-65
Sample projects for Profibus communication .....	2-66
Technical data .....	2-74
<b>Chapter 3 Interbus .....</b>	<b>3-1</b>
System overview.....	3-2
Principles .....	3-3
IM 253IBS - Interbus coupler - Construction .....	3-7
Connection to Interbus.....	3-10
Deployment with Interbus .....	3-11
Commissioning .....	3-15
Technical data .....	3-18

<b>Chapter 4 CANopen .....</b>	<b>4-1</b>
System overview.....	4-2
Principles .....	4-3
IM 253CAN - CANopen Slave - Construction.....	4-5
IM 253CAN, DO 24xDC 24V - Construction.....	4-9
CANopen fast introduction .....	4-13
Baudrate and module-ID settings.....	4-17
Message structure .....	4-18
PDO - process data object.....	4-20
SDO - service data object.....	4-24
Object directory .....	4-26
Emergency Object .....	4-67
NMT - network management .....	4-68
Technical data .....	4-70
<b>Chapter 5 DeviceNet .....</b>	<b>5-1</b>
System overview.....	5-2
Principles .....	5-3
IM 253DN - DeviceNet coupler - Construction .....	5-5
Configuration by means of the DeviceNet-Manager.....	5-8
Specifying baudrate and DeviceNet address .....	5-9
Test in conjunction with the DeviceNet .....	5-10
Module configuration in the DeviceNet-Manager .....	5-11
I/O addressing of the DeviceNet scanner .....	5-16
Diagnostics .....	5-17
Profibus interface.....	5-22
Technical data .....	5-23
<b>Chapter 6 SERCOS .....</b>	<b>6-1</b>
System overview.....	6-2
Principles .....	6-3
IM 253Sercos - SERCOS coupler - Construction.....	6-5
Basic parameterization via address adjuster.....	6-8
SERCOS Identifier.....	6-10
Example for the automatic ID assignment .....	6-13
Technical Data.....	6-22
<b>Chapter 7 Ethernet coupler .....</b>	<b>7-1</b>
System overview.....	7-2
Principles of Ethernet.....	7-3
Planning a network .....	7-7
IM 253NET - Ethernet coupler - Construction .....	7-9
Access to the Ethernet coupler .....	7-11
Principle of the automatic address allocation .....	7-14
Project engineering under WinNCS .....	7-15
Diagnosis and test via Internet Browser.....	7-16
ModbusTCP.....	7-20
Modbus function codes.....	7-21



Siemens S5 Header Protocol.....	7-25
Programming sample.....	7-27
Technical data .....	7-28
<b>Chapter 8 PC 288 - CPU.....</b>	<b>8-1</b>
System overview.....	8-2
Principles .....	8-3
Properties .....	8-4
PC 288 - CPU - Construction .....	8-4
Components .....	8-5
Storage media applications.....	8-9
Deployment in the System 200V .....	8-10
Using the BIOS setup .....	8-13
Register description .....	8-21
Technical data .....	8-23
<b>Chapter 9 Communication processor CP 240.....</b>	<b>9-1</b>
System overview.....	9-2
Principles ASCII, STX/ETX, 3964(R), RK512 .....	9-3
Principles Modbus.....	9-10
CP 240 with 20mA/RS232C interface - Construction .....	9-11
CP 240 with RS422/RS485 interface - Construction .....	9-16
Parameterization.....	9-22
Access to the CP 240 interface under ASCII, STX/ETX, 3964(R).....	9-30
Deployment under Modbus .....	9-32
Modbus function codes.....	9-36
Example for the deployment under Modbus.....	9-39
Communication by means of standard handler blocks .....	9-45
Standard handler blocks for the CPU 24x.....	9-46
Standard handler blocks for the CPU 21x.....	9-61
Technical data .....	9-78
<b>Chapter 10 Counter modules .....</b>	<b>10-1</b>
System overview.....	10-2
FM 250S - SSI-Interface - Construction .....	10-3
FM 250 - Counter module - Construction .....	10-9
Summary of counter modes and interfacing .....	10-12
Counter modes .....	10-14
Technical data .....	10-58
<b>Chapter 11 MotionControl Modules.....</b>	<b>11-1</b>
System Overview.....	11-2
FM 253 - MotionControl Stepper .....	11-3
FM 253 - MotionControl Stepper - Construction.....	11-4
FM 253 - Connecting a drive.....	11-6
FM 253 - Data transfer >> FM 253.....	11-8
FM 253 - Parameterization .....	11-9
FM 253 - Operating modes .....	11-11
FM 253 - Data transfer >> CPU .....	11-15
FM 253 - Handling blocks .....	11-17

FM 254 - MotionControl Servo .....	11-23
FM 254 - MotionControl Servo - Construction .....	11-24
FM 254 - Connecting a drive with encoder .....	11-26
FM 254 - Summary of parameters and transfer values .....	11-28
FM 254 - Parameterization .....	11-29
FM 254 - Data transfer >> FM 254 .....	11-30
FM 254 - Operating modes .....	11-31
FM 254 - Data transfer >> CPU .....	11-37
Technical data .....	11-38
<b>Chapter 12 Power supplies .....</b>	<b>12-1</b>
Safety precautions .....	12-2
System overview .....	12-3
PS 207/2 - Power supply - Construction .....	12-4
PS 207/2CM - Power supply with Clamps - Construction .....	12-6
Installation .....	12-8
Wiring .....	12-9
Technical data .....	12-10
<b>Chapter 13 Digital input modules .....</b>	<b>13-1</b>
System overview .....	13-2
DI 8xDC 24V .....	13-4
DI 8xDC 24V 0.2ms .....	13-6
DIa 8xDC 24V .....	13-8
DI 8xDC 24V NPN .....	13-10
DI 4xAC/DC 90...230V .....	13-12
DI 8xAC/DC 60...230V .....	13-14
DI 8xAC/DC 24...48V .....	13-16
DI 8xAC 240V .....	13-18
DI 8xAC/DC 180...265V .....	13-20
DI 16xDC 24V with UB4x .....	13-22
DI 16xDC 24V .....	13-24
DI 16xDC24V/1C .....	13-26
DI 16xDC 24V NPN .....	13-36
DI 32xDC 24V .....	13-38
<b>Chapter 14 Digital output modules .....</b>	<b>14-1</b>
System overview .....	14-2
DO 8xDC 24V 1A .....	14-4
DO 8xDC 24V 2A .....	14-6
DO 8xDC 24V 2A separated 4 á 2 .....	14-8
DO 16xDC 24V 0.5A with UB4x .....	14-10
DO 16xDC 24V 1A .....	14-12
DO 16xDC 24V 2A .....	14-14
DO 16xDC 24V 0.5A NPN .....	14-16
DO 32xDC 24V 1A .....	14-18
DO 8xRelay COM .....	14-20
DO 4xRelay .....	14-22
DO 4xRelay bistable .....	14-24

DO 8xSolid State COM .....	14-26
DO 4xSolid State .....	14-28
<b>Chapter 15 Digital input/output modules .....</b>	<b>15-1</b>
System overview .....	15-2
Security hints for DIO modules .....	15-2
DIO 8xDC 24V 1A .....	15-3
DI 16xDC 24V, DO 16xDC 24V 1A .....	15-5
<b>Chapter 16 Analog input modules .....</b>	<b>16-1</b>
System overview .....	16-2
General .....	16-4
AI 4x16Bit, multiinput .....	16-5
AI 4x12Bit, 4 ... 20mA, isolated .....	16-16
AI 4x12Bit, $\pm 10V$ , isolated .....	16-19
AI 4x16Bit f .....	16-22
AI 8x16Bit .....	16-32
<b>Chapter 17 Analog output modules .....</b>	<b>17-1</b>
System overview .....	17-2
General .....	17-3
AO 4x12Bit, multioutput .....	17-4
AO 4x12Bit f, multioutput .....	17-12
<b>Chapter 18 Analog input/output modules .....</b>	<b>18-1</b>
System overview .....	18-2
Security note for range allocation .....	18-2
General .....	18-3
AI 2/AO 2x12Bit, multiin-/output .....	18-4
<b>Chapter 19 System expansion modules .....</b>	<b>19-1</b>
System overview .....	19-2
Bus expansion IM 260, IM 261 .....	19-4
4port mini switch CM 240 .....	19-7
Terminal module CM 201 .....	19-10
<b>Chapter 20 Assembly and installation guidelines .....</b>	<b>20-1</b>
Overview .....	20-2
Assembly .....	20-4
Wiring .....	20-9
Installation dimensions .....	20-11
Automatic labeling .....	20-12
Installation guidelines .....	20-13
<b>Appendix .....</b>	<b>A-1</b>
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** At present the manual consists of 20 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 Introduction

## Outline

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.

The chapter ends with a description of the 7 layer model and a table of the communication levels available in automation technology.

Below follows a description of:

- Introduction of the System 200V
- General information, i.e. installation, operational safety and environmental conditions
- 7 layer model and communication layers

## Content

Topic	Page
<b>Chapter 1 Introduction.....</b>	<b>1-1</b>
Safety information for Users .....	1-2
Hints for the deployment of the Green Cable from VIPA.....	1-3
Overview.....	1-4
Components .....	1-5
Overview over GSD-files from VIPA .....	1-6
General description System 200V.....	1-7
ISO/OSI reference model .....	1-8
Communication layers employed by automation systems .....	1-11

## 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.



## Hints for the deployment of the Green Cable from VIPA

### What is the Green Cable?



The Green Cable is a green connection cable, manufactured exclusively for the deployment at VIPA System components.

The Green Cable is a programming and download cable for VIPA CPUs 11x, 21x, 31x, 51x 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<sup>2</sup>I jack. This allows you to connect components to your VIPA-CPU that are able to communicate serial via an 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 CPUs 11x, 21x, 31x, 51x 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 be deployed directly at the concerning jacks of the VIPA components (in between plugs are not permitted).

At this time, the following components support the Green Cable:

CPUs 11x, 21x, 31x, 51x and the fieldbus masters 208-1xx01 from VIPA.



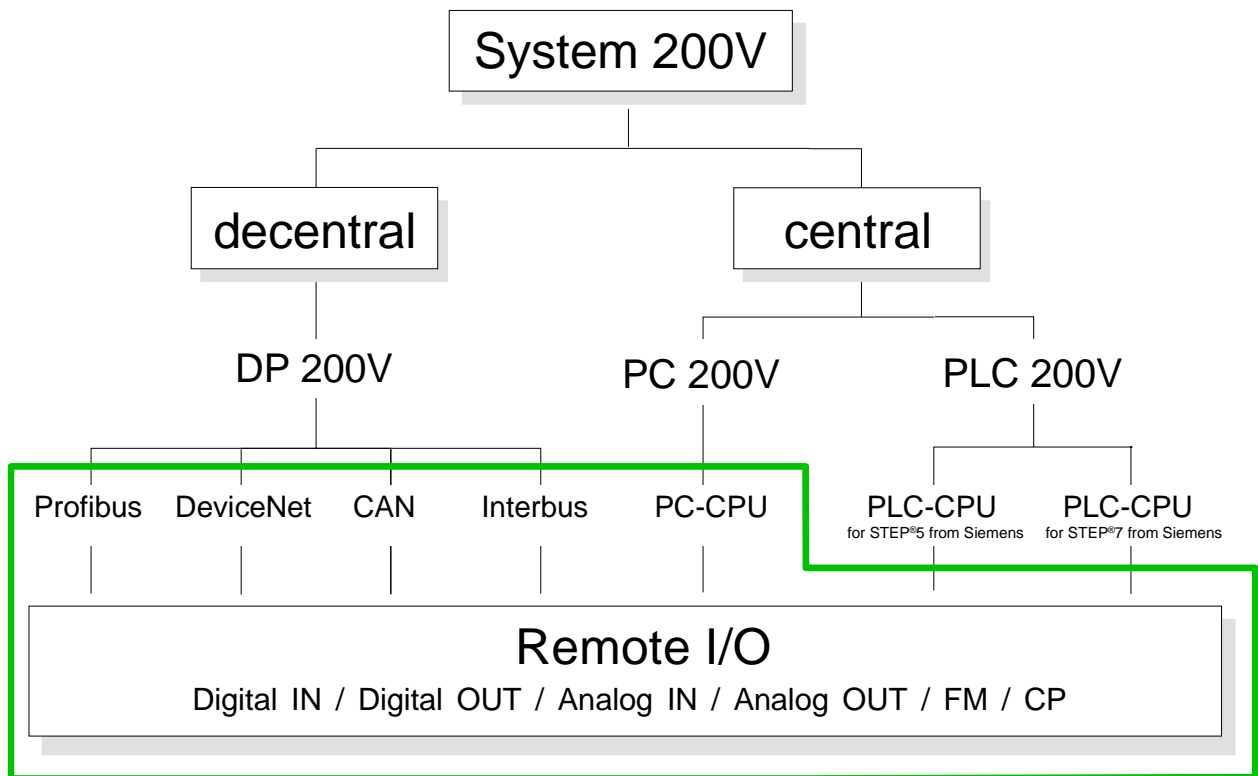
### Note to the lengthening

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).

## Overview

**The System 200V** The System 200 V 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-CPU's. These are programmed in STEP<sup>®</sup>5 or STEP<sup>®</sup>7 from Siemens.

CPU's 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-CPU's 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 any common configuration tool.

The module can also be configured directly via the Profibus network by means of the VIPA software WinNCS when this is used in conjunction with a Profibus master PC plug-in module that is available from the Softing company. Alternatively, all Profibus modules are available with a plastic FO-connector.

Other fieldbus systems may be connected by means of slaves for INTERBUS-S, CANopen and DeviceNet.

### 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.

## Overview over GSD-files from VIPA

### General

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 STEP<sup>®</sup>7 manager from Siemens.

For every system family there is an own GSD-file. The assignment to the single systems and the according name is in the hardware catalog (system label) is to find in the following table:

GSD		System			Comment
File name	Symbol label	100V	200V	300V	
DP2V0550.GSD	VIPA_DP200V	✓	✓		Old GSD-file is no longer updated
<b>System 100V</b>					
VIPA04D4.GSD	VIPA_DP100V	✓			For including decentral periphery 15x into DP master system
VIPA_11x.GSD	VIPA_CPU11x	✓			For projekt engineering of the direct I/O periphery in the CPU as virtual Profibus system
VIPA04Dx.GSD	VIPA_CPU11xDP	✓			For including CPU 11x as intelligent slave into DP master system
<b>System 200V</b>					
VIPA0550.GSD	VIPA_DP200V_2		✓		For including IM 253 slave into DP master system
VIPA_21x.GSD	VIPA_CPU21x		✓		For project engineering of the direct I/O periphery in the CPU as virtual Profibus system
VIPA04D5.GSD	VIPA_CPU2xxDP		✓		For including CPU 21x as intelligent slave into DP master system
VIPA2ETH.GSD	VIPA_ETH200V		✓		For project engineering of IM 253NET in WinNCS
<b>System 300V</b>					
VIPA056B.GSD	VIPA_DP300V			✓	For including IM 353 slave into DP master system
VIPA802F.GSD	VIPA_CPU31xDP			✓	For including CPU 31x as intelligent slave into DP master system

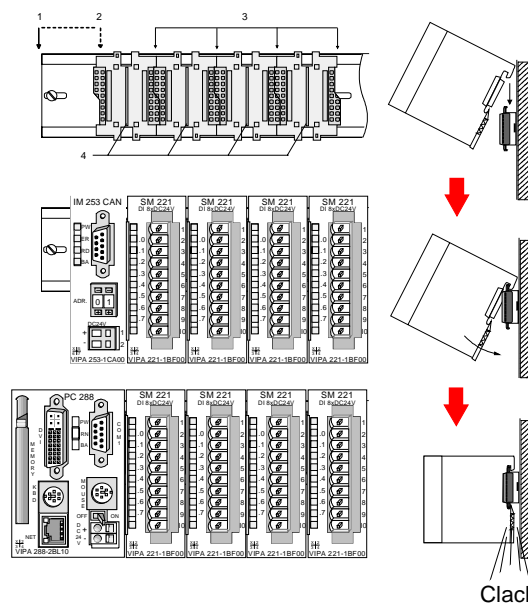
## General description System 200V

### Structure/ dimensions

- Standard 35mm DIN rail
- Peripheral modules with recessed labelling
- Dimensions of the basic enclosure:
  - 1tier width: (HxWxD) in mm: 76x25.4x76 in inches: 3x1x3
  - 2tier width: (HxWxD) in mm: 76x50.8x76 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

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

The chapter "Assembly and installation guidelines" contains an overview over the current consumptions.

### Reliability

- Wiring by means of spring pressure connections (CageClamps) at the front-facing connector, core cross-section 0.08...2.5mm<sup>2</sup> or 1.5 mm<sup>2</sup> (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

## ISO/OSI reference model

### Outline

The ISO/OSI reference model is based on a proposal that was developed by the International Standards Organization (ISO). This represents the first step towards an international standard for the different protocols. It is referred to as the ISO-OSI layer model. OSI is the abbreviation for **O**pen **S**ystem **I**nterconnection, the communication between open systems. The ISO/OSI reference model does not represent a network architecture as it does not define the services and protocols used by the different layers. The model simply specifies the tasks that the different layers must perform.

All current communication systems are based on the ISO/OSI reference model which is defined by the ISO 7498 standard. The reference model structures communication systems into 7 layers that cover different communication tasks. In this manner the complexity of the communication between different systems is divided amongst different layers to simplify the task.

The following layers have been defined:

Layer	Function
Layer 7	Application Layer
Layer 6	Presentation Layer
Layer 5	Session Layer
Layer 4	Transport Layer
Layer 3	Network Layer
Layer 2	Data Link Layer
Layer 1	Physical Layer

Depending on the complexity and the requirements of the communication mechanisms a communication system may use a subset of these layers.

INTERBUS-S and Profibus for instance only use layers 1 and 2. The following pages give a short description of the layers.

**Layers****Layer 1** Bit communication layer (physical layer)

The bit communication layer (physical layer) is concerned with the transfer of data bits via the communication channel. This layer is therefore responsible for the mechanical, electrical and the procedural interfaces and the physical communication medium located below the bit communication layer:

- Which voltage represents a logical 0 or a 1.
- The minimum time that the voltage be present to be recognized as a bit.
- The pin assignment of the respective interface.

**Layer 2** Security layer (data link layer)

This layer performs error-checking functions for bit strings transferred between two communicating partners. This includes the recognition and correction or flagging of communication errors and flow control functions.

The security layer (data link layer) converts raw communication data into a sequence of frames. This is where frame limits are inserted on the transmitting side and where the receiving side detects them. These limits consist of special bit patterns that are inserted at the beginning and at the end of every frame. The security layer often also incorporates flow control and error detection functions.

The data security layer is divided into two sub-levels, the LLC and the MAC level.

The MAC (**M**edia **A**ccess **C**ontrol) is the lower level and controls how senders are sharing a single transmit channel.

The LLC (**L**ogical **L**ink **C**ontrol) is the upper level that establishes the connection for transferring the data frames from one device into the other.

**Layer 3** Network layer

The network layer is an agency layer.

Business of this layer is to control the exchange of binary data between stations that are not directly connected. It is responsible for the logical connections of layer 2 communication. Layer 3 supports the identification of the single network addresses and the establishing and disconnecting of logical communication channels.

Additionally, layer 3 manages the prior transfer of data and the error processing of data packets.

**Layer 4** Transport layer

Layer 4 connects the network structures with the structures of the higher levels by dividing the messages of higher layers into segments and pass them on to the network layer. Hereby, the transport layer converts the transport addresses into network addresses.

Common transport protocols are: TCP, SPX, NWLink and NetBEUI.

**Layers  
continued...****Layer 5** Session layer

The session layer is also called the communication control layer. It relieves the communication between service deliverer and the requestor by establishing and holding the connection if the transport system has a short time fail out.

At this layer, logical users may communicate via several connections at the same time. If the transport system fails, a new connection is established if needed.

Additionally this layer provides methods for control and synchronization tasks.

**Layer 6** Presentation layer

This layer manages the presentation of the messages, when different network systems are using different representations of data.

Layer 6 converts the data into a format that is acceptable for both communication partners.

Here compression/decompression and encrypting/decrypting tasks are processed.

This layer is also called interpreter. A typical use of this layer is the terminal emulation.

**Layer 7** Application layer

The application layer is the link between the user application and the network. The tasks of the application layer include the network services like file, print, message, data base and application services as well as the according rules.

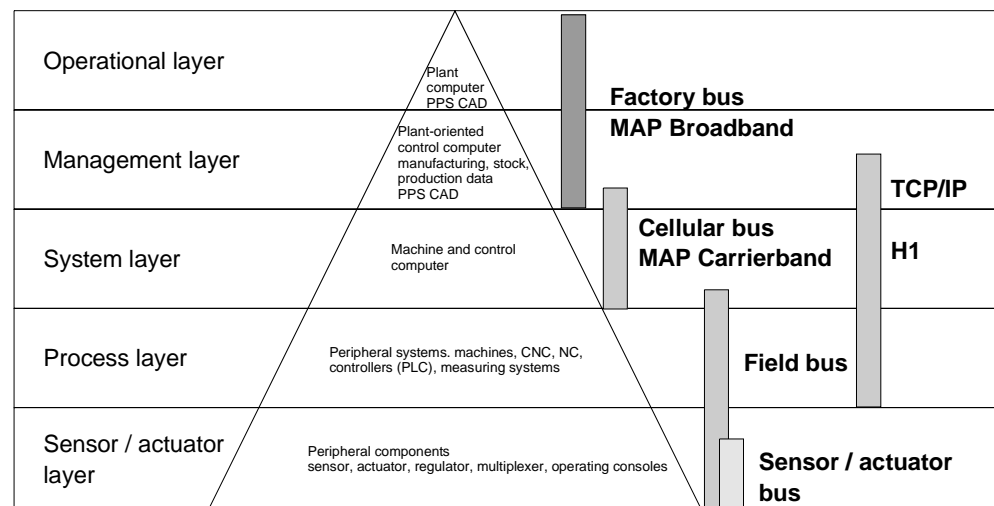
This layer is composed from a series of protocols that are permanently expanded following the increasing needs of the user.



## Communication layers employed by automation systems

The flow of information in a company presents a vast spectrum of requirements that must be met by the communication systems. Depending on the area of business the bus system or LAN must support a different number of users, different volumes of data must be transferred and the intervals between transfers may vary, etc.

It is for this reason that different bus systems are employed depending on the respective task. These may be subdivided into different classes. The following model depicts the relationship between the different bus systems and the hierarchical structures of a company:



It is common that very large volumes of data are transferred on the operational level that are not subject to timing restrictions. However, on the lowest level, i.e. the sensor / actuator level, an efficient transfer of rather small data volumes is essential. In addition, the bus system must often meet real-time requirements on the sensor / actuator level.



## Chapter 2 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.

Below follows a description of:

- System overview of the Profibus modules that are available from VIPA
- The principles of Profibus-DP
- Construction and project engineering of the Profibus masters IM 208DP
- Construction and project engineering of the Profibus slaves IM 253DP
- Sample projects
- Technical data

### Content

Topic	Page
<b>Chapter 2 Profibus-DP</b> .....	<b>2-1</b>
System overview .....	2-2
Principles .....	2-5
IM 208DP - Master with RS485 - Construction .....	2-10
IM 208DP - Master with RS485 - Deployment at CPU 21x .....	2-13
IM 208DP - Master with RS485 - Project engineering .....	2-14
IM 208DP - Master with RS485 - Overall-Reset .....	2-22
IM 208DPO - Master with FO link - Construction .....	2-23
IM 208DPO - Master with FO link - Deployment at CPU 21x .....	2-26
IM 208DP - Master with FO link - Project engineering .....	2-27
IM 253DP - Slave (Standard) - Construction .....	2-33
IM 253DPR - Slave (redundant) - Construction .....	2-36
IM 253DP, DO 24xDC 24V - Construction .....	2-39
IM 253DP - Slave - Block diagram .....	2-43
IM 253DP - Slave - Project engineering .....	2-44
IM 253DP - Slave - Parameters .....	2-46
IM 253DP - Slave - Diagnostic functions .....	2-47
Installation guidelines .....	2-54
Commissioning .....	2-64
Using the diagnostic LEDs .....	2-65
Sample projects for Profibus communication .....	2-66
Technical data .....	2-74

## System overview

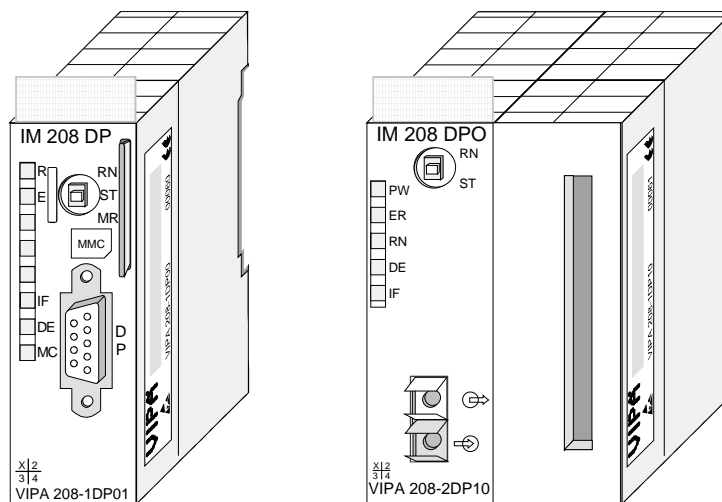
### System 200V Profibus-DP modules

Most System 200V Profibus modules 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
- Profibus-DP slave combination modules
- CPU 21xDP - CPU 21x for S7 from Siemens with integrated Profibus-DP slave (refer to manual HB103)
- 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
- Project design by means of COM Profibus of Siemens is possible
- Project-related data is saved in the internal Flash-ROM or stored on a Flash-Memory card.

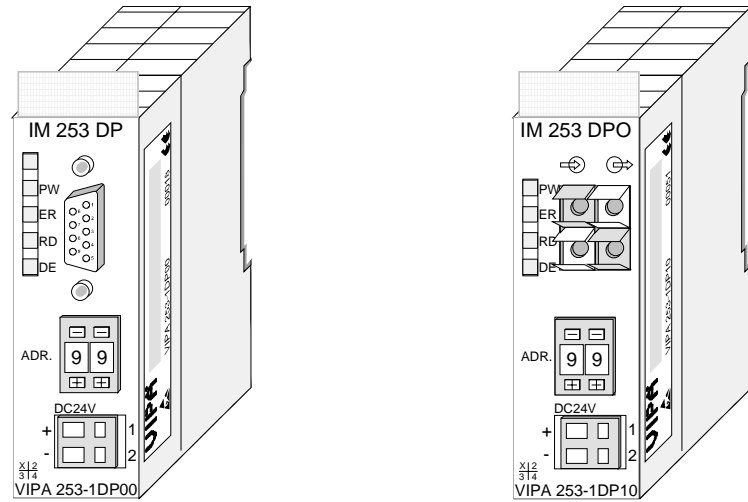


### Order data DP master

Type	Order number	Description
IM 208DP	VIPA 208-1DP01	Profibus-DP master with RS485
IM 208DPO	VIPA 208-2DP10	Profibus-DP master with FO connector

**Profibus-DP slaves (standard)**

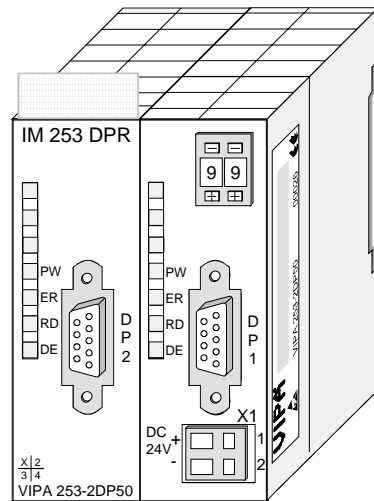
- Version with RS485 interface or fiber optic connectors
- Online diagnostic protocol with time stamp



**Order data**

Type	Order number	Description
IM 253DP	VIPA 253-1DP00	Profibus-DP slave
IM 253DPO	VIPA 253-1DP10	Profibus-DP slave with FO connector

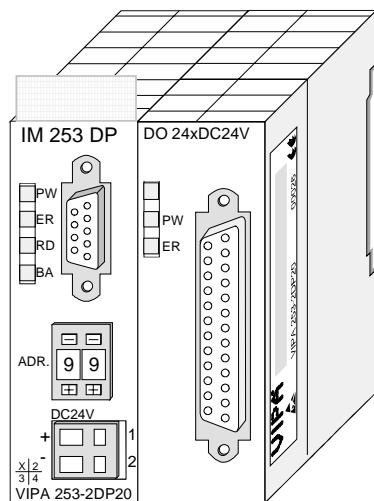
**Profibus-DPR slave (redundant)**



**Order data**

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

**Profibus-DP slave  
(combi modules)**



**Order data**

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

## Principles

### 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.

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

### Profibus-DP

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

The data transfer referred to as "Data Exchange" is cyclical. During one bus cycle, the master reads input values from the slaves and writes output information to the slave.

### 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.

---

**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.

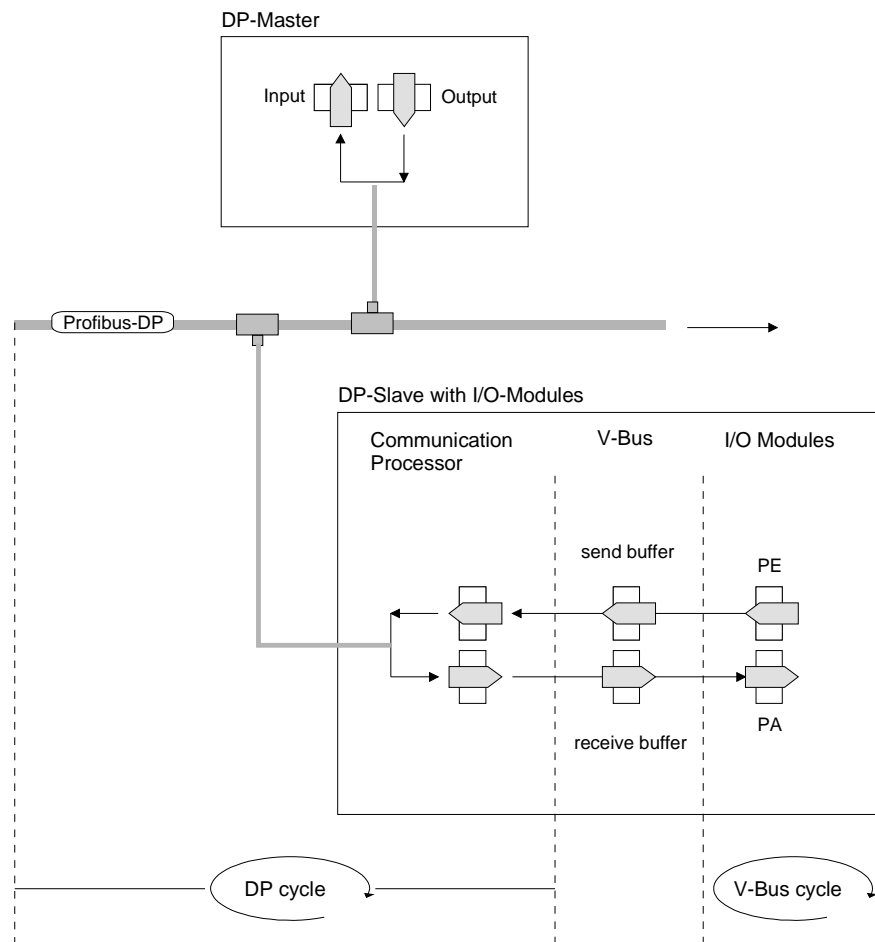
The master handles the transfer of application related data independently and automatically. You can, however, also send new configuration settings to a bus coupler.

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 transfer operation

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



PI: process image of the inputs

PO: 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 PI and all the output data from the PO in the output modules. When the data has been saved the PI is transferred into the "send buffer" and the contents of the "receive buffer" is transferred into PO.

### DP cycle

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

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

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

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!**

When the V-bus cycle time exceeds the DP cycle time the RUN-LED on the VIPA Profibus slave is extinguished.

This function is supported as of hardware revision level 6.

**Data consistency**

The VIPA Profibus-DP masters provide "word-consistency"!

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

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

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

**Restrictions**

- 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.

**Diagnostics**

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.

---

**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 "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 unique number in the bus system between 0 and 125. 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.

**GSD-file**

For configuration purposes you receive a GSD-file containing the performance specifications of VIPA components.

The structure, contents and coding of the GSD-file are defined by the Profibus user organization (PNO) and are available from this organization.

**The GSD-file for VIPA Profibus-DP slaves is:                   VIPAO550.GSD**

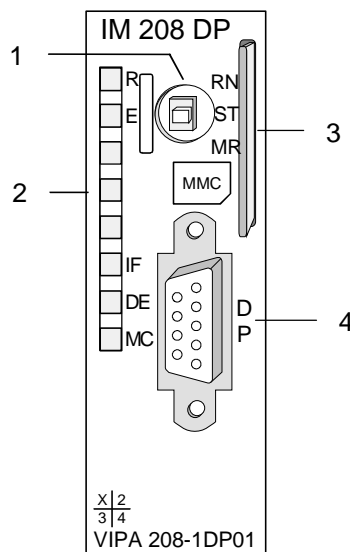
Install this GSD-file into your configuration tool. You can obtain more detailed information on the installation of GSD-files from the manual supplied with your configuration tool.

## IM 208DP - Master with RS485 - Construction

### Properties

- Class 1 Profibus-DP master
- 125 DP slaves connectable to one DP master
- Inserts the data areas of the slaves located on the V-bus into the addressing area of the CPU 2xx
- Project engineering by means of VIPAs WinNCS or Siemens 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

### Components

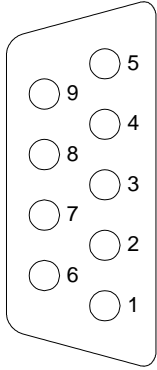
#### LEDs

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 RN+DE are on the status of the Master is "operate". It is communicating with the slaves.
E	red	On at slave failure (ERROR).
IF	red	Initialization error for bad parameterization
DE	yellow	DE (Data exchange) indicates Profibus communication activity.
MC	yellow	Blinks at reading the parameters from MMC. Is on at wrong parameterization.

**RS485 interface**

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
4	CNTR-P
5	GND
6	5V (max. 70mA)
7	n.c.
8	RxD/TxD-N
9	n.c.

**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 detailed information about those options is to find further below.

**MMC as external storage medium**

The VIPA MMC memory card is employed as an external storage medium. You can transfer your project related data from the internal Flash-ROM into this memory card by means of the command *Copy RAM to ROM* of the Siemens Hardware manager.

The MMC memory card is available from VIPA with the order no.: VIPA 953-0KX00.

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 IM 208 interface is powered on. The configuration data is read from the memory card, the validity is verified and the data is stored in the internal RAM of the IM 208.

The master will change automatically to RUN mode when the operating mode lever is in position RUN and the parameters are valid. In RUN mode the LEDs R, DE and E are turned on. The E-LED is extinguished when all the configured slaves are available via data exchange.

*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 IM 208 interface.

*STOP → RUN*

In the RN position the master will re-boot: configuration data and bus parameters are retrieved from the Flash-ROM and saved into the internal RAM of the IM 208. **An existing hardware configuration is deleted by the boot procedure of the DP master.**

Next, the communication link to the slaves is established. At this time only the R-LED will be on. Once communication has been established by means of valid bus parameters the IM 208 will change to RUN mode. The master interface displays this status by means of the LEDs R and DE.

The IM 208 will remain in the STOP mode and display a configuration error by means of the IF-LED if the parameters are bad or if the memory card was not inserted. The interface will then be active on the bus using the following default bus parameters:

**Default bus parameters: Address:1, Communication rate:1.5MBaud.**

*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 IM 208 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 with RS485 - Deployment at CPU 21x

### Communication

Via the IM 208 master modules you may connect up to 125 Profibus-DP slaves 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. "Befehlsausgabesperre" = 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.



### Note!

For the deployment with a CPU 24x, the firmware version of the DP master has to be set to level V1.0.7.

## IM 208DP - Master with RS485 - Project engineering

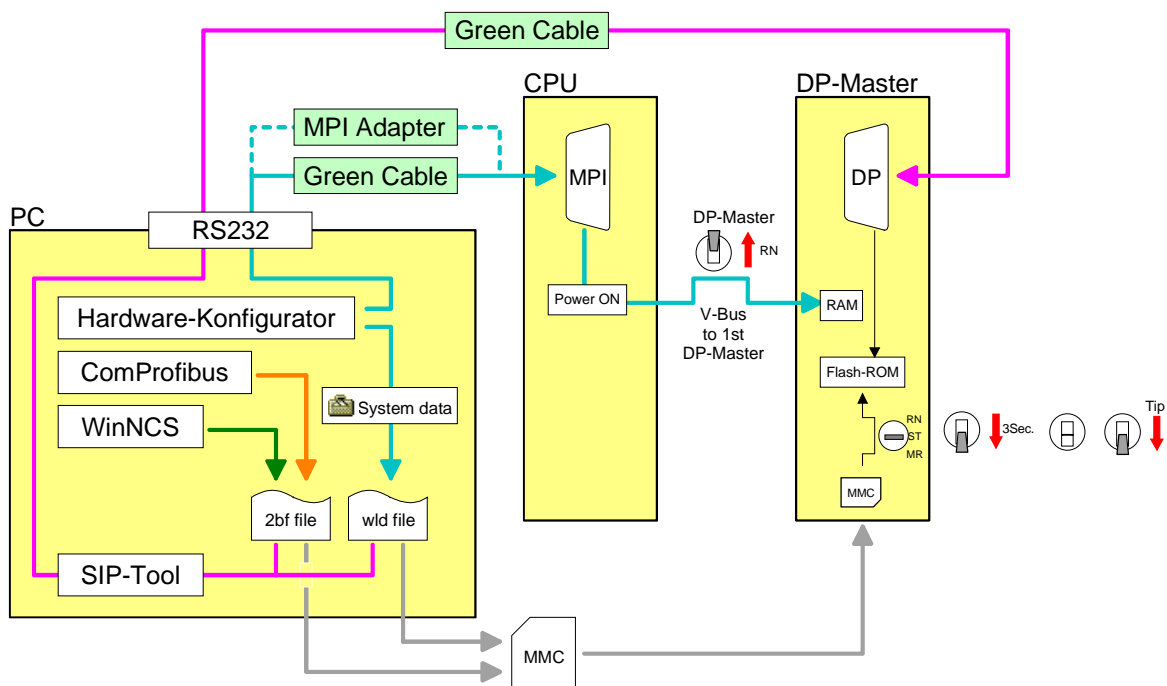
### General

You may configure the DP master in different ways:

- Project engineering in the hardware configurator from Siemens and transfer via the system bus as hardware configuration. This allows you only to configure the 1<sup>st</sup> master (IM 208 res. CPU 21xDPM).

**Please regard that the operating mode lever of the DP master has to be in the position RN for accepting the configuration via system bus.**

- Project engineering in the hardware configurator from Siemens and export of a wld-file.
- Project engineering via WinNCS from VIPA res. ComProfibus from Siemens and export of a 2bf-file.



### Required firmware versions

DP master and CPU should have a firmware version V.300 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 as HW configuration**

In the hardware configurator from Siemens you project your PLC system together with the DP master. You transfer this "hardware configuration" via MPI into the CPU. At Power ON, the configuration data is transferred to the DP master if the operating mode lever is in position RN.



**Note!**

Please make sure, that the operating mode lever of the DP master is in RN position. Otherwise, a STOP-RUN switch causes the master to reboot and the project is deleted.

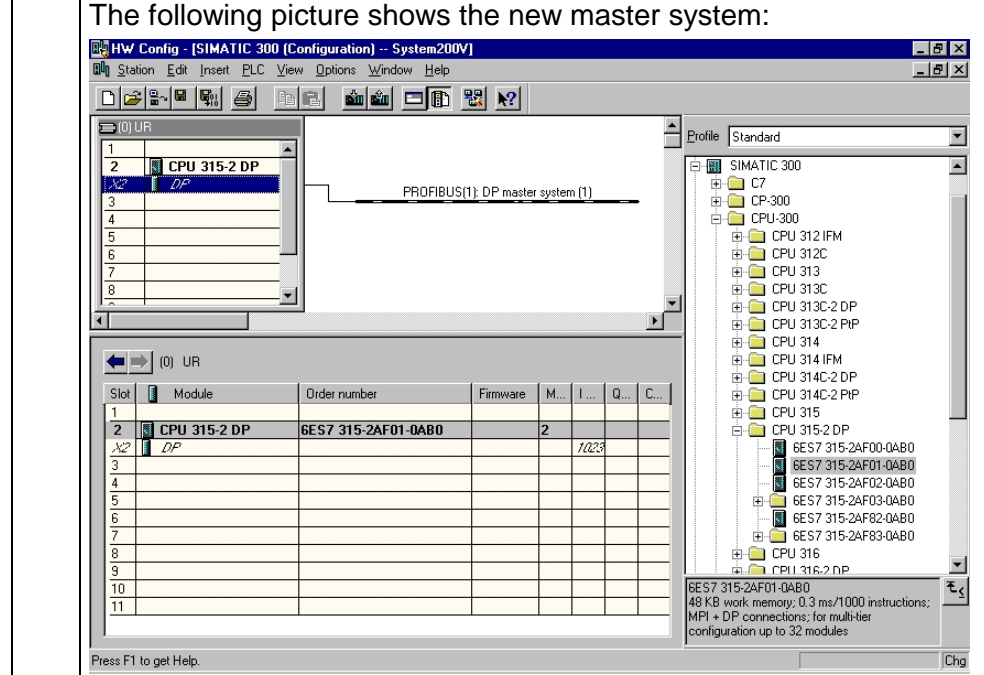
Please regard also that this allows you only to configure the 1<sup>st</sup> master in the system! Additional DP master have to be exported as wld-file res. 2bf-file.

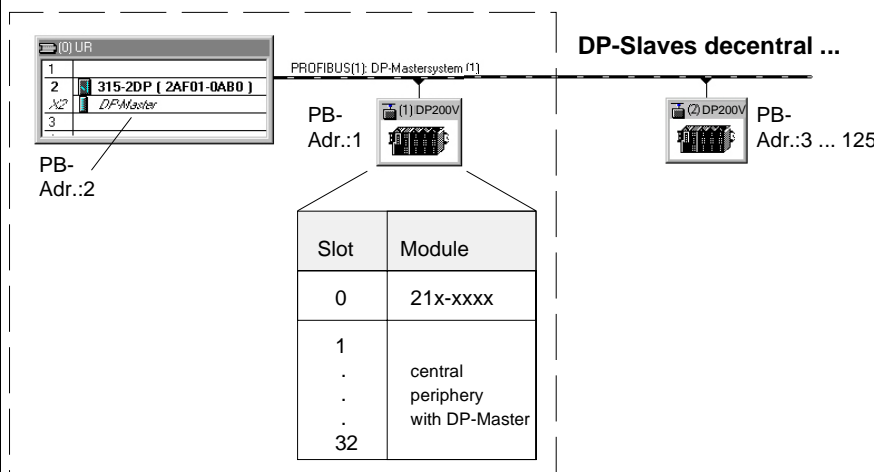
**Approach with hardware configuration**


The steps you have to follow with the hardware configurator from Siemens shall be shortly described here:

1. Create a new project System 300 and add a profile rail from the hardware catalog.
2. Insert the CPU 315-2DP. This is to find under:  
*Simatic300 > CPU-300 > CPU315-2DP > 6ES7 315-2AF01-0AB0*
3. Assign a Profibus address 2 or higher to your master.
4. Click at DP, select the operating mode "DP master" under *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:



6. To be compatible to the STEP<sup>®</sup>7 projecting tool from Siemens, the System 200V CPU has to be included explicitly.  
 For this, you add a System "VIPA\_DP200V\_2" to the subnet. This system is to find in the hardware catalog under:  
*PROFIBUS DP > Additional field devices > IO > VIPA\_System\_200V > VIPA\_DP200V\_2.*  
 Assign the Profibus address 1 to this slave.  
  
 Set the according CPU 21x from VIPA at plug-in location 0 by choosing it in the hardware catalog under *VIPA\_CPU21x.*  
**The plug-in location 0 is mandatory!**
7. For including the modules connected to the VIPA-Bus, you drag and drop the according System 200V modules from the hardware catalog under *VIPA\_DP200V\_2* to the plug-in locations following the CPU. Start with plug-in location 1. The same is to do for the DP master.
8. For projecting DP slaves connected to the DP master, execute 6. (Project engineering of 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 > 3 to the slave.  
  
**CPU 21x central**  


Slot	Module
0	21x-xxxx
1	central periphery with DP-Master
.	
.	
32	
9. Click on  (save and translate).

**Transfer variants**

Depending on the used firmware at CPU and DP master, you have the following transfer possibilities:

1. Transfer via MPI (only for 1<sup>st</sup> master at the bus)  
Your DP master project is transferred to the CPU together with the PLC program. The CPU transferees the DP master project automatically to the 1<sup>st</sup> master at the bus system (IM 208DP or CPU 21xDPM) at Power ON.
2. Export of the project as wld-file to MMC  
Export your project as wld-file and transfer it to a MMC. The MMC is to plug in the according DP master.
3. Export of the project as wld-file and data transfer via Green Cable deploying the SIP-Tool from VIPA.

**to 1.**  
**Transfer via MPI**

10.	<p>Connect your CPU 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 in the Principles at the beginning of this manual! At deployment of the Green Cable from VIPA, the MPI interface has to be configured (PC Adapter MPI, 38400Baud).</p>
11.	Switch your DP master to RUN.
12.	Switch on the power supply of the CPU.
13.	Transfer your project into the CPU with <b>PLC</b> > <i>Load to module</i> in the hardware configurator from Siemens.
14.	For additional saving of your project on a MMC, you plug a MMC in the CPU slot and transfer the project via <b>PLC</b> > <i>Copy RAM to ROM</i> . 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.

Now the project engineering is completed.

**Note!**

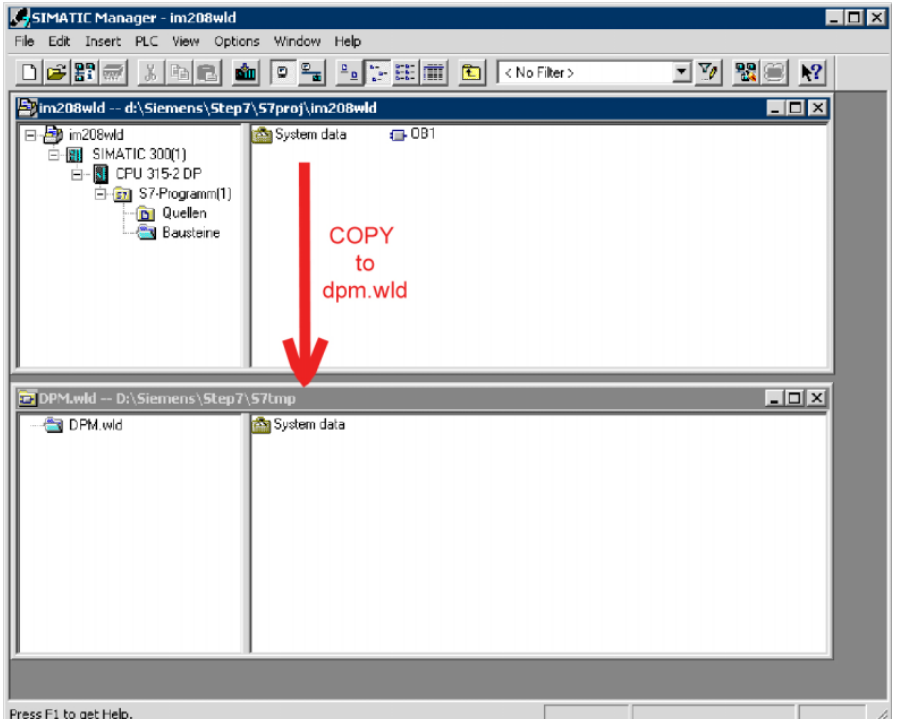
Please regard also that this allows you only to configure the 1<sup>st</sup> master in the system! Additional DP masters have to be exported as wld-file and loaded to MMC.

to 2.  
Export as dpm.wld  
to MMC

To project additional DP masters, you export your project to a MMC by creating a wld-file. The MMC is then plugged in the according DP master. By using the operating mode lever, you may transfer your project from the MMC into the Flash-ROM of the master.

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

### Approach

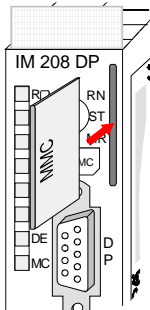
10..	Create with <b>File &gt; Memory Card File &gt; New ...</b> a new wld-file. This need to have the file name <b>dpm.wld</b> to be recognized from the Profibus master. → This file is additionally shown to the configuration window.
11.	Go into your project into the directory <i>modules</i> and copy the directory "System data" into the created dpm.wld-file.  The screenshot shows two windows in SIMATIC Manager. The top window, titled 'im208wld', displays a project tree with folders for 'SIMATIC 300(1)', 'CPU 315-2 DP', 'S7-Programm(1)', 'Quellen', and 'Bausteine'. A 'System data' folder is highlighted in the right pane. A red arrow points from this folder down to the 'DPM.wld' window below. The bottom window, titled 'DPM.wld', shows a 'System data' folder in its right pane, indicating it has been copied. The text 'COPY to dpm.wld' is written in red next to the arrow.



### Note!

If an already existing "System data" directory shall be overwritten, you first have to delete that.

**to 2. continued**  
**Transfer data from**  
**MMC to Flash-ROM**



12.	Transfer the wld-file to the MMC by means of a MMC reading device.
13.	Plug-in the MMC memory module into your IM 208DP master
14.	Turn on the power supply for the System 200V.
15.	Hold the operating mode lever of the Profibus master in position MR until the blinking MC-LED switches to permanent on.
16.	<p>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.</p> <p>At successful data transfer, the green R-LED blinks 3 times.</p> <p>At error, the red E-LED blinks 3 times.</p> <p style="text-align: center;">Transfer → OK Error</p>
17.	Now you may release the MMC again..
18.	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.



**Note!**

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

Now the project engineering is completed.

**to 3.  
Export as dpm.wld  
Transfer via  
SIP-Tool from VIPA**

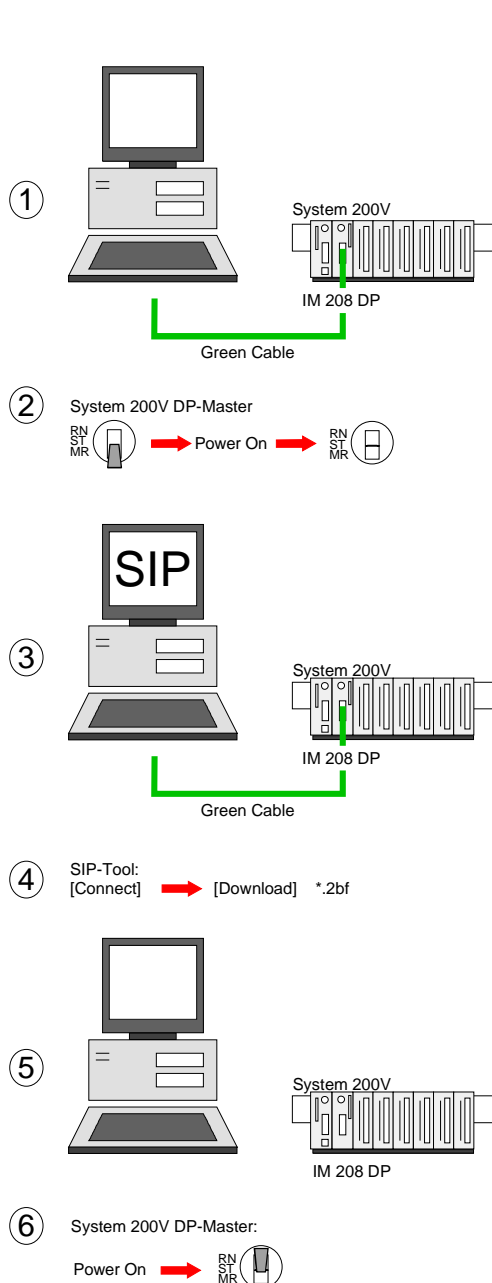
The SIP-Tool is a transfer tool. It is supplied together with WinNCS from VIPA. It allows you to deploy the Green Cable from VIPA to transfer your project as wld- res. 2bf-file into the master serial via the Profibus interface. The transferred project is stored in the internal Flash-ROM of the DP master.

The Green Cable is available at VIPA with the order no. VIPA 950-0KB00.



**Attention!**

Please regard the instruction for deploying the Green Cable in the Principles at the beginning of this manual!







12.	Disconnect the Profibus plug from the DP master.
13.	Connect the "Green Cable" to the serial interface of your PC and to the Profibus interface of the IM 208DP master.
14.	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.
15.	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.
16.	Click [Download], select your dpm.2bf- res. dpm.wld-file and transfer this file into the DP master
17.	Terminate the connection and the SIP tool when the data has been transferred.
18.	Disconnect the "Green Cable" from the master.
19.	Turn off the power supply of your master.
20.	Connect the master to the Profibus network and turn the power supply on again.
21.	Change the operating mode of the master to RUN. → 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.

## Configuration under WinNCS

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.

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 <b>File</b> > <i>create/open</i> .
2.	If you have not yet done so, use  to insert a <b>Profibus function group</b> into the network window and click [Accept] in the parameter box.
3.	Use  to insert a <b>Profibus host/master</b> into the network window and specify the Profibus address of your master in the parameter window.
4.	Insert a <b>Profibus slave</b> 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].  <b>Please take care that the automatic address allocation does not cause conflicts with the local periphery!</b>  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 baudrate 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.

## IM 208DP - Master with RS485 - 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**

1.	Turn on the power supply of the System 200V.
2.	<p>Push the operating mode lever of the master module in position MR. Hold it for app. 9s</p> <p>→ 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.</p>
3.	<p>Release the lever and tip it within 3s once more in pos. MR.</p> <p>→ 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.</p> <div style="text-align: center;"> </div> <p>As soon as you switch the master to RUN, this boots and starts with its default parameters at the bus.</p> <p><b>Default parameter: Address: 1, Transfer rate: 1,5Mbaud</b></p>

**Project engineering via CPU after power-on**

If there is a valid hardware configuration in the CPU, this is automatically transferred via backplane bus into the RAM of the master after power-on.

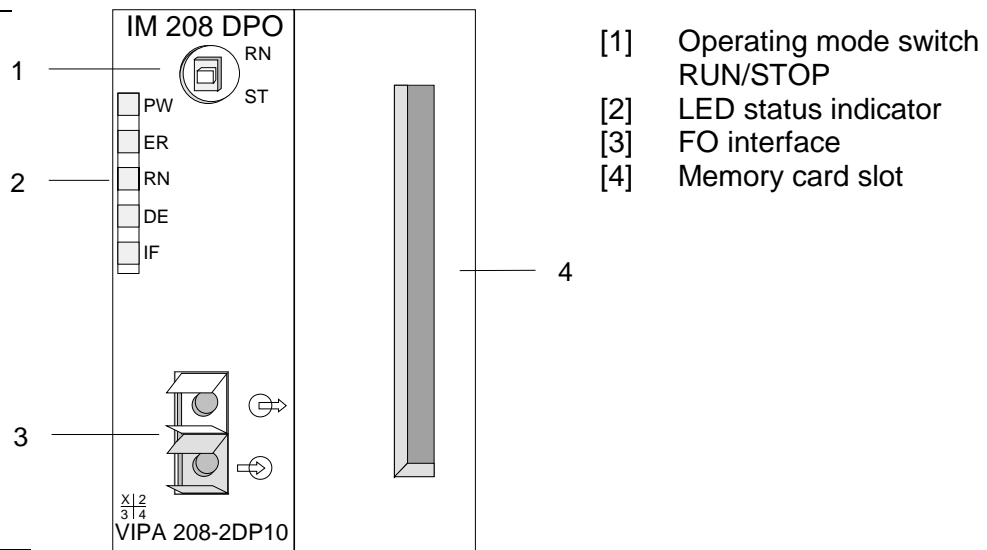


## IM 208DPO - Master with FO link - Construction

### Properties

- Class 1 Profibus-DP master
- Max. 16 DPO slaves can be connected to one DPO master
- Maps the data areas of the slaves into the addressing area of the CPU 24x via the V-Bus
- Project configuration by means of VIPA WinNCS or Siemens ComProfibus
- Diagnostic facilities

### Front view IM 208DPO

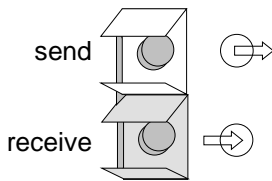


### Components

### LEDs

The master 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 significance of the different colors of the diagnostic LEDs.

Label	Color	Description
PW	yellow	Indicates that the supply voltage is available on the backplane bus.
ER	red	On when a slave has failed (ERROR).
RN	green	When only the RN-LED is on, then the master status is RUN. The slaves are being accessed and the outputs are 0 ("Clear" state). If both RN+DE are on the status of the master is OPERATE. It is communicating with the slaves.
DE	yellow	DE (Data exchange) indicates Profibus communication activity.
IF	red	Initialization error for bad parameterization.

**FO link interface**

This socket is provided for the optical waveguide between your Profibus coupler and the Profibus. The figure shows the connections for this interface.

**Power supply**

The Profibus master receives supply voltage via the backplane bus.

**Operating mode lever**

The operating mode lever is used to select between the operating modes STOP (ST) and RUN (RN).

When the operating mode lever is placed in position RN and the parameters are valid, the master changes to RUN mode.

When the operating mode switch is placed in position ST, the master changes to STOP mode. It terminates communication and all outputs are set to 0. An alarm is issued to the system on the next higher level.

This chapter contains a detailed explanation under the heading "Operating modes".

**Flash Memory Card**

You can insert a Flash memory card into this slot to transfer your configurations.

The memory card is available from VIPA under the order no.: VIPA 374-1KH21.

When you are using a PU with a slot for a memory card you can save your project directly into the memory card.

If you are using a PC to configure your projects, you can order an EPROM programming device from VIPA under the order no.: VIPA Multi-Prommer.

*Applications in the IM 208DPO*

When the master receives power while the memory card is inserted or when the operating mode lever is changed from ST to RN, the configuration data and bus parameters are transferred from the memory card into the internal RAM of the DP master.

As soon as the master is in DataExchange (RN and DE are blinking), you may remove the memory card.

You may remove the memory card at any time. Please regard, that at a STOP/RUN transition, the DP master reboots and tries to read from MMC.

---

**Operating modes  
start-up behavior***Power On*

Power is applied to the IM 208DPO interface. Configuration data is retrieved from the memory card, verified, and saved into the internal RAM.

The master will automatically change to RUN mode if the operating mode selector is set to RUN and the parameters are acceptable. In RUN mode the LEDs RN, DE and ER are on. As soon as all configured slaves are available in the data exchange, the ER-LED is extinguished.

*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 FO plug must be removed.

*STOP → RUN*

In the RN position the master will re-boot: configuration data and bus parameters are retrieved from the memory card and saved into the internal RAM of the DP master.

Next, the communication link to the slaves is established. At this time only the RN-LED will be on. Once communication has been established by means of valid bus parameters, the master will change to RUN mode. The master interface displays this status by means of the LEDs RN and DE.

The IM 208DPO will remain in the STOP mode and display a configuration error by means of the IF-LED if the parameters are bad or if the memory card was not inserted. The interface will then be active on the bus using the following default bus parameters:

**Default bus parameters: address: 1, transfer rate: 1.5MBaud.**

*RUN*

In RUN mode the RN- and DE-LEDs are on. In this condition data transfers can take place. If an error should occur, e.g. slave failure, the DP master will indicate the event by means of the ER-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 208DPO - Master with FO link - Deployment at CPU 21x

**Communication** IM 208DPO master modules can be used to connect up to 16 Profibus-DP slaves to a System 200V CPU. The master communicates with the slaves and maps the data areas into the memory area of the CPU via the backplane bus. Input and output data are limited to a maximum of 1024Byte each.

Firmware versions < V3.0.0 allow a maximum of 256Byte for input and output data.

The master automatically fetches the I/O mapping data from all the masters when the CPU is re-started.

**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. "Befehlsausgabesperre" = 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.



### Note!

For the deployment with a CPU 24x, the firmware version of the DP master has to be set to level V1.0.7.

## IM 208DP - Master with FO link - Project engineering

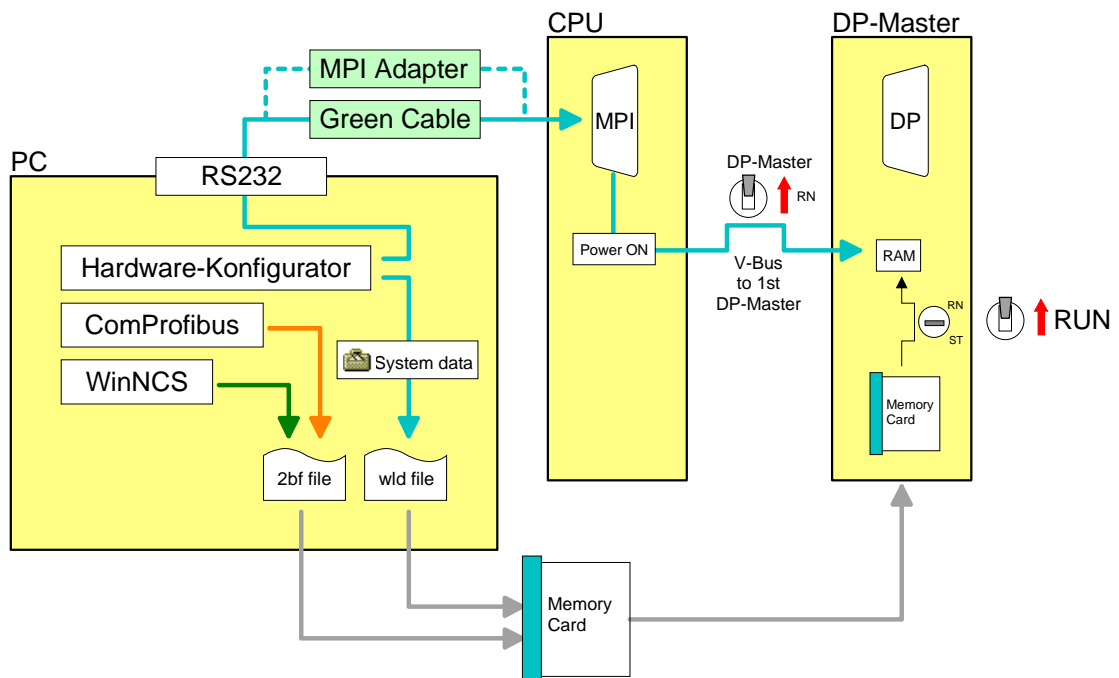
### General

You may configure the DP master in different ways:

- Project engineering in the hardware configurator from Siemens and transfer via the system bus as hardware configuration. This allows you only to configure the 1<sup>st</sup> master (IM 208 res. CPU 21xDPM).

**Please regard that the operating mode lever of the DP master has to be in the position RN for accepting the configuration via system bus.**

- Project engineering in the hardware configurator from Siemens and export of a wld-file.
- Project engineering via WinNCS from VIPA res. ComProfibus from Siemens and export of a 2bf-file.



### 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

**Project engineering as HW configuration**

In the hardware configurator from Siemens you project your PLC system together with the DP master. You transfer this "hardware configuration" via MPI into the CPU. At Power ON, the configuration data is transferred to the DP master if the operating mode lever is in position RN.



**Note!**

Please make sure, that the operating mode lever of the DP master is in RN position. Otherwise, a STOP-RUN switch causes the master to reboot and the project is deleted.

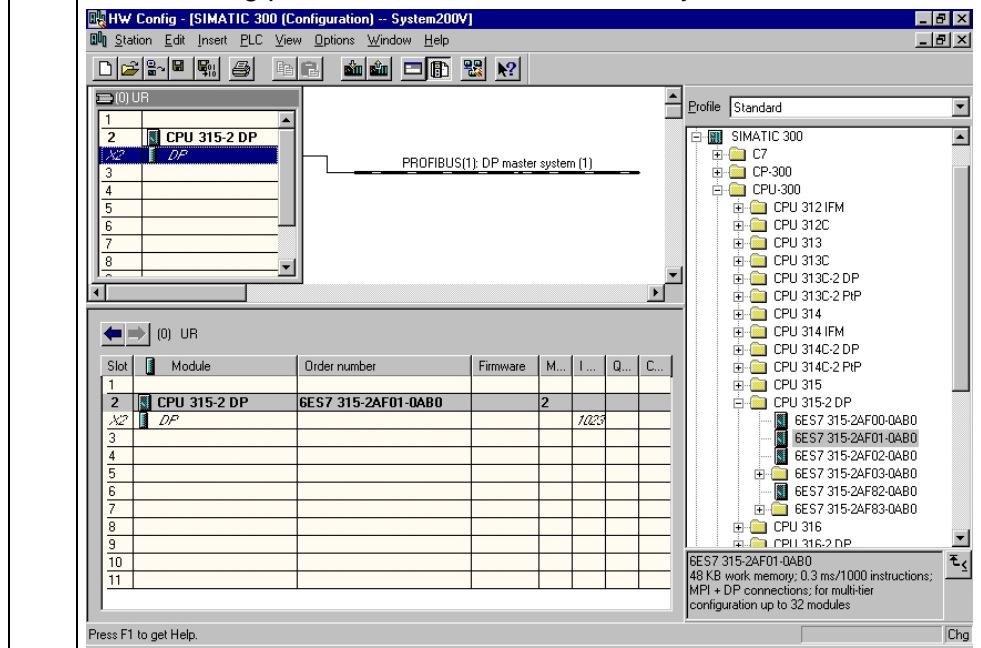
Please regard also that this allows you only to configure the 1<sup>st</sup> master in the system! Additional DP master have to be exported as wld-file res. 2bf-file.

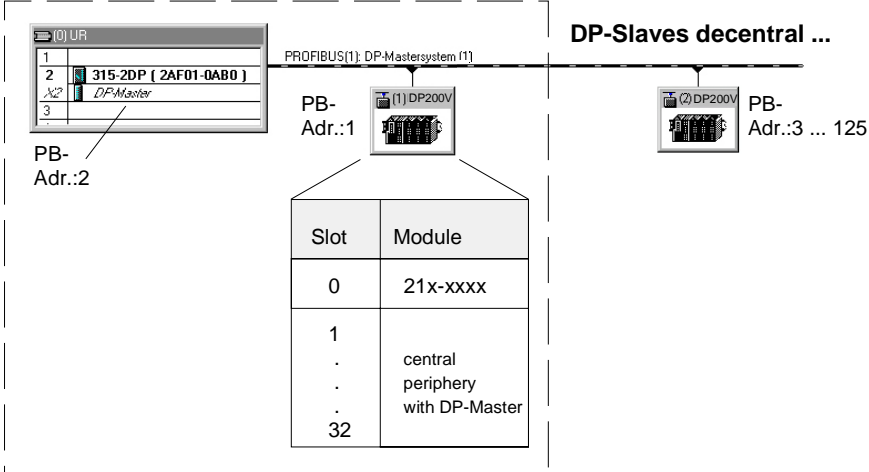

**Approach with hardware configuration**

The steps you have to follow with the hardware configurator from Siemens shall be shortly described here:

1. Create a new project System 300 and add a profile rail from the hardware catalog.
2. Insert the CPU 315-2DP. This is to find under:  
*Simatic300 > CPU-300 > CPU315-2DP > 6ES7 315-2AF01-0AB0*
3. Assign a Profibus address 2 or higher to your master.
4. Click at DP, select the operating mode "DP master" under *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:



6. To be compatible to the STEP<sup>®</sup>7 projecting tool from Siemens, the System 200V CPU has to be included explicitly.  
 For this, you add a System "VIPA\_DP200V\_2" to the subnet. This system is to find in the hardware catalog under:  
*PROFIBUS DP > Additional field devices > IO > VIPA\_System\_200V > VIPA\_DP200V\_2.*  
 Assign the Profibus address 1 to this slave.  
  
 Set the according CPU 21x from VIPA at plug-in location 0 by choosing it in the hardware catalog under *VIPA\_CPU21x.*  
**The plug-in location 0 is mandatory!**
7. For including the modules connected to the VIPA-Bus, you drag and drop the according System 200V modules from the hardware catalog under *VIPA\_DP200V\_2* to the plug-in locations following the CPU. Start with plug-in location 1. The same is to do for the DP master.
8. For projecting DP slaves connected to the DP master, execute 6. (Project engineering of 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 > 3 to the slave.  
**CPU 21x central**
- 
- | Slot | Module                           |
|------|----------------------------------|
| 0    | 21x-xxxx                         |
| 1    | central periphery with DP-Master |
| .    |                                  |
| .    |                                  |
| 32   |                                  |
9. Click on  (save and translate).

**Transfer variants**

Depending on the used firmware at CPU and DP master, you have the following transfer possibilities:

1. Transfer via MPI (only for 1<sup>st</sup> master at the bus)  
Your DP master project is transferred to the CPU together with the PLC program. The CPU transfers the DP master project automatically to the 1<sup>st</sup> master at the bus system (IM 208DP or CPU 21xDPM) at Power on.
2. Export of the project as wld-file to MMC  
Export your project as wld-file and transfer it to a MMC. The MMC is to plug in the according DP master.  
To transfer the project into your System 200V Profibus master, a MMC is required as well as an external EPROM programming device with software. The memory card is available at VIPA under the order no. VIPA 374-1KH21.

**to 1.**  
**Transfer via MPI**

10.	<p>Connect your PU 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 in the Principles at the beginning of this manual! At deployment of the Green Cable from VIPA, the MPI interface has to be configured (PC Adapter MPI, 38400Baud).</p>
11.	Switch on the power supply of the CPU.
12.	Transfer your project into the CPU with <b>PLC</b> > <i>Load to module</i> in the hardware configurator from Siemens.
13.	For additional saving of your project on a MMC, you plug a MMC in the CPU slot and transfer the project via <b>PLC</b> > <i>Copy RAM to ROM</i> . 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.

Now the project engineering is completed.

**Note!**

Please regard also that this allows you only to configure the 1<sup>st</sup> master in the system! Additional DP masters have to be exported as wld-file and loaded to MMC.

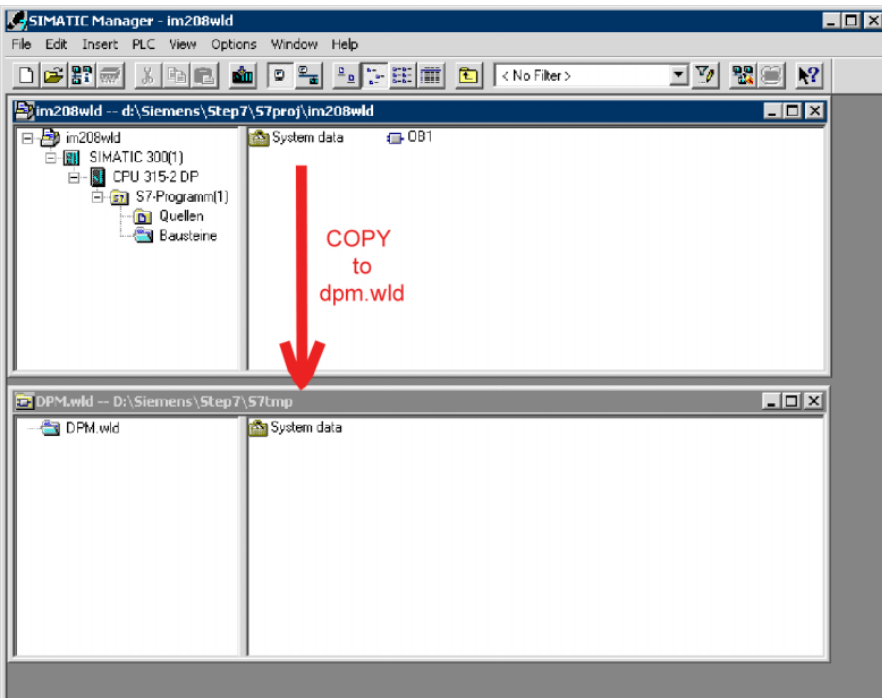


**to 2.  
Export as dpm.wld  
to MMC**

To project additional DP masters, you export your project to a MMC by creating a wld-file. The MMC is then plugged in the according DP master. By using the operating mode lever, you may transfer your project from the MMC into the Flash-ROM of the master.

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

*Approach*

10..	<p>Create with <b>File &gt; Memory Card File &gt; New ...</b> a new wld-file. This need to have the file name <b>dpm.wld</b> to be recognized from the Profibus master. → This file is additionally shown to the configuration window.</p>
11.	<p>Go into your project into the directory <i>modules</i> and copy the directory "System data" into the created dpm.wld-file.</p>  <p>The screenshot shows two windows in SIMATIC Manager. The top window, titled 'im208wld', displays a project tree with 'System data' selected. A red arrow points from this directory to the bottom window, titled 'DPM.wld', which shows 'System data' being copied into the file. The text 'COPY to dpm.wld' is written in red next to the arrow.</p>



**Note!**

If an already existing "System data" directory shall be overwritten, you first have to delete that.






If there are a wld- and a 2bf-file on the MMC, the wld-file has the priority.

Now the project engineering is complete.

**Configuration via WinNCS**

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.

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 <b>File &gt; create/open</b> .
2.	If you have not yet done so, use  to insert a <b>Profibus function group</b> into the network window and click [Accept] in the parameter box.
3.	Use  to insert a <b>Profibus host/master</b> into the network window and specify the Profibus address of your master in the parameter window.
4.	Insert a <b>Profibus slave</b> 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].  <b>Please take care that the automatic address allocation does not cause conflicts with the local periphery!</b>  For intelligent modules like the CP240 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 baudrate 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. Transfer the 2bf-file to the memory card with the help of a programming device.  WinNCS supports the data transfer via a master-PC-plug-in card from Softing. This card allows you to establish a master-master-communication via Profibus and to transfer your 2bf-file in both directions via the module transfer functions  .



**Note!**

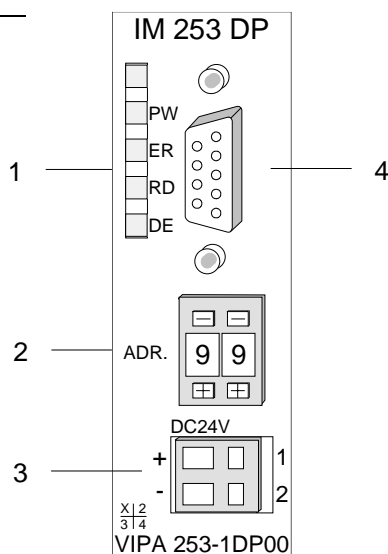
For the IM 208 DP master is configured like the IM 308-C from Siemens, you may configure the VIPA module also as IM 308-C under "ComProfibus" from Siemens and export it as 2bf-file.

## IM 253DP - Slave (Standard) - Construction

### Properties

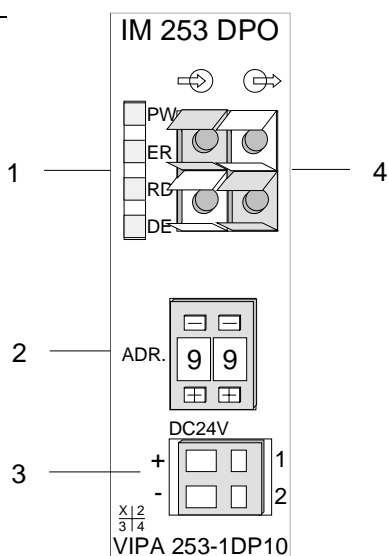
- Profibus-DP slave for max. 32 peripheral modules (max. 16 analog modules)
- Max. 152Byte input data and 152Byte output data
- Internal diagnostic protocol with a time stamp
- Integrated DC 24V power supply for the peripheral modules (3.5A max.)
- Supports all Profibus data transfer rates

### Front view 253-1DP00



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

### Front view 253-1DP10



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

**Components**

**LEDs**

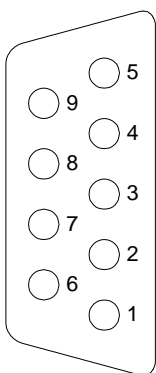
The Profibus slave modules carry a number of LEDs that are available for diagnostic purposes on the bus and for displaying the local status. The following table explains the different colors of the diagnostic LEDs.

Label	Color	Description
PW	yellow	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).
RD	green	Blinks in time with RD when the configuration is bad. 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).
DE	yellow	Blinks in time with ER when the configuration is bad DE (Data exchange) indicates Profibus communication activity.

**RS485 interface**

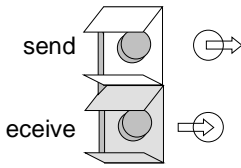
A 9pin socket is provided for the RS485 interface between your Profibus slave and the Profibus.

The following diagram shows the pin assignment for this interface:



Pin	Assignment
1	shield
2	n.c.
3	RxD/TxD-P
4	CNTR-P
5	GND
6	5V (max. 70mA)
7	n.c.
8	RxD/TxD-N
9	n.c.

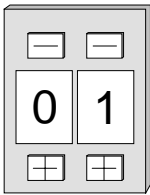
### FO interface



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 by 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 power supply is protected against reverse polarity.

Profibus and backplane bus are galvanically isolated from each other.



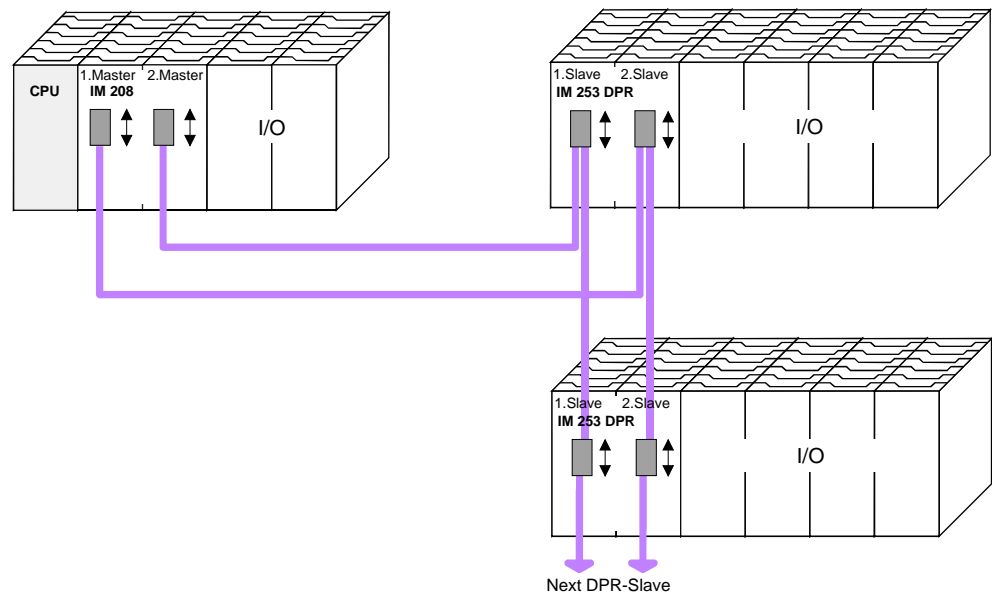
### Attention!

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

## IM 253DPR - Slave (redundant) - Construction

**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.



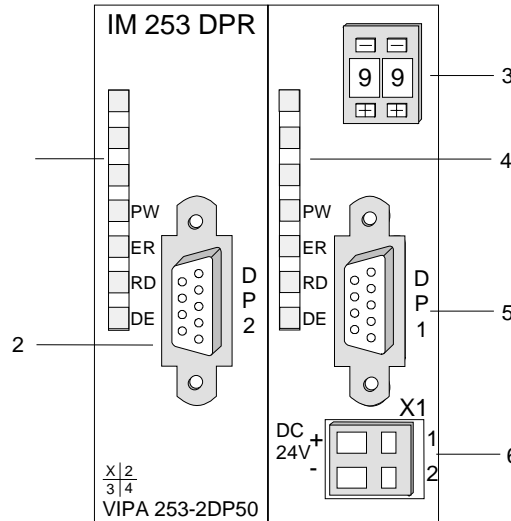
### Requirements for the deployment

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. 3A )
- 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**

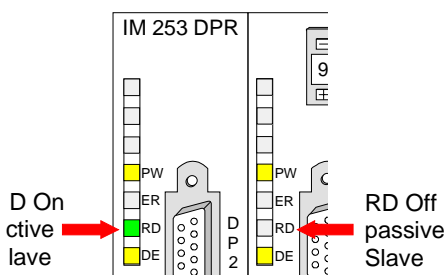
**LEDs**

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	yellow	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	yellow	DE (Data exchange) indicates Profibus communication activity.

**LEDs at redundant operation**

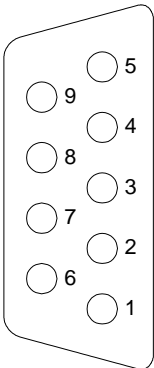
During redundant operation the active slave shows its activity via the green RD-LED, at the passive slave the RD-LED is off. At both slaves the PW- and the DE-LED are on.



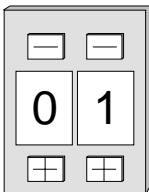
RD	DE	Description
on	on	active slave (write and read)
off	on	passive backup slave (read)

**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
4	CNTR-P
5	GND
6	5V (max. 70mA)
7	n.c.
8	RxD/TxD-N
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 by 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 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 253DP, DO 24xDC 24V - Construction

### General

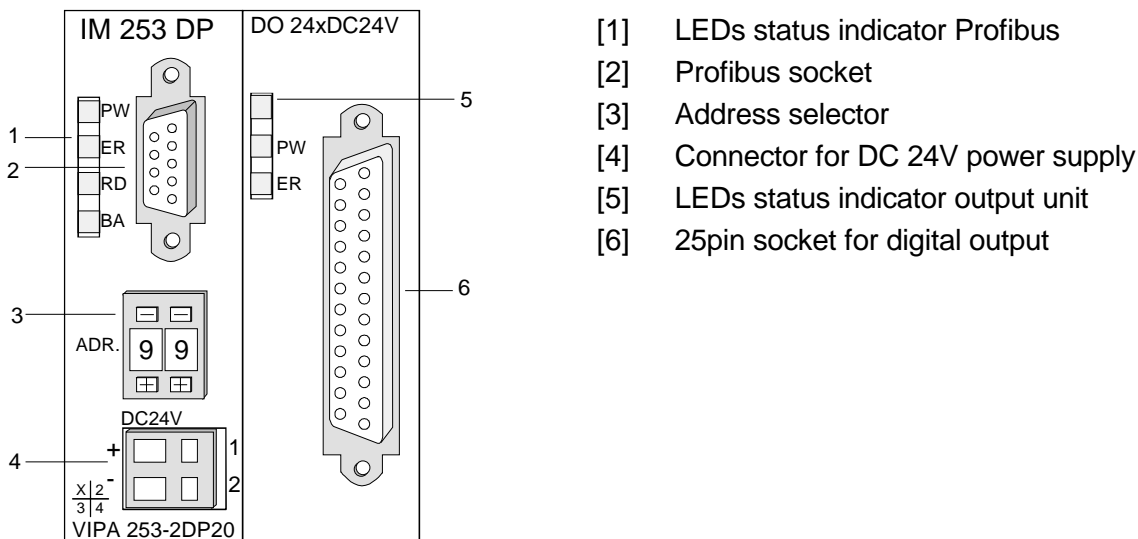
This module consists of a Profibus slave with an integrated 24port output unit. The 24 output channels are controlled directly via the Profibus. The output channels are capable of a maximum load current of 1A each. The total output current must never exceed 4A. The outputs are dc-coupled.

### 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



### 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**

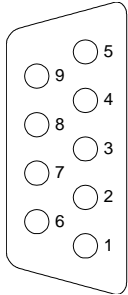
The digital output section is provided with 2 LEDs with the following function:

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.

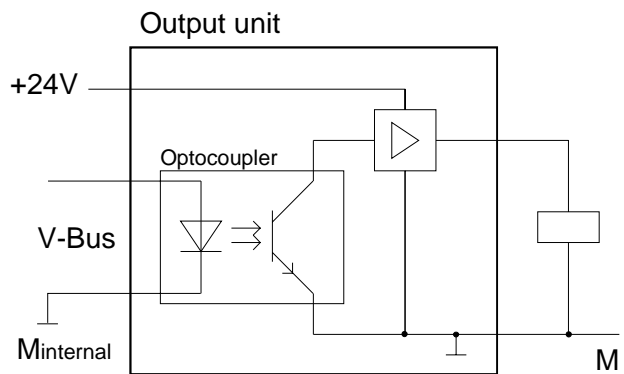
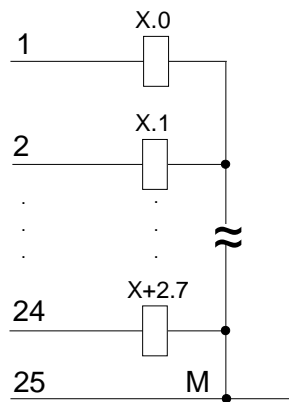
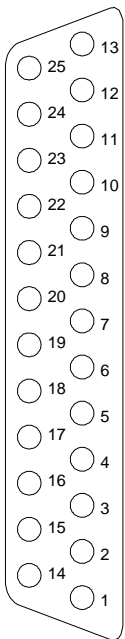
The following diagram shows the pin assignment for this interface:

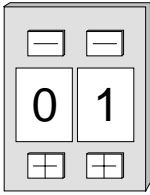


Pin	Assignment
1	shield
2	n.c.
3	RxD/TxD-P
4	CNTR-P
5	GND
6	5V (max. 70mA)
7	n.c.
8	RxD/TxD-N
9	n.c.

**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 by the next PowerOn 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!

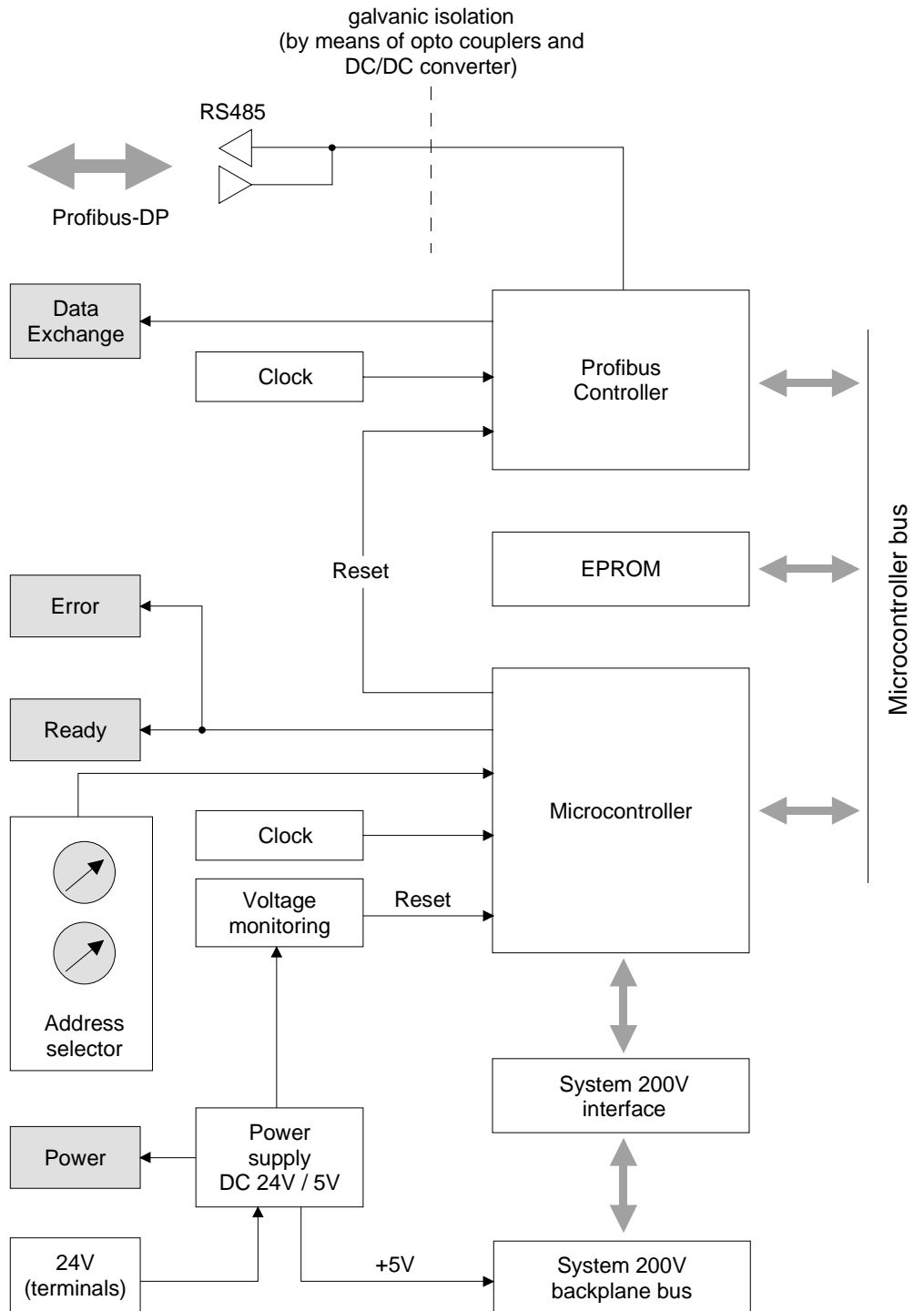
**Configuration of the outputs**

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 plug-in location. Seen from the hardware side, the module is directly beside the slave.

# IM 253DP - Slave - Block diagram

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



## IM 253DP - 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 here presented slaves are projected as "VIPA\_DP200V\_2" at the hardware configuration. After the installation of the GSD-file you will find this entry e.g. in the hardware catalog from Siemens under:

*Profibus-DP>Additional field devices>I/O>VIPA\_System\_200V*

### GSD-file

VIPA supplies a disc with every Profibus module. This disc contains all the GSD and type files of the VIPA Profibus modules.

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

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

### Deployment IM 253DP, DO 24xDC 24V

In a configuration employing Profibus slave combination modules, e.g. the VIPA 253-2DP20, you have to define the same parameters as indicated in table 4 above. At the project engineering of your peripheral modules (5.) you select the module type "253-2DP20". The plug-in location 1 is mandatory, because the module is, seen from the hardware side, directly beside the slave.



### Note

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

Deployment at a IM 208DP master

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

### Applications with the S7-400 from Siemens

The system S7-400 from Siemens uses double-word addressing for the configuration, i.e. a double-word is assigned to every module during configuration. For digital modules the high bytes of the double-words are not used. You can avoid this problem by using the GSD-file for the S7-400 from Siemens. This GSD-file is located in the subdirectory `.\S7-400\` on the accompanying disc.

Using the this GSD-file, you first have to configure all digital inputs followed by all the digital outputs and specify the respective sum in bytes. If there are no input or output modules, you enter 0Byte.

When the digital modules have been configured, you continue with the configuration of the analog modules as before.



#### Note!

Please note that the S7-400 system from Siemens requires the plug-in location number as module parameters for the analog modules. In this case, the first peripheral module is located at plug-in location 0.

### Applications with the IM 308-B from Siemens

The disc contains additional configuration type files for the Siemens configuration tools COM ET200 (DOS version for IM 308-B) and COM Profibus from Siemens (Windows version).

Please refer to the `readme.txt` located on the disc.



#### Note!

It may be necessary to implement certain modifications to the type file to ensure reliable operation of applications including the IM308-B from Siemens. Please contact the VIPA hotline in this respect.

### 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 selftest. 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 253DP - 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:

### *Plug-in location number*

For reasons of compatibility to VIPA slaves with revision level 4 or lower, you may here select the start number of the plug-in location numeration. With DP slaves rev. level 5 and higher, this parameter is ignored.

The following values are possible:

- 0: plug-in location 0 (default)
- 1: plug-in location 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

### *Diagnosis*

Via this parameter you influence the diagnosis 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 diagnosis*

Via this parameter you may influence the redundant diagnosis 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 253DP - Slave - Diagnostic functions

<b>Overview</b>	<p>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.</p> <p>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.</p>
<b>Internal diagnostic system messages</b>	<p>The system also stores diagnostic messages like the status "Ready" or "DataExchange". These are not send to the master.</p> <p>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.</p>
<b>Saving diagnostic data manually</b>	<p>You can manually save the diagnostic data in Flash-ROM by changing the address switch to 00 during "DataExchange" for a short while.</p>
<b>Diagnostic message in case of a power failure</b>	<p>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.</p> <p>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.</p>
<b>Direct diagnostics at the Profibus slave module</b>	<p>If you are employing VIPA Profibus slaves you may transfer the latest diagnostic data directly from the module into your PC for analysis by means of the download cable and the concerning "Slave Info Tool" software that are available form VIPA.</p>
<b>Diagnostic addition at IM 253DPR</b>	<p>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 right plug-in location), you are able to include 2Byte of the redundant state into the peripheral area.</p> <p>This virtual state "module" is available from GSD version 1.30 on.</p>

**Structure of the Profibus diagnostic data**

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 only for Profibus transfers
Byte 6	Header device related diagnostics	
<b>Byte 7 ... 29</b>	<b>Device related diagnostic data</b>	<b>Diagnostic data that is saved internally</b>
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 diagnostics 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 diagnostics**

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 diagnostics**

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 plug-in location 1 ... 32: module no. or plug-in location 0: module no. or plug-in location 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 plug-in location 1 ... 32: module no. or plug-in location 0: module no. or plug-in location 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 plug-in location 1 ... 32: module no. or plug-in location 0: module no. or plug-in location 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 plug-in location 1 ... 32: module no. or plug-in location 0: module no. or plug-in location 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 plug-in location failed.

Byte	Bit 7 ... Bit 0
0	28h: V-bus configuration error
1	Module no. or plug-in location 1 ... 32: module no. or plug-in location 0: module no. or plug-in location 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 plug-in location 1 ... 32: module no. or plug-in location 0: module no. or plug-in location 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 plug-in location 1 ... 32: module no. or plug-in location 0: module no. or plug-in location 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 plug-in location, otherwise the slave will throw a parameterization error.

**(De)activate  
diagnosis**

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

## Installation guidelines

### Profibus in general

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

9.6 ... 187.5kBaud	→	1000m
500kBaud	→	400m
1.5MBaud	→	200m
3 ... 12MBaud	→	100m
- The network may have a maximum of 10 segments. Segments are connected by means of repeaters. Every repeater is also seen as participant on the network.
- All devices communicate at the same baudrate, slaves adapt automatically to the baudrate.

### 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 **Optical Link Plug**, 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.

**Note!**

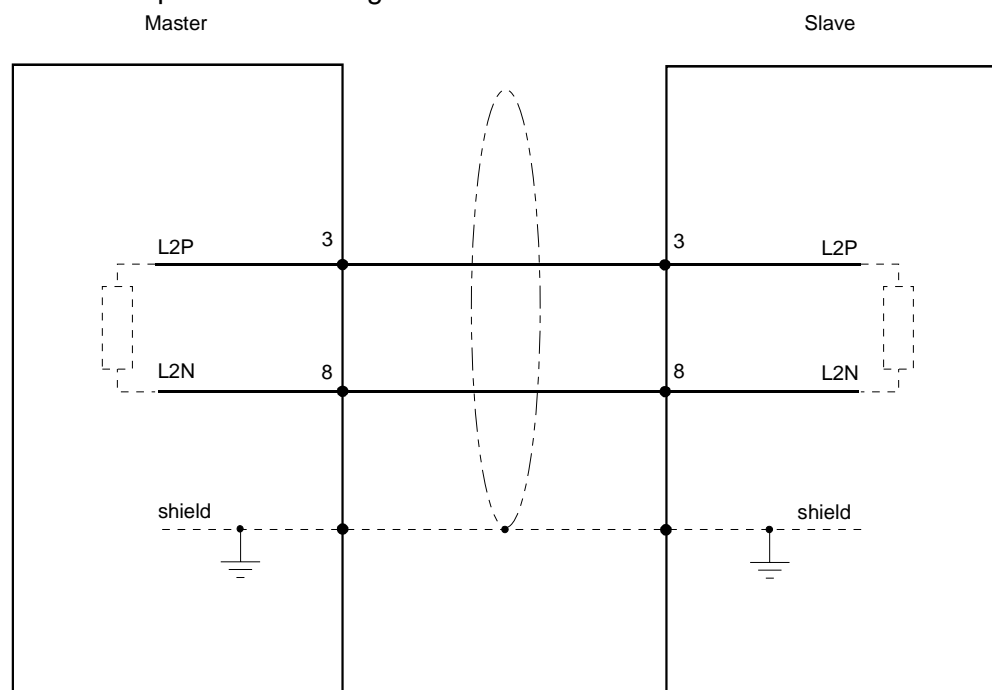
The Profibus line must be terminated with ripple resistor. Please ensure that the last participant the line is terminated by means of a terminating resistor.

The FO Profibus system does not require termination!

**Profibus using RS485**

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

The following figure shows a Profibus connection using RS485 together with the required terminating resistors:

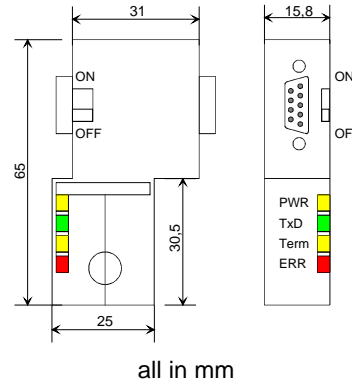


**Bus connector**



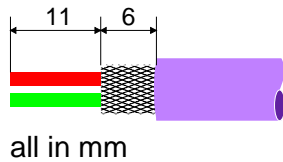
In systems with more than two stations all partners are wired in parallel. For that purpose, the bus cable must be feed-through uninterrupted.

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 diagnosis.



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.

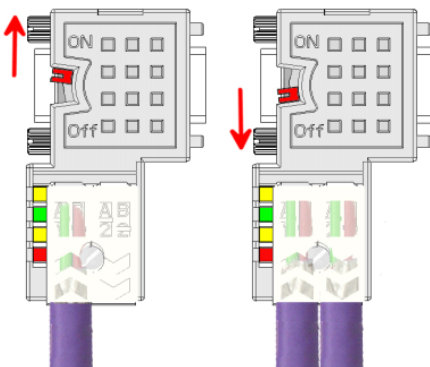


**Attention!**

The bus cable has always to be terminated at the bus ends with the ripple resistor to avoid reflections and therefore communication problems!

**Termination**

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!**

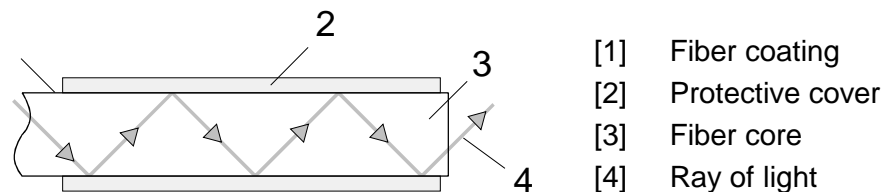
A complete description of installation and deployment of the terminating resistors is delivered with the connector.

### 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 construction of a fiber optic cable:



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 crosstalk 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

**Fiber optic cabling under Profibus**

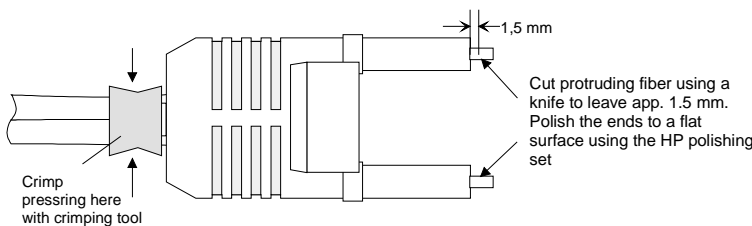
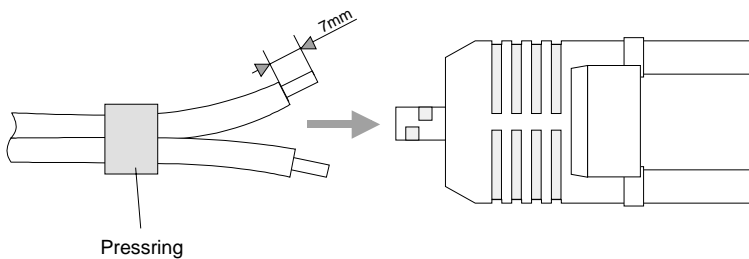
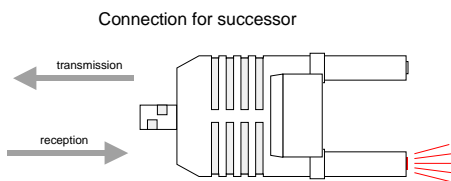
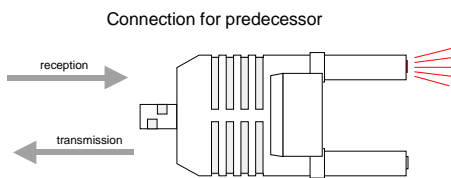
The VIPA fiber optic Profibus coupler employs dual core plastic fiber optic cable as the communication medium. Please keep the following points in mind when you connect your Profibus FO coupler: predecessor and successor must always be connected by means of a dual core FO cable.

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. VIPA recommends to use the FO connector supplied by Hewlett Packard (HP). These connectors are available in two different versions:

- FO connector with crimp-type assembly
- FO connector without crimp-type assembly

**FO connector with crimp-type assembly**



**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.

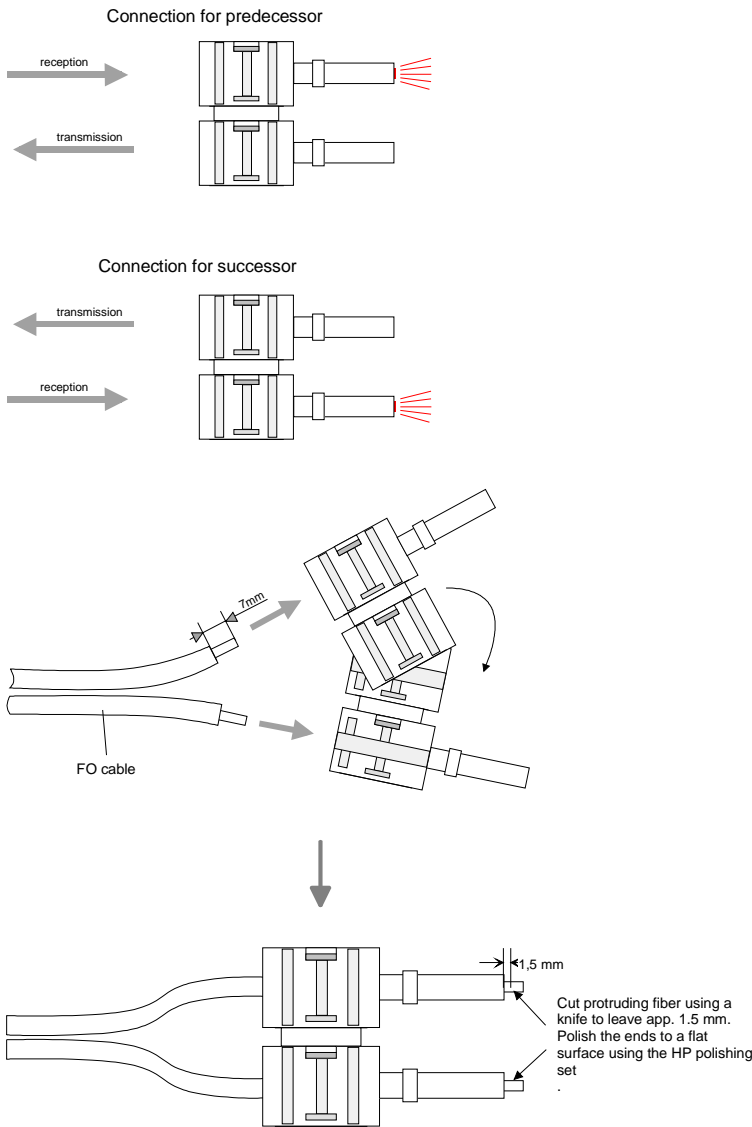
**Connector installation**

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.

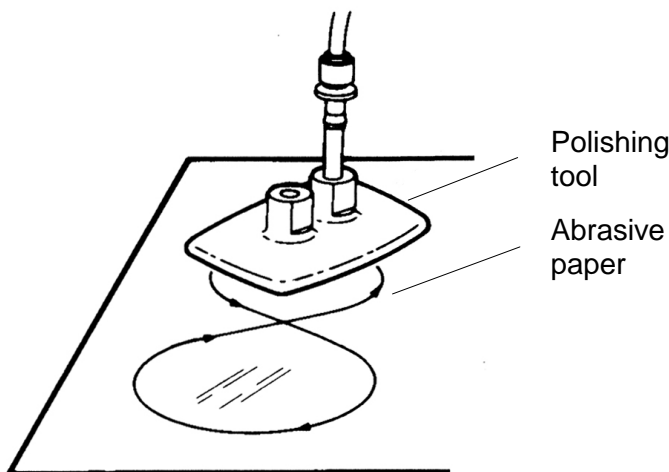
Insert the two cores into the plug so that the ends of the fiber optic cable protrude at the front. Keep an eye on the polarity of the cores (s.a.).

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 2<sup>nd</sup> connector type shown below.

**FO connector without crimp-type assembly**



**Cutting and polishing the ends of the FO cable**



**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.).

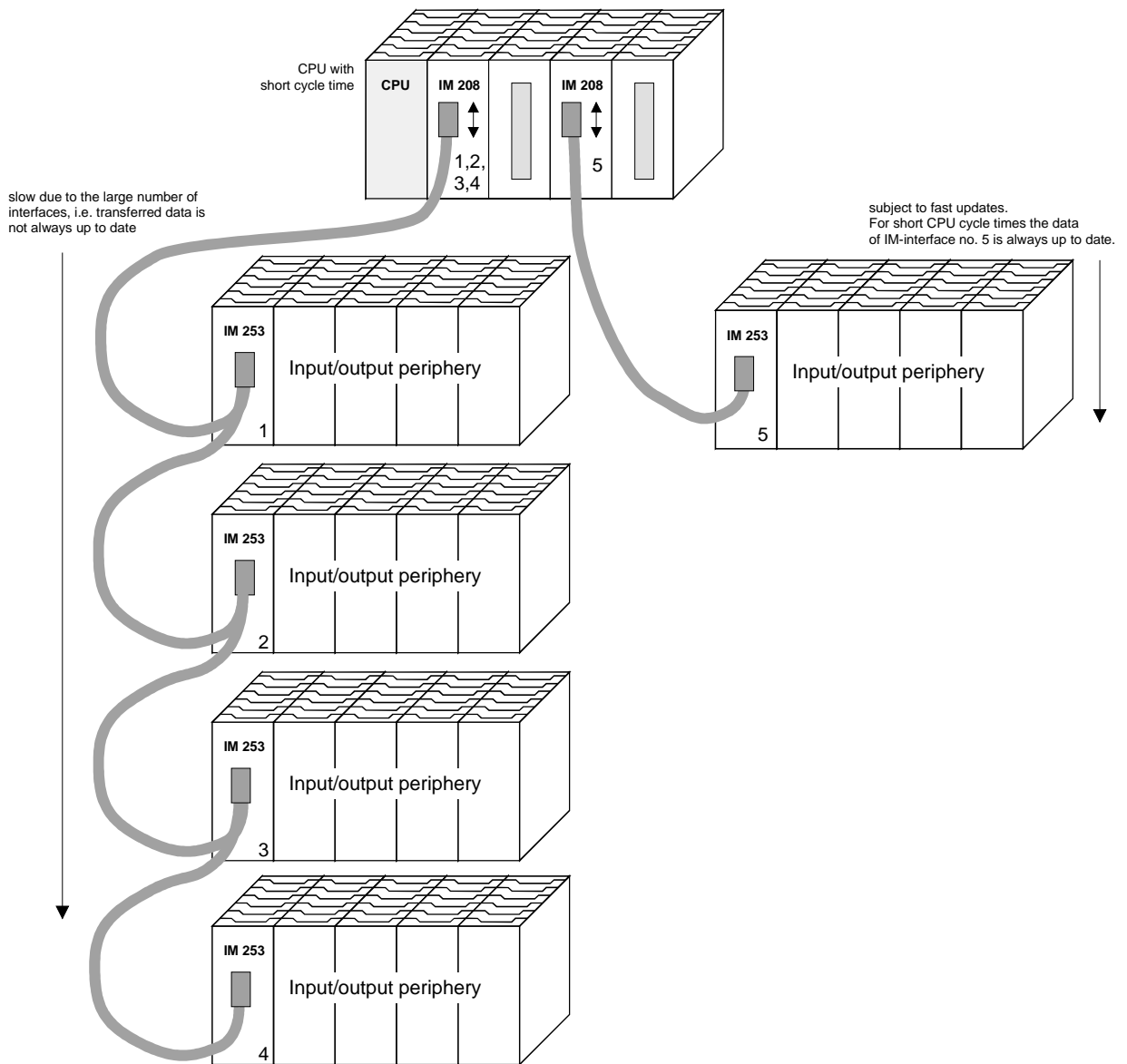
Cut protruding fiber using a knife so that app. 1.5mm are still visible. Polish the ends to a flat surface using the HP polishing set (HP order no.:HFBR-4593).

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

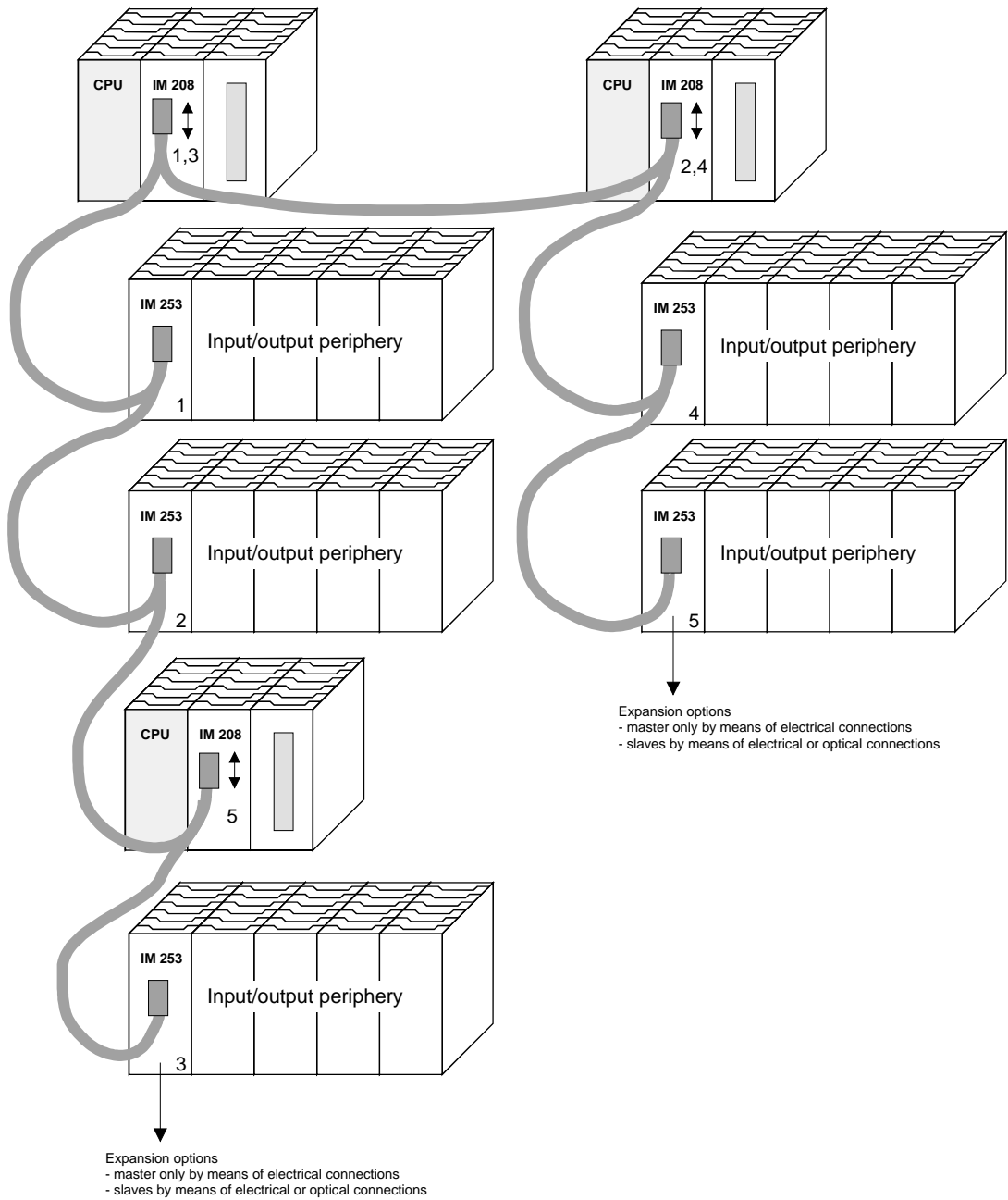
#### One CPU and multiple master connections

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.

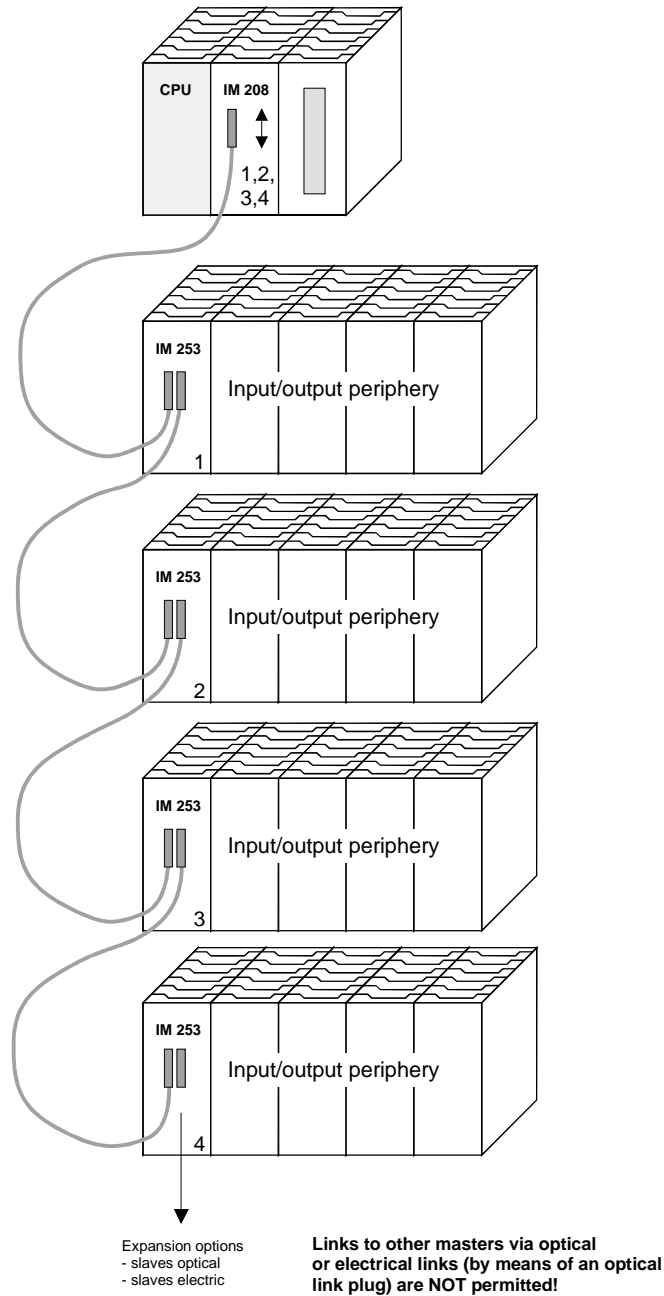


**Multi master system**

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



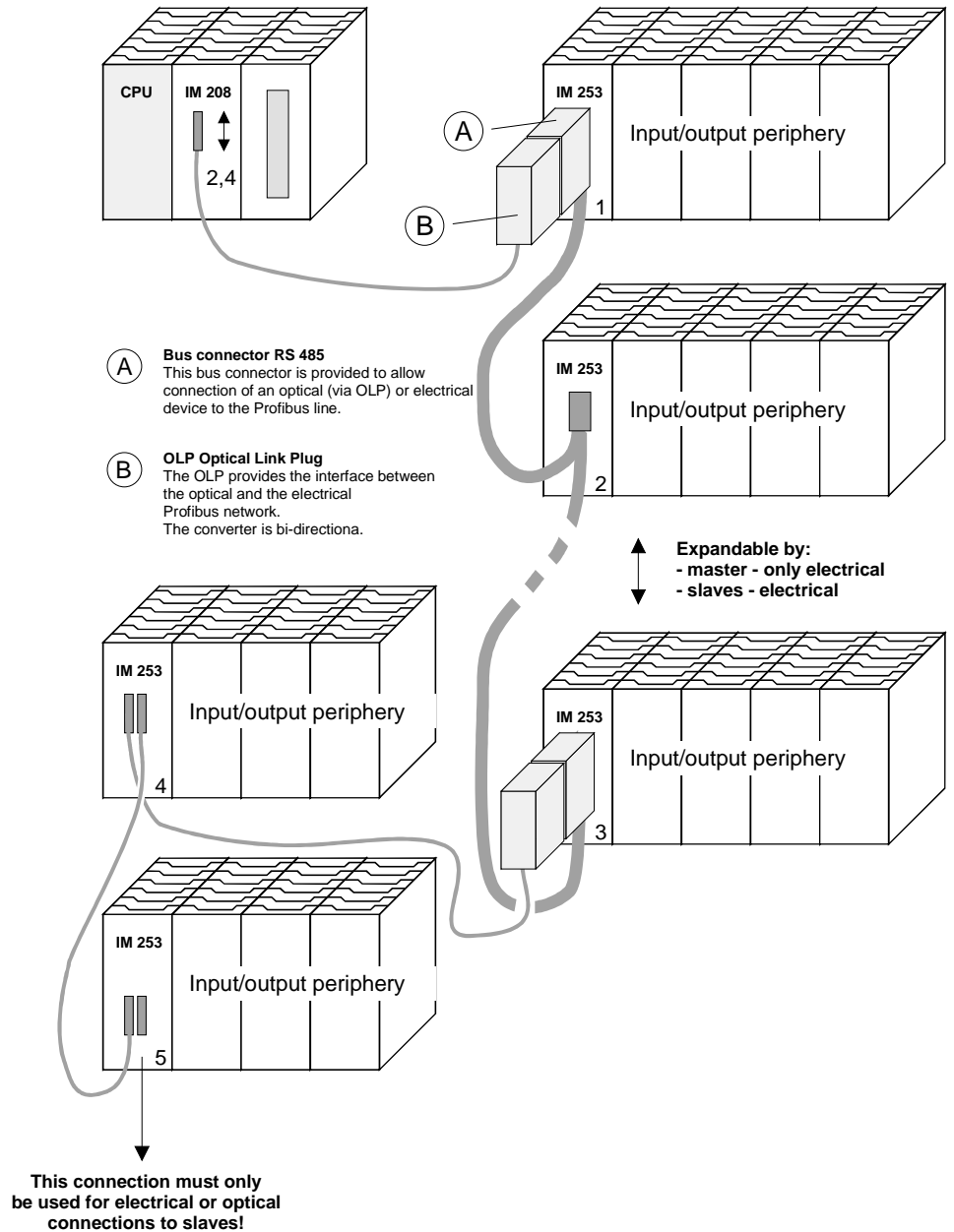
Optical Profibus





**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!**

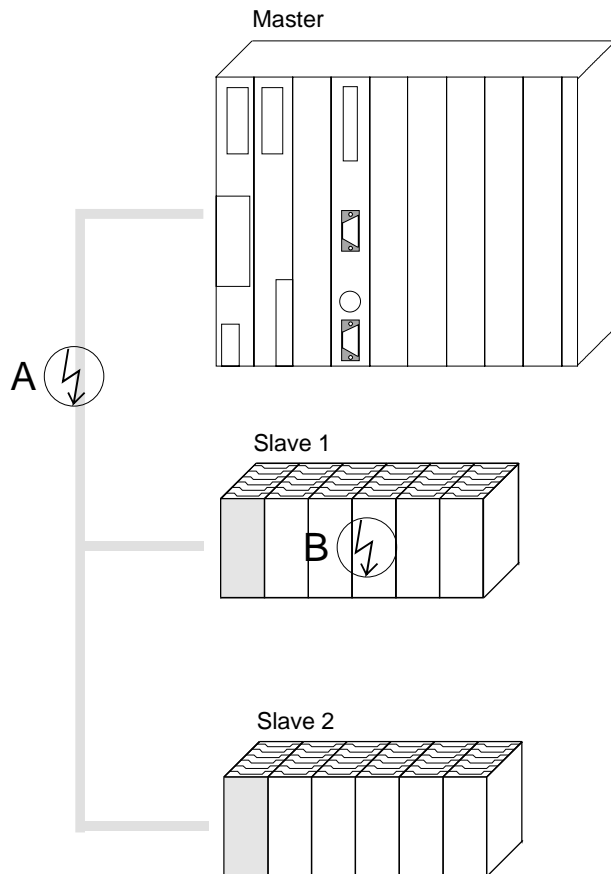


#### Note!

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	
	A	B
RD	blinks	off
ER	off	on
DE	off	off

LED slave 2	Position of interruption	
	A	B
RD	blinks	on
ER	off	off
DE	off	on

## Sample projects for Profibus communication

### Example 1

#### Problem

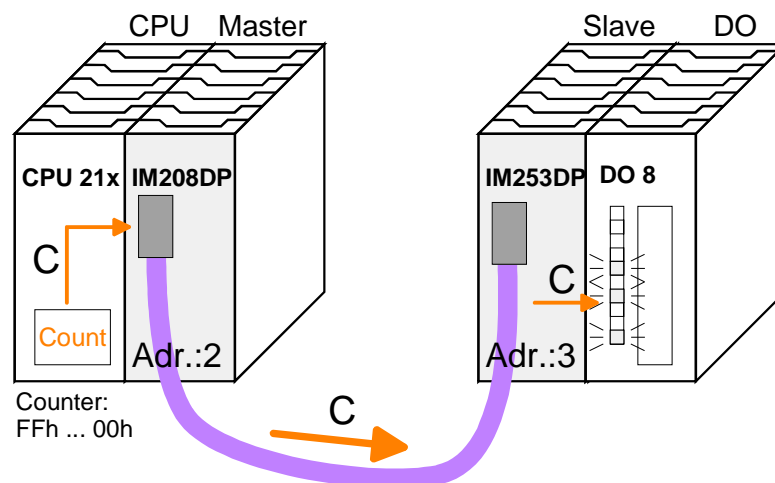
The following example describes a communication between a master and a slave system.

The master system consists of a CPU 21x 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 and IM 208DP (Master)

Counter value: MB 0 (FFh ... 00h)  
 Profibus address: 2

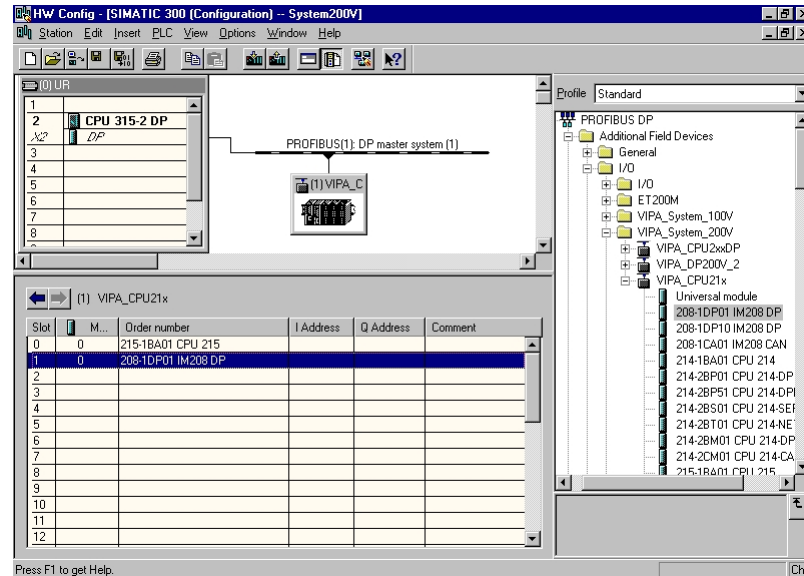
##### IM 253DP and DO (Slave)

Profibus address: 3  
 Output area: Address 0, length: 1 Byte

## Engineering IM 208DP

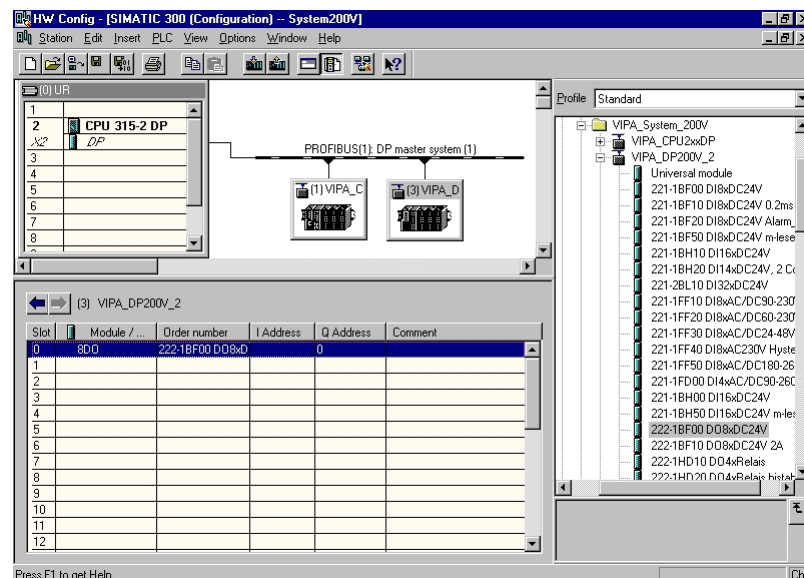
To be compatible with the STEP<sup>®</sup>7 projecting tool from Siemens, you have to execute the following steps for the System 200V:

- Start the Hardware configurator from Siemens
- Project a CPU 315-2DP with DP master (master address 2)
- Add a "virtual" Profibus slave "VIPA\_CPU21x" with address 1.
- Include the CPU 21x at plug-in location 0.
- Include the DP master 208-1DP01 at plug-in location 1.



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

- Add the Profibus slave "VIPA\_DP200V\_2" with address 3.
- Include the digital output module 222-1BF00 at plug-in location 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   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 under *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.

## Example 2

### Problem

This example shows a communication between a CPU 21x with IM 208 DP master and a CPU 21xDP (DP slave).

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

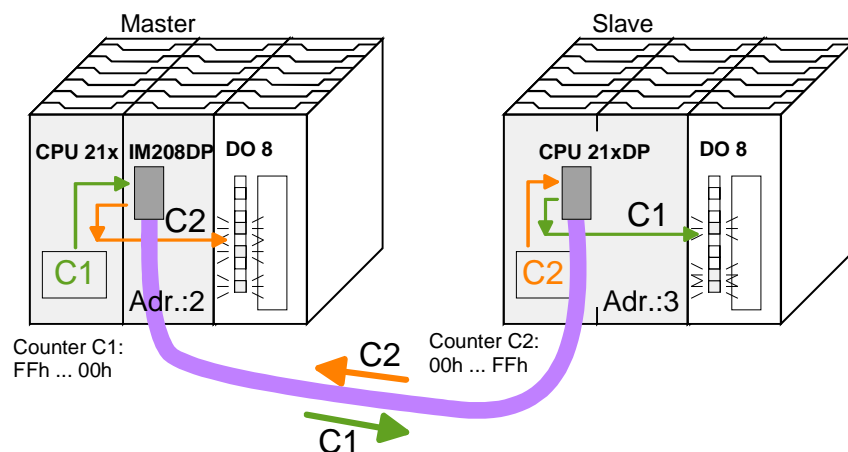
### Problem in detail

The CPU 21x 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 21xDP.

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 21xDP 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 21x (address 0).



### Project data

#### CPU 21x 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

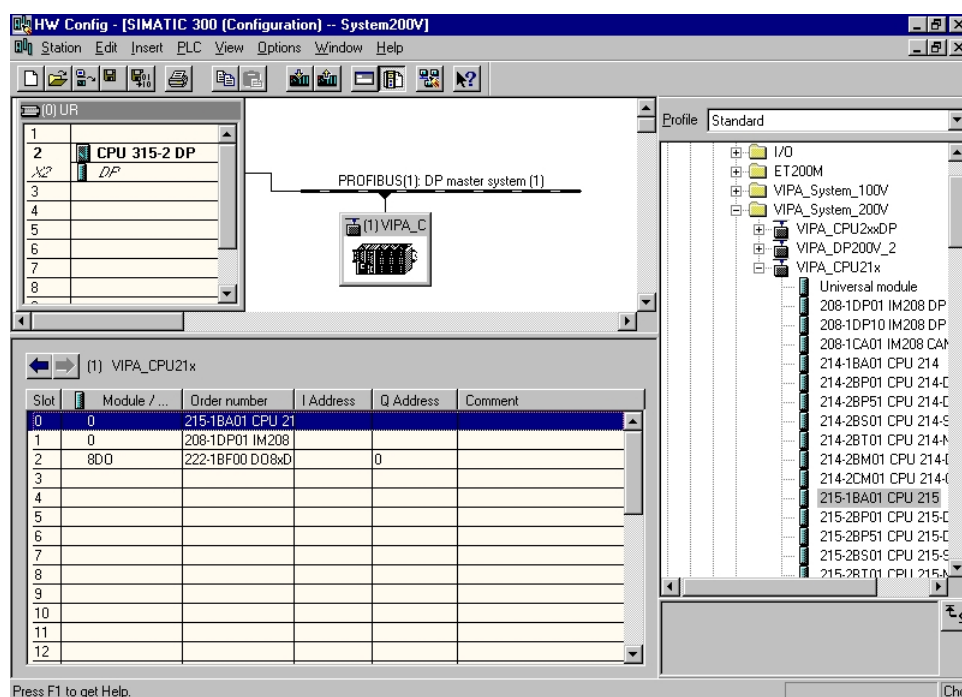
#### CPU 21xDP

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 21x of the DP master

To be compatible with the STEP<sup>®</sup>7 projecting tool from Siemens, you have to execute the following steps for CPU 21x and DP master:

- Start the Hardware configurator from Siemens
- Project a CPU 315-2DP with DP master (master address 2)
- Add a "virtual" Profibus slave **"VIPA\_CPU21x"** with address 1.
- Include a CPU 21x-1BA01 at plug-in location 0 of the slave system (like e.g. the CPU 215-1BA01).
- Include the DP master 208-1DP01 at plug-in location 1.
- Include the output module 222-1BF00 at plug-in location 2.

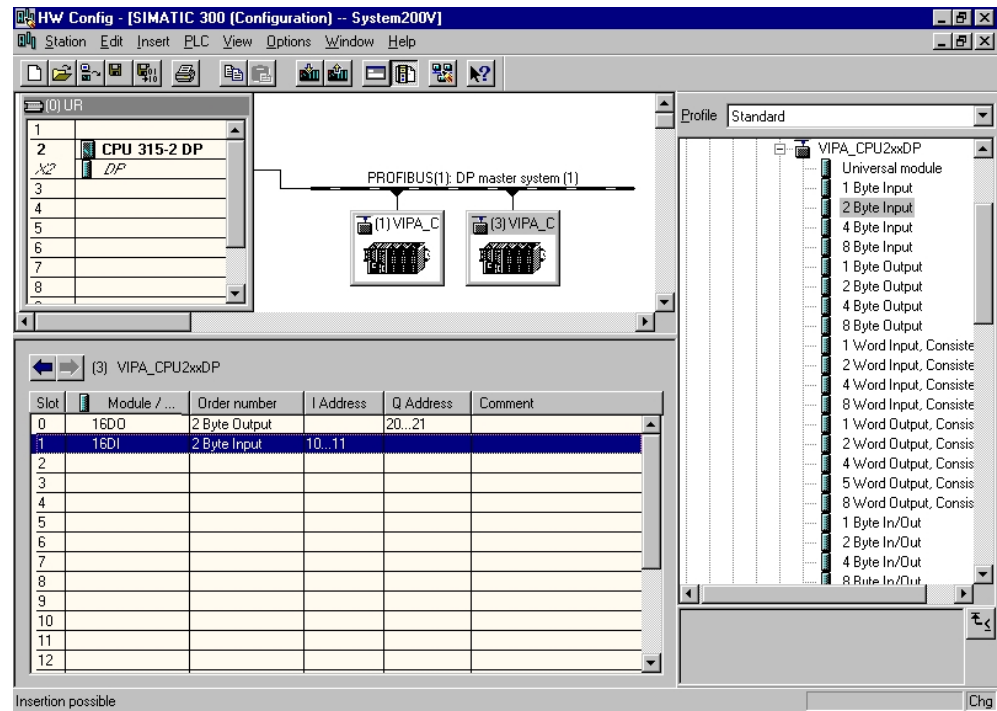


To connect your real CPU 21xDP, you have to execute the following steps after including the GSD-file:

- Add the Profibus slave **"VIPA\_CPU2xxDP"** (address 3)
- Include the "2 Byte Output" element at plug-in location 0 and choose the output address 20.
- Include the "2 Byte Input" element at plug-in location 1 and choose the input address 10.
- Save your project.

The hardwareconfiguration of CPU 21x and DP master ist now finished. The project is shown at the following screenshot.





### User application in the CPU 21x of the DP master

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   AB  20           control byte for slave CPU
L   B#16#FE
L   EB  10           load control value 0xFE
<>I control byte from slave
BEB CPU correct?
no -> End

-----
L   EB  11           Data transfer via Profibus
T   AB  0            load input byte 11 (output data
BE                               of the CPU214DP) and
                               transfer to output byte 0

```

- 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)

```

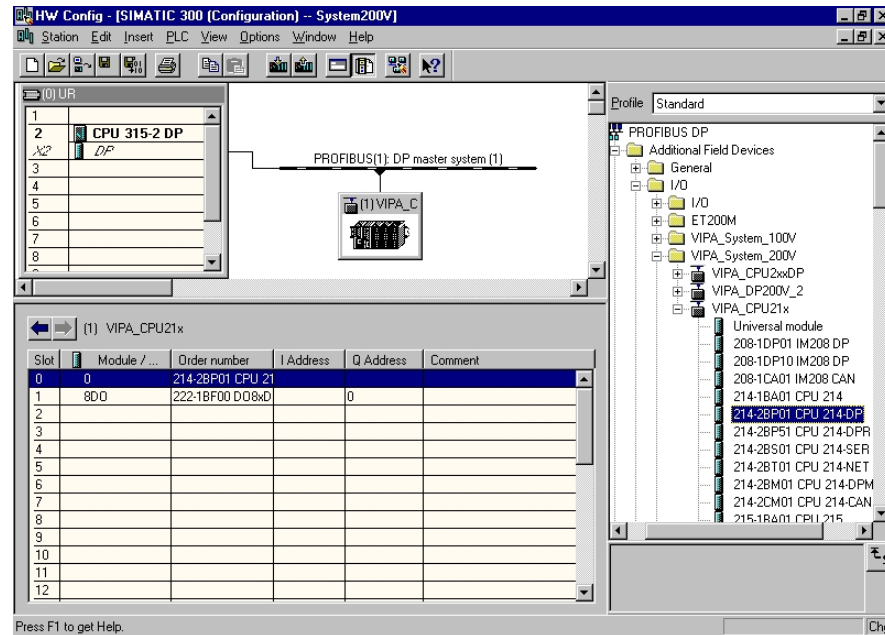
L   MB  0           counter from 0xFF to 0x00
L   1
-I
T   MB  0
T   AB  21           Transfer to output byte 21
BE                               (input data of the CPU214DP)

```

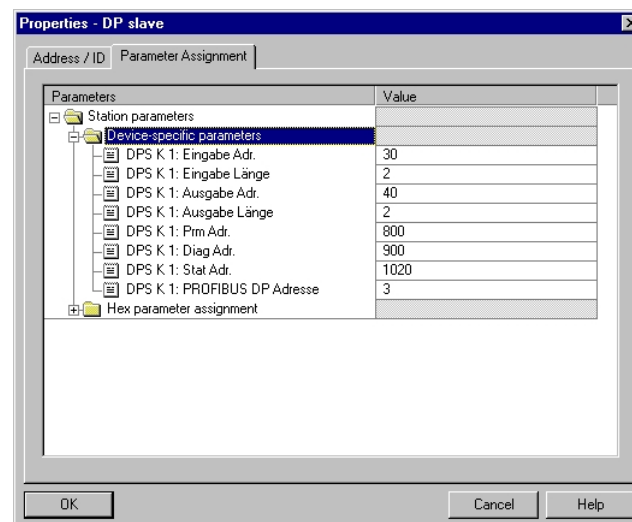
## Engineering CPU 21xDP

To be compatible with the STEP<sup>®</sup>7 projecting tool from Siemens, you have to execute the following steps for the CPU 21xDP:

- Start the Hardware configurator from Siemens
- Project a CPU 315-2DP with DP master (master address 2)
- Add a virtual Profibus slave "VIPA\_CPU21x" with address 1.
- Include the CPU 21x-2BP01 at plug-in location 0 (like e.g. the CPU 214-2BP01).
- Include the output module 222-1BF00 at plug-in location 1.



- In the parameter window of the CPU 214-2BP01, you select the following parameters:



- Save your project

**User application  
in the CPU 21xDP**

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   PEW 100           load status data and store it
T   MW 100           in the bit memory word

UN  M 100.5          commissioning by DP-Master
BEB                               successful? no -> End

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

L   B#16#FE          control byte for Master-CPU
T   PAB 40

-----
Data transfer via Profibus

L   PEB 31           load periphery byte 31 (input
T   AB 0             data from Profibus slave) and
                               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 21x.

**OB 35 (Time-OB)**

```

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

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

BE

```

## Technical data

### Profibus-DP master

#### IM 208DP

Electrical data	VIPA 208-1DP01
Power supply	via backplane bus
Current consumption	max. 380mA
Isolation	≥ AC 500V
Status indicators	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 repeaters for 12MBaud, 1000m with repeaters
Max. no. of stations	32 stations in any segment without repeaters. Extendible to 126 stations when using repeaters.
Combination with peripheral modules	
max. no of slaves	125
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	25.4x76x76
Weight	110g

**IM 208DPO**

Electrical data	VIPA 208-2DP10
Power supply	via backplane bus
Current consumption	max. 380mA
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 (see below)
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	50.8x76x76
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 (standard)

### IM 253DP

Electrical data	VIPA 253-1DP00
Power supply	DC 24V (20.4 ... 28.8V), ext. power supply at front
Current consumption	max. 1A
Output current backplane bus	max. 3.5A
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 (automatic adjustment)
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 diagnostics	The last 100 results are stored in Flash-ROM together with a time stamp. This data is accessible by means of a special tool and a cable.
Extended diagnostics	-
Combination with peripheral modules	
max. no. of modules	32 (depending on current consumption)
max. digital I/Os	32
max. analog I/Os	16
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	80g

**IM 253DPO**

Electrical data	VIPA 253-1DP10
Power supply	DC 24V (20.4 ... 28.8V), ext. power supply at front
Current consumption	1A max.
Output current backplane bus	max. 3.5A
Isolation	≥ AC 500V
Status indicator	via LEDs on the front
Connections/interfaces	4pole FO jack                                  Profibus connector
Profibus interface	
Connection	4pole socket for fiber optic cable
Network topology	Linear structure with dual FO cable, no bus termination 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 (see below)
Diagnostic functions	
Standard diagnostics	The last 100 results are stored in Flash-ROM together with a time stamp. This data is accessible by means of a special tool and a cable.
Extended diagnostics	-
Combination with peripheral modules	
max. no of modules	32 (depending on current consumption)
max. digital I/Os	32
max. analog I/Os	16
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	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**

Electrical data	VIPA 253-2DP50
Power supply	DC 24V (20.4 ... 28.8V), ext. power supply at front
Current consumption	max. 1A
Output current backplane bus	max. 3.5A
Isolation	≥ AC 500V
Status indicator	via LEDs on the front
Connections/interfaces	9pin D-type socket (2x)      Profibus connector
2 channels	DP1 / DP2
Profibus interface	
Connection	9pin D-type socket (2x)
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 (automatic adjustment)
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 diagnostics	The last 100 results are stored in Flash-ROM together with a time stamp. This data is accessible by means of a special tool and a cable.
Extended diagnostics	-
Combination with peripheral modules	
max. no of modules	32 (depending on current consumption)
max. digital I/Os	32
max. analog I/Os	16
Dimensions and weight	
Dimensions (WxHxD) in mm	50.8x76x76
Weight	120g



**Profibus-DP  
slave  
combination  
module**

**IM 253DP  
DO 24xDC 24V**

Electrical data	VIPA 253-2DP20
Power supply	DC 24V (20.4 ... 28.8V), ext. power supply at front
Current consumption	max. 5A
Output current backplane bus	max. 3.5A
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.
Medium	Screened twisted pair cable, under certain conditions unscreened lines are permitted.
Data transfer rate	9.6kBaud to 12MBaud (automatic adjustment)
Total length	100m without repeaters for 12MBaud; 1000m with repeaters
Max. no of stations	32 stations in any segment without repeaters. Extendible to 126 stations when using repeaters.
Status indicator	via LEDs on the front
Combination with peripheral modules	
max. no of modules	31 (depending on current consumption)
max. digital I/Os	31
max. analog I/Os	16
Output unit	
Number of outputs	24
Nominal load voltage	DC 24V (18...35V) supplied internally via Profibus coupler
Output current per channel	1A (sum current max. 4A)
Status indicator	Power (PW) fuse OK, Error (ER) short circuit, overload
Programming data	
Output data	4Byte (3Byte are used)
Dimensions and weight	
Dimensions (WxHxD) in mm	50.8x76x76
Weight	150g



## Chapter 3 Interbus

### Overview

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

A description of the Interbus principles 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 principles
- Hardware structure, deployment and commissioning of the Interbus coupler
- Technical data

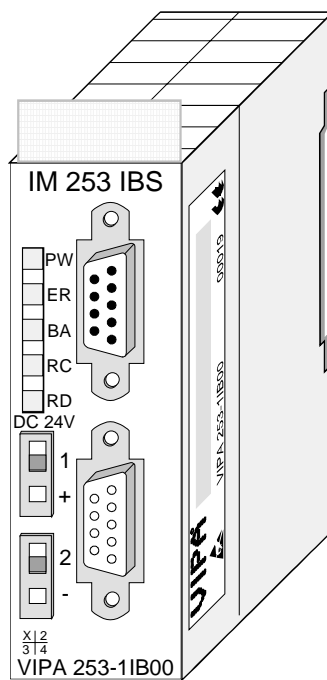
### Content

Topic	Page
<b>Chapter 3 Interbus</b> .....	<b>3-1</b>
System overview .....	3-2
Principles .....	3-3
IM 253IBS - Interbus coupler - Construction .....	3-7
Connection to Interbus.....	3-10
Deployment with Interbus .....	3-11
Commissioning .....	3-15
Technical data .....	3-18

## 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

## Principles

### 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 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.

**Modes of operation**

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.

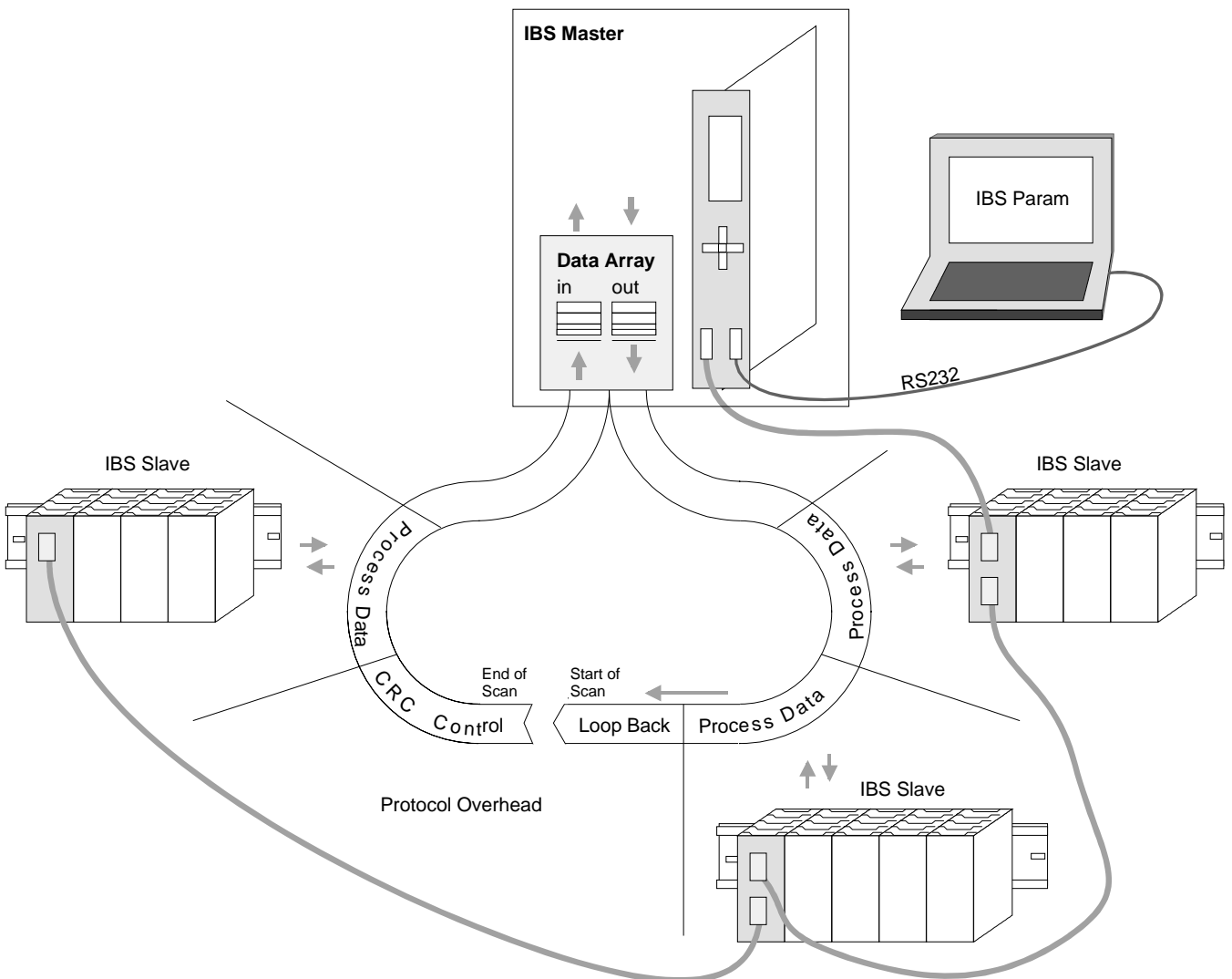
The physical level of Interbus is based upon the RS422 standard. The signals are connected by means of twisted pair lines. The outbound signal as well as the return signal of Interbus is re-routed via the same cable and every connected station. Communications between 2 devices require a 5core cable due 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.

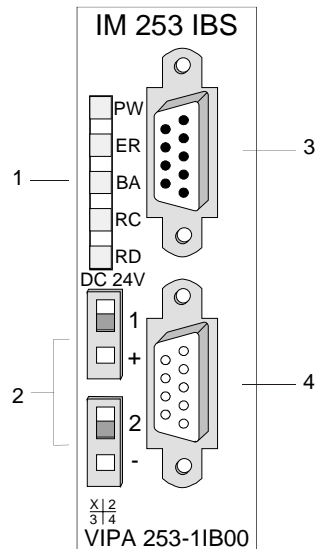
When the data check sequence has been processed, output data for the process is transferred from the shift registers. This is stored in the devices connected to the bus and transferred to the respective periphery. At the same time, new information is read from the periphery into the shift registers of 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 - Construction

### Construction



- [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

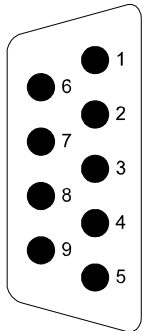
The module has a number of LEDs are available for diagnostic purposes on the bus. The following table explains the purpose and the color of the different LEDs.

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.

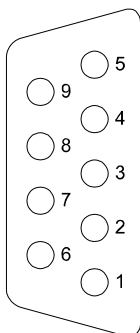
The following diagram shows the pin assignment for this interface:

**Inbound bus line (9pin D-type plug)**

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

<sup>\*)</sup> 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

**Voltage 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.

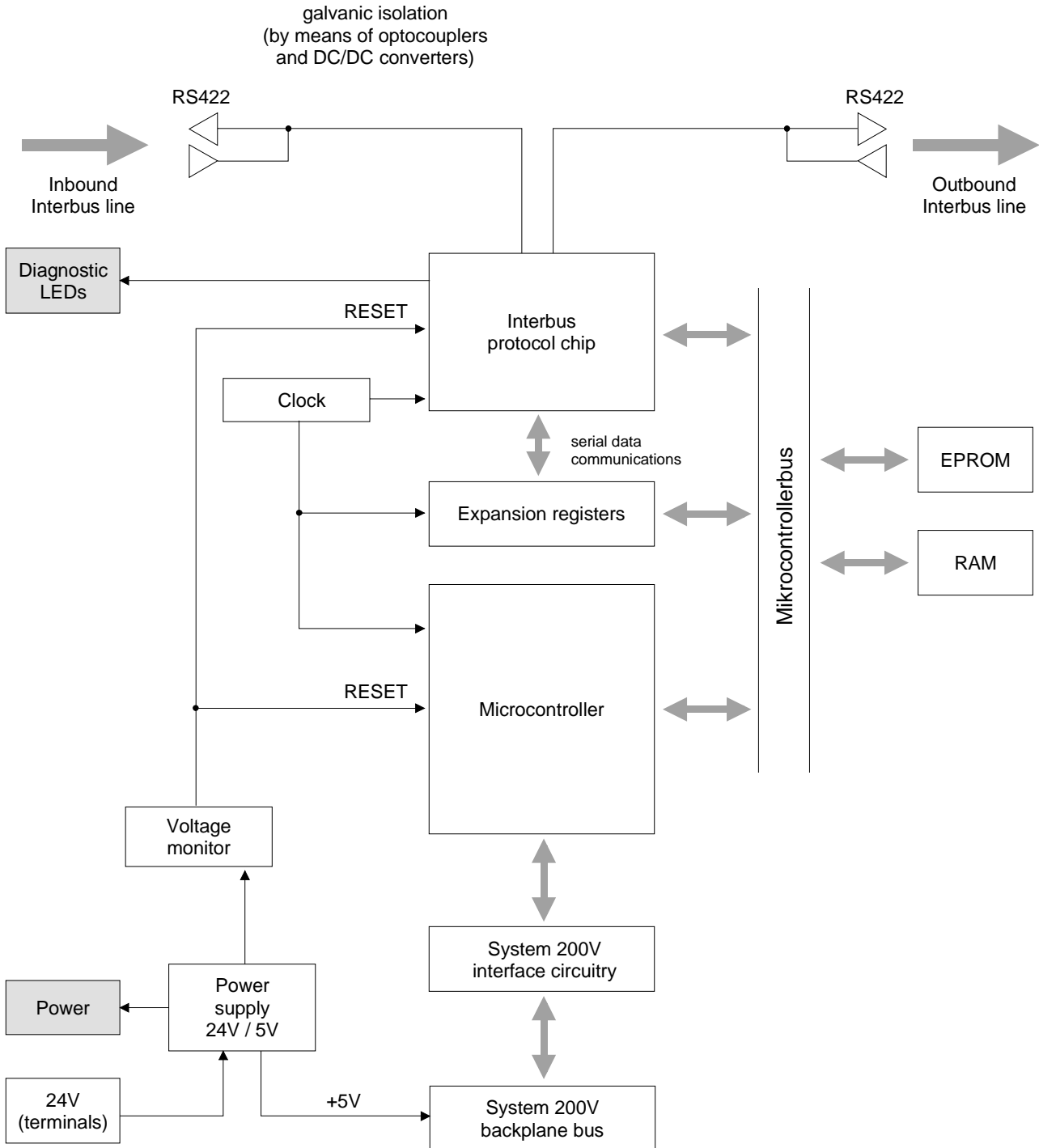
Interbus and the backplane bus are isolated from each other.

**Note!**

Please pay attention to the polarity of the power supply!

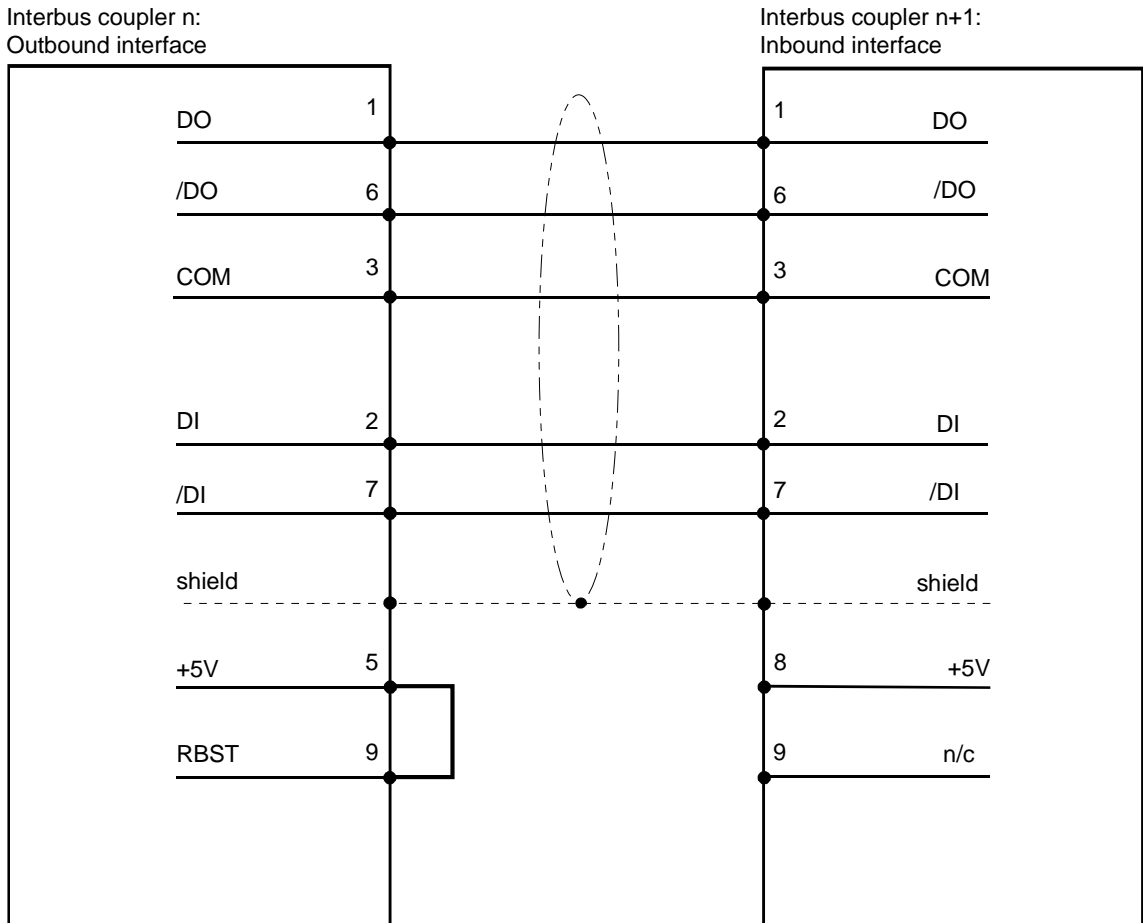
**Block diagram**

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 data allocation

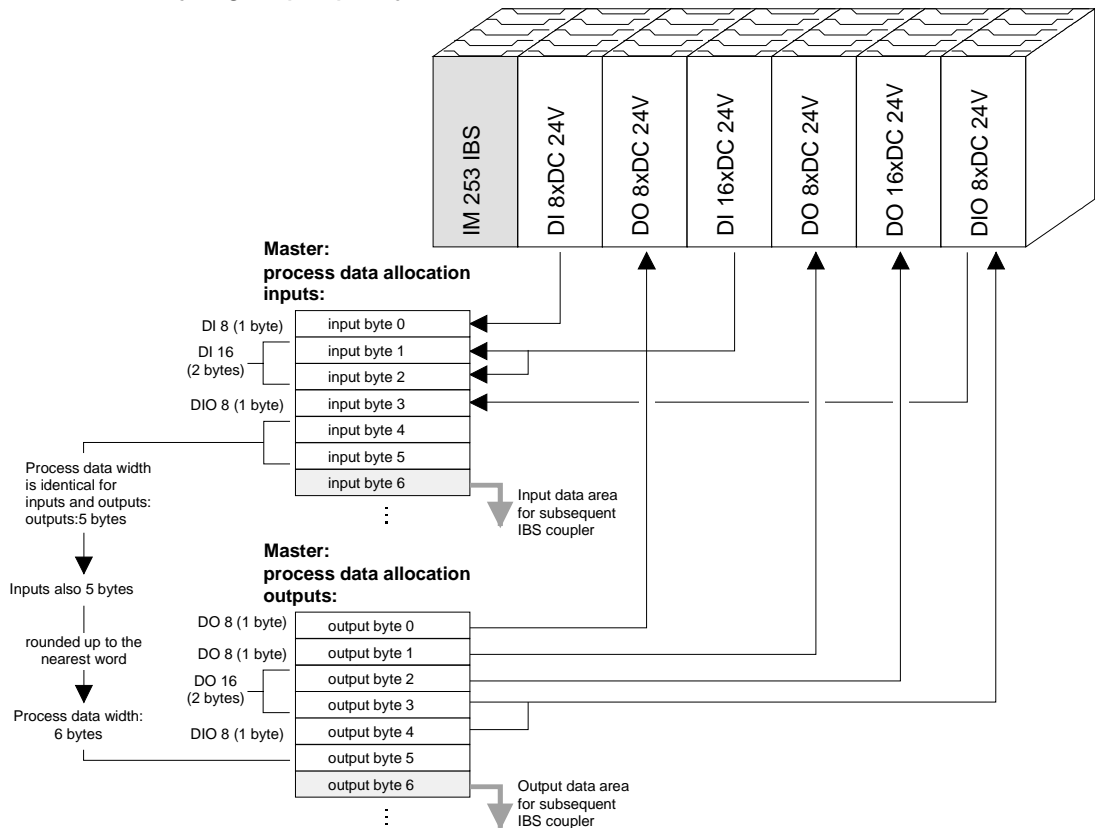
The bus coupler determines the configuration of the installed modules after power on and enters the respective data into the internal process image. This process image is sent to the master. From the process images the master generates a process data list for all couplers connected to the bus. The following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

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

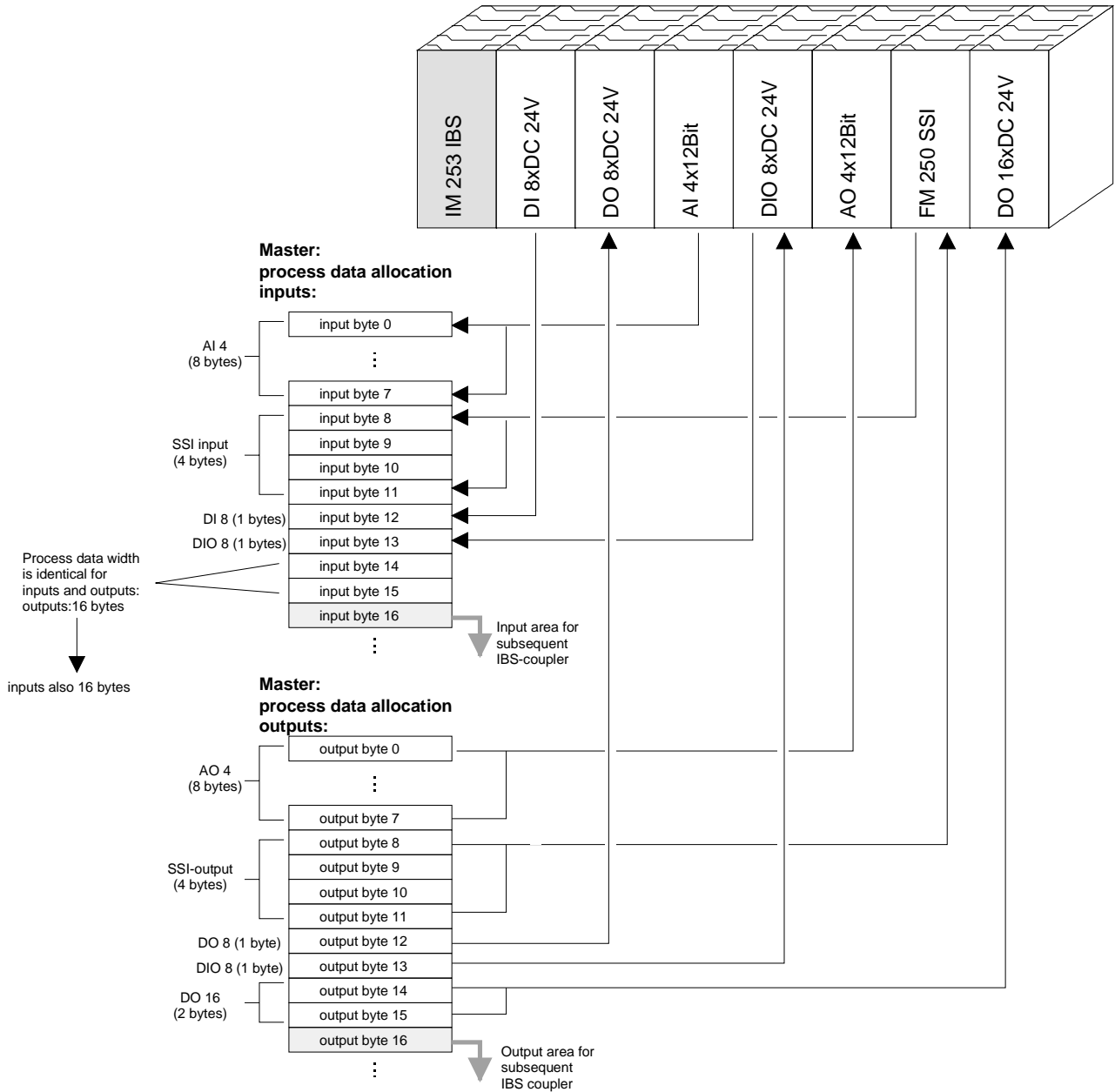
- Digital signals are bit orientated, i.e. each channel is associated with one bit in the process image.
- Separate areas exist for input and output data.
- In the input and output areas non-digital modules are always placed before digital modules.
- The sequence of these allocations depends on the plug-in 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 ET200V 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

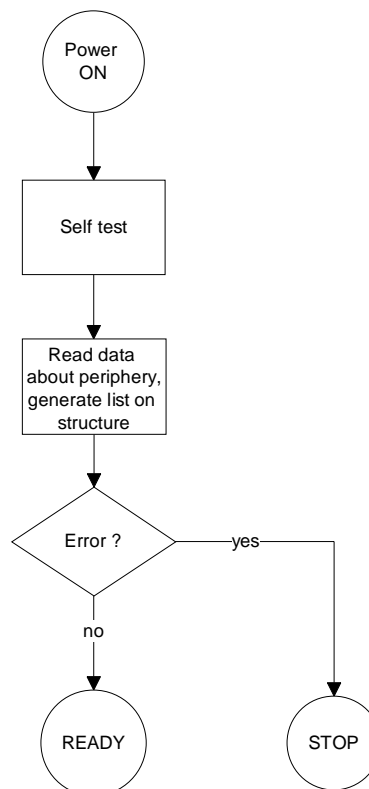
- Assemble your Interbus coupler using the required modules.
- Configure the Interbus coupler by means of the configuration tool that was supplied with the master.
- Connect the Interbus cable to the coupler and turn the power on.

### 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.

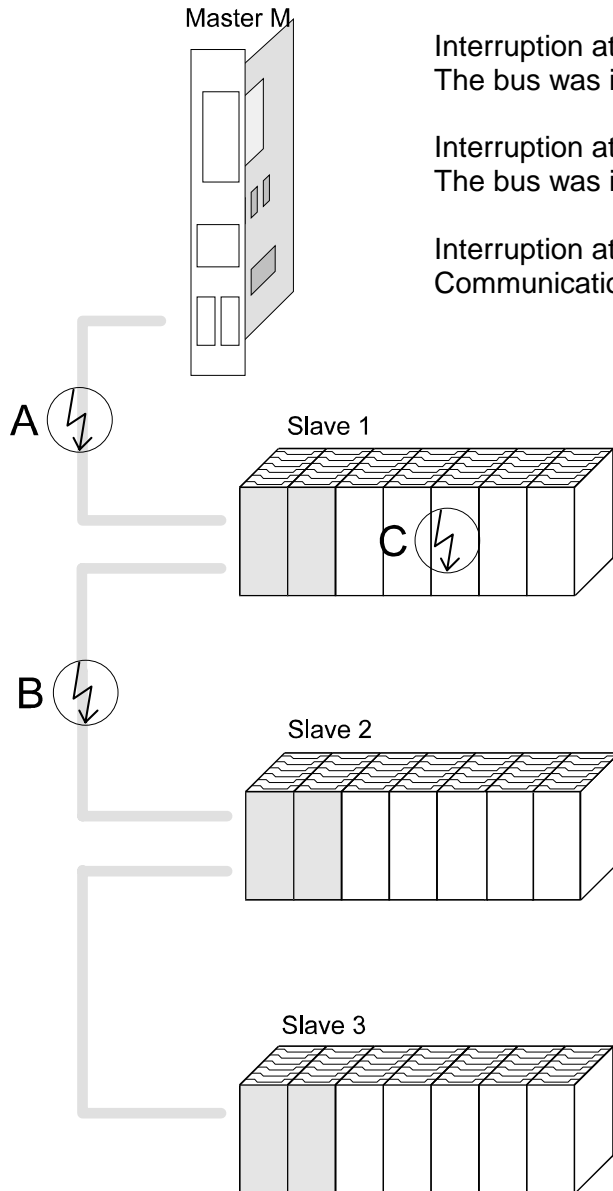
Now the peripheral structure is read in. First the number of modules connected to the bus is determined. Then the modules are identified by means of their type identifier. When the peripheral structure has been registered the location identifiers for the modules are generated. This is then transferred 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	max. 300mA
Output current backplane bus	max. 3.5A
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
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	25.4x76x76
Weight	80g

## Chapter 4 CANopen

### Overview

This chapter contains the description of the VIPA CANopen 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 principles
- The VIPA CANopen 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
<b>Chapter 4 CANopen .....</b>	<b>4-1</b>
System overview.....	4-2
Principles .....	4-3
IM 253CAN - CANopen Slave - Construction.....	4-5
IM 253CAN, DO 24xDC 24V - Construction.....	4-9
CANopen fast introduction .....	4-13
Baudrate and module-ID settings.....	4-17
Message structure .....	4-18
PDO - process data object.....	4-20
SDO - service data object.....	4-24
Object directory .....	4-26
Emergency Object .....	4-67
NMT - network management .....	4-68
Technical data .....	4-70

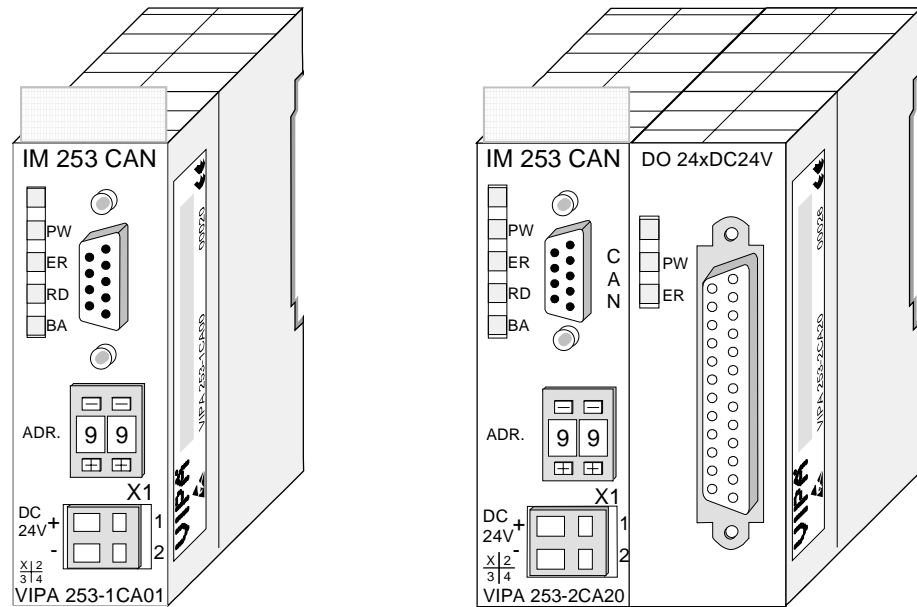
## System overview

Currently two CANopen bus couplers are available from VIPA:

- CANopen slave IM 253CAN
- CANopen slave IM 253CAN with DO 24xDC 24V

### CANopen slave IM 253CAN

- 10 receive and 10 transmit PDOs (PDO linking, PDO mapping)
- 253-2CA20: 1 receive PDO (PDO linking, PDO mapping: fix)
- 2 SDOs
- Support of all baudrates



### Order data

Order number	Description
VIPA 253-1CA01	CAN-Bus CANopen Slave
VIPA 253-2CA20	CAN-Bus CANopen Slave with DO 24xDC 24V

## Principles

### 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  $4.7 \times 10^{-11}$ . Bad messages are flagged and retransmitted automatically.

In contrast to Profibus and INTERBUS-S, CAN defines under the CAL-level-7-protocol (CAL=**CAN** application layer) defines various level-7 user profiles for the CAN-Bus. One standard user profile defined by the CiA (**CAN** in **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 120 $\Omega$  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.

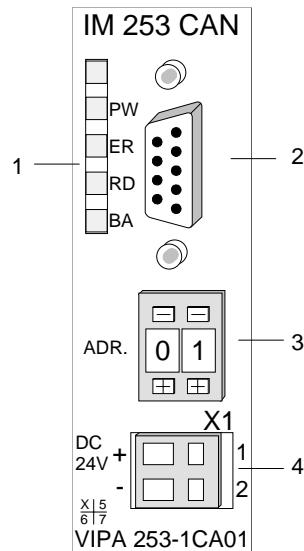
CAN-Bus access control is performed by means of a collision-free, bit-based arbitration algorithm. Collision-free means that the final winner of the arbitration process does not have to repeat his message. The station with the highest priority is selected automatically when more than one station accesses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



## IM 253CAN - CANopen Slave - Construction

- Properties**
- 10 Rx and 10 TxPDOs
  - 2 SDOs
  - Support of all baudrates
  - PDO linking
  - PDO mapping

**Construction**



- [1] LED status indicators
- [2] CAN-Bus socket
- [3] Address or baudrate selector
- [4] Connector for an external 24V supply

### Components

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

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

**Status indicator as a combination of LEDs**

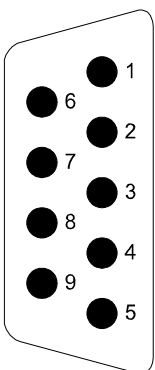
Various combinations of the LEDs indicate the different operating states:

- PW on
  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	optional ground
7	CAN high
8	n.c.
9	optional pos. supply

**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 "Baudrate and module-ID settings" in this chapter.

**Power supply**

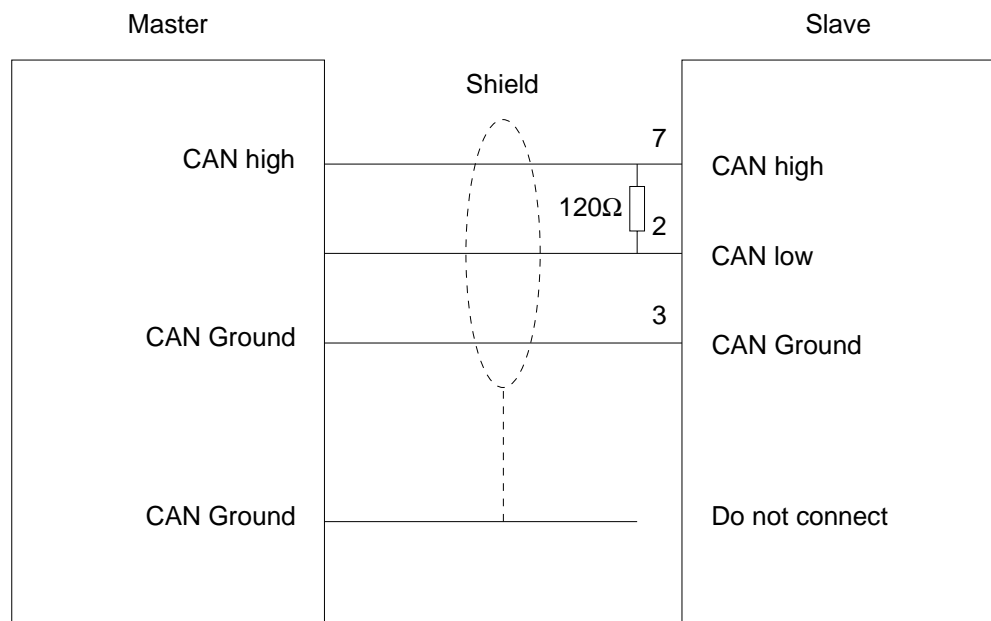
The CAN-bus coupler is equipped with an internal power supply. This power supply requires an external supply of DC 24V. In addition to the internal circuitry of the bus coupler the supply voltage is also used to power any devices connected to the backplane bus. Please note that the maximum current available 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.

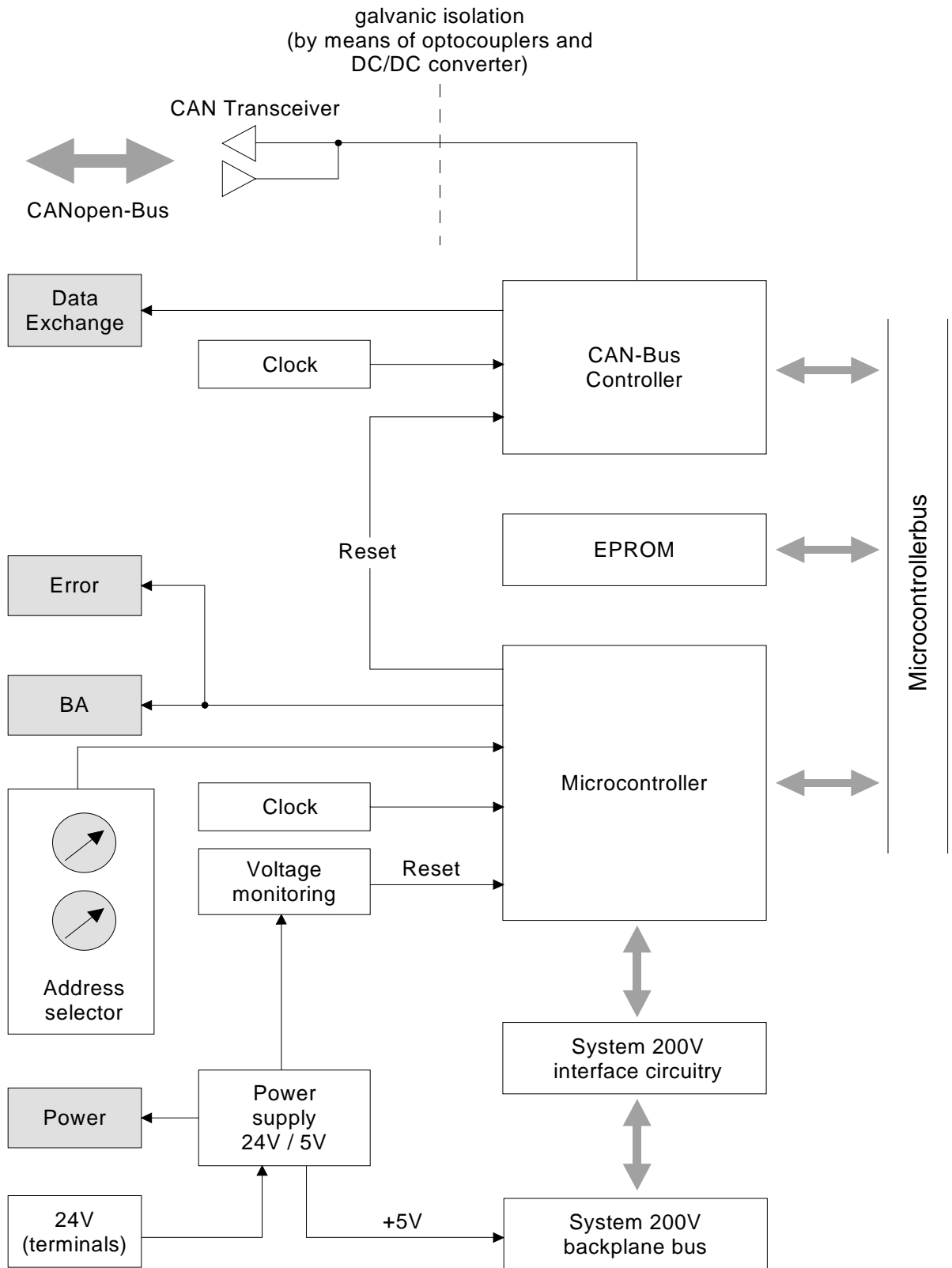


**Note!**

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

**Block diagram**

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

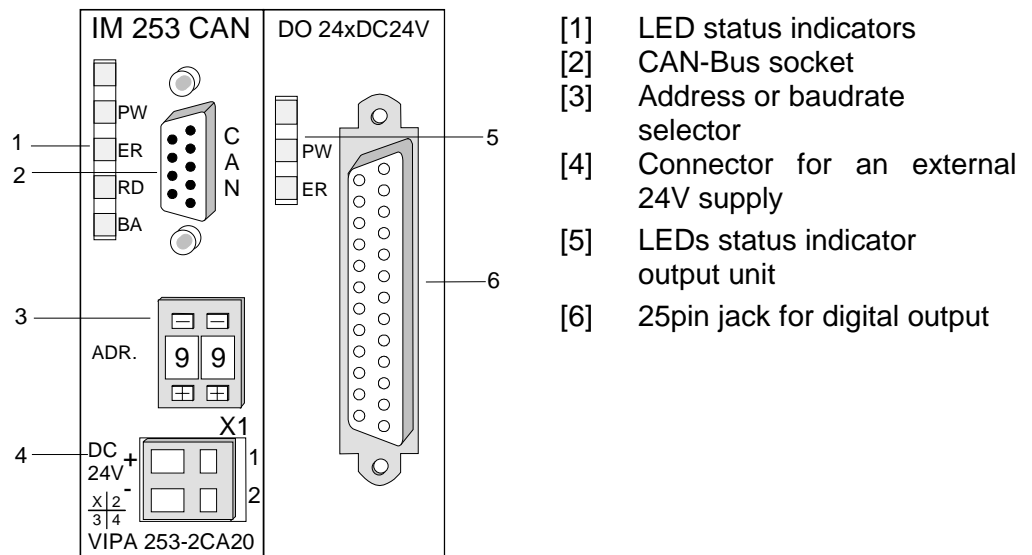


## IM 253CAN, DO 24xDC 24V - Construction

### 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

### Construction



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

**Status indicator as a combination of LEDs**

Various combinations of the LEDs indicate the different operating states:

-  PW on
  -  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

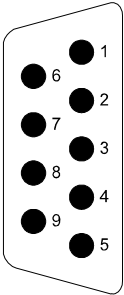
**LEDs digital output unit**

The digital output unit provides 2 LEDs with the following function:

Label	Color	Description
PW	yellow	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.

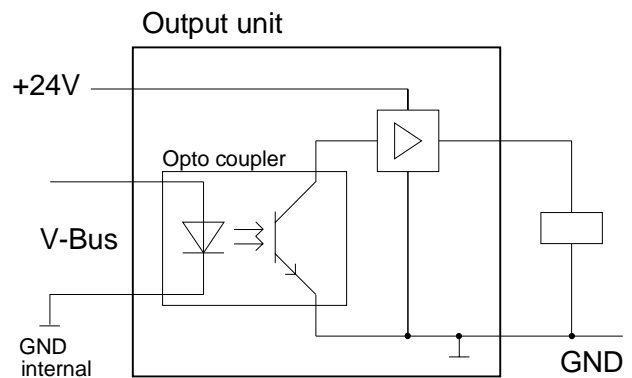
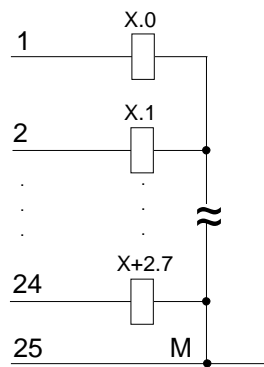
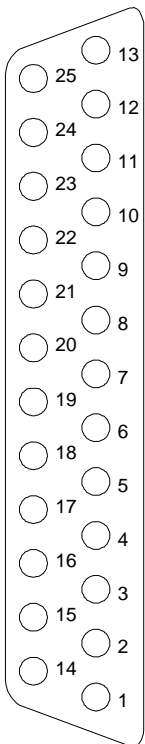
The following diagram shows the pin assignment for the interface.



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

**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**

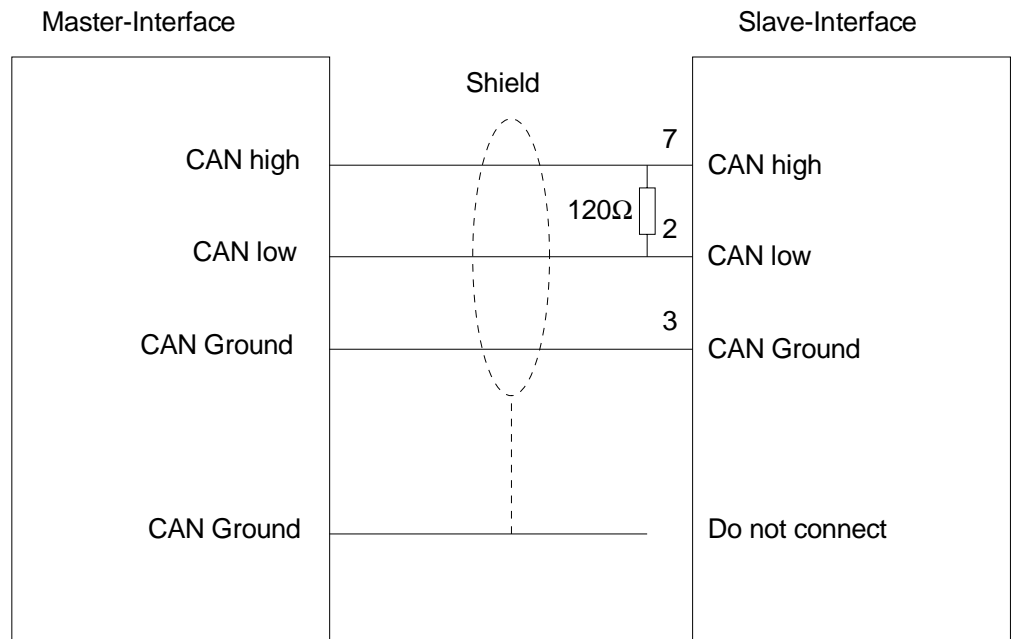
The address selector is used to specify the module-ID as well as the CAN baudrate.  
 For details please refer to the section under the heading "Baudrate and module-ID settings" in this chapter.

**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!



## CANopen 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.: **0x15AE = 15AEh**

### 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 "Baudrate and module-ID settings" 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*

<b>Identifier</b> 11Bit	<b>DI 0</b> 8Bit	<b>DI 1</b> 8Bit	<b>DI 2</b> 8Bit	...	<b>DI 7</b> 8Bit
-------------------------	------------------	------------------	------------------	-----	------------------

The CAN messages with digital output data are represented as follows:  
*Identifier 0x200+Node address + up to 8Byte user data*

<b>Identifier</b> 11Bit	<b>DO 0</b> 8Bit	<b>DO 1</b> 8Bit	<b>DO 3</b> 8Bit	...	<b>DO 7</b> Bit
-------------------------	------------------	------------------	------------------	-----	-----------------

**Analog in-/outputs** The CAN messages with analog input data are represented as follows::  
*Identifier 0x280+Node address + up to 4Words user data*

<b>Identifier</b> 11Bit	<b>AI 0</b> 1Word	<b>AI 1</b> 1Word	<b>AI 2</b> 1Word	<b>AI 3</b> 1Word
-------------------------	-------------------	-------------------	-------------------	-------------------

The CAN messages with analog output data are represented as follows:  
*Identifier 0x300+Node address + up to 4Words user data*

<b>Identifier</b> 11Bit	<b>AI 0</b> 1Word	<b>AI 1</b> 1Word	<b>AI 2</b> 1Word	<b>AI 3</b> 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*

<b>Identifier</b> 11Bit
-------------------------

The System 200V node answers with a telegram that contains one state byte:

*Identifier 0x700+Node address + State byte*

<b>Identifier</b> 11Bit	<b>Status</b> 8Bit
-------------------------	--------------------

Bit 0 ... 6: Node state  
 0x7F: Pre-Operational  
 0x05: Operational  
 0x04: Stopped resp. 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*

<b>Identifier</b> 11Bit	<b>Status</b> 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 0x1014 in hexadecimal view: **0x80 + Module-ID**.

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

*Identifier 0x80 + Node address + 8Byte user data*

<b>Identifier</b> 11Bit	<b>EC0</b>	<b>EC1</b>	<b>Ereg</b>	<b>Inf0</b>	<b>Inf1</b>	<b>Inf2</b>	<b>Inf3</b>	<b>Inf4</b>
-------------------------	------------	------------	-------------	-------------	-------------	-------------	-------------	-------------

Error Code	Meaning	Info 0	Info 1	Info 2	Info 3	Info 4
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
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.

## Baudrate and module-ID settings

### Outline

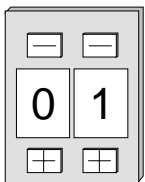
You have the option to specify the baudrate and the module-ID by setting the address selector to 00 within a period of 10s after you have turned the power on.

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.

## 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

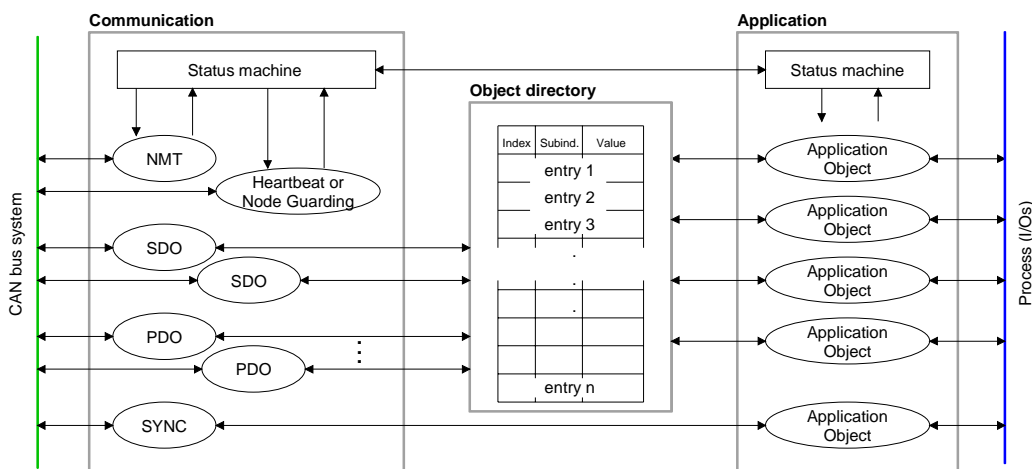


**Note!**

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.

## PDO - process data object

---

### 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 **process data objects** (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 resp. 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 resp. expected cyclically: after every "n<sup>th</sup>" 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".

## SDO - service data object

### SDO

The **S**ervice **D**ata **O**bject (SDO) serves the read or write access to the object directory. The CAN 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)

## 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 – 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
<b>Additional information device</b>	<b>Profile number</b>
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 Last error state to have occurred
	1	Actual error	Unsigned32	ro	N		
	... 254	...	... Unsigned32	... ro	... N		

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		
	2	Number of asynchronous PDOs supported	Unsigned32	ro	N		

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 $\mu$ s.

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 $\mu$ s.

### Device name

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

VIPA IM 253 1CA01 = VIPA CANopen slave IM253-1CA01

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

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

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: 0x64616663) 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	0xAFFEAF00	Vendor ID
	2	Product Code	Unsigned32	ro	N	0x2531CA01	Product Code
	3	Revision Number	Unsigned32	ro	N		Revision Number
	4	Serial Number	Unsigned32	ro	N		Serial Number

---

**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

I/O-Module type	Identification (hex)	No. of Digital-Input-Bytes	No. of Analog-Input-Bytes	No. of Digital-Output-Bytes	No. of Analog-Output-Bytes
DI 8xDC24V	0x9FC1	1	-	-	-
DI 16xDC24V	0x9FC2	2	-	-	-
DI 14xDC24V/2C	0x08C0	2	4	-	6
DI 32xDC24V	0x9FC4	4	-	-	-
DO 8xDC24V	0xAFC8	-	-	1	-
DO 16xDC24V	0xAFD0	-	-	2	-
DO 32xDC24V	0xAFD8	-	-	4	-
DIO 8xDC24V	0xBFC9	1	-	1	-
DIO 16xDC24V	0xBFD2	2	-	2	-
AI2x12Bit	0x15C3	-	4	-	-
AI4x12Bit	0x15C4	-	8	-	-
AI8x12Bit	0x15C5	-	16	-	-
AO4x12Bit	0xA5E0	-	-	-	8
AI2/AO2x12Bit	0x45DB	-	4	-	4
CP 240	0x1CC1	16	-	16	-
FM 250	0xB5F4	-	10	-	10
FM 250-SSI	0xB5DB	-	4	-	4
FM 253,FM 254	0x18CB	-	12	-	12

## Error Behavior

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1029	0	Error behavior	Unsigned8	ro	N	0x02	Number of Error Classes Communication Error
	1	Communication Error	Unsigned8	ro	N	0x00	
	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	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

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

The sub-index 2 contains the transmission type.

### Communication parameter RxPDO2

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1401	0	Number of Elements	Unsigned8	ro	N	0x02	Communication parameter for the first receive PDOs, sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0xC0000300 + NODE_ID	COB-ID RxPDO2
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

### Communication parameter RxPDO3

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1402	0	Number of Elements	Unsigned8	ro	N	0x02	Communication parameter for the first receive PDOs, sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0xC0000400 + NODE_ID	COB-ID RxPDO3
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

---

**Communication  
parameter RxPDO4**

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1403	0	Number of Elements	Unsigned8	ro	N	0x02	Communication parameter for the first receive PDOs, sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0xC0000500 + NODE_ID	COB-ID RxPDO4
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

---

**Communication  
parameter RxPDO5**

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1404	0	Number of Elements	Unsigned8	ro	N	0x02	Communication parameter for the first receive PDOs, sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0xC0000780 + NODE_ID	COB-ID RxPDO5
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

---

**Communication  
parameter RxPDO6**

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1405	0	Number of Elements	Unsigned8	ro	N	0x02	Communication parameter for the first receive PDOs, sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0xC0000240 + NODE_ID	COB-ID RxPDO6
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

---

**Communication  
parameter RxPDO7**

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1406	0	Number of Elements	Unsigned8	ro	N	0x02	Communication parameter for the first receive PDOs, sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0xC0000340 + NODE_ID	COB-ID RxPDO7
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

---

**Communication  
parameter RxPDO8**

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1407	0	Number of Elements	Unsigned8	ro	N	0x02	Communication parameter for the first receive PDOs, sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0xC0000440 + NODE_ID	COB-ID RxPDO8
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

---

**Communication  
parameter RxPDO9**

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1408	0	Number of Elements	Unsigned8	ro	N	0x02	Communication parameter for the first receive PDOs, sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0xC0000540 + NODE_ID	COB-ID RxPDO9
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

### Communication parameter RxPDO10

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1409	0	Number of Elements	Unsigned8	ro	N	0x02	Communication parameter for the first receive PDOs, sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0xC00007C0 + NODE_ID	COB-ID RxPD10
	2	transm. type	Unsigned8	rw	N	0xFF	Transmission type of the PDO

### Mapping 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	1 st mapped object	Unsigned32	rw	N	0x62000108	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	2	2 nd mapped object	Unsigned32	rw	N	0x62000208	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	...	...	...	...	...	...	...
	8	8 th 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	1 st mapped object	Unsigned32	rw	N	0x64110110	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	2	2 nd mapped object	Unsigned32	rw	N	0x64110210	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	...	...	...	...	...	...	...
	8	8 th mapped	Unsigned32	rw	N	0x00000000	(2 byte index, 1 byte sub-index, 1 byte bit-width)

The 2<sup>nd</sup> 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 3 rd to 10 th receive PDO; sub-index 0: number of mapped objects
	1	1 st 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	8 th mapped	Unsigned32	rw	N	0x00000000	(2 byte index, 1 byte sub-index, 1 byte bit-width)

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

### Communication parameter TxPDO1

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1800	0	Number of Elements	Unsigned8	ro	N	0x05	Communication parameter of the first transmit PDO, sub-index 0: number of following parameters
	1	COB-ID	Unsigned32	rw	N	0x80000180 + NODE_ID	COB-ID TxPDO1
	2	Transmission type	Unsigned8	rw	N	0xFF	Transmission type of the PDO
	3	Inhibit time	Unsigned16	rw	N	0x0000	Repetition delay [value x 100 µs]
	5	Event time	Unsigned16	rw	N	0x0000	Event timer [value x 1 ms]

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

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

### Communication parameter TxPDO2

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

### Communication parameter TxPDO3

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

### Communication parameter TxPDO4

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

### Communication parameter TxPDO5

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

### Communication parameter TxPDO6

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x1805	0	Number of Elements	Unsigned8	ro	N	0x05	Communication parameter for the 6 <sup>th</sup> 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 7 <sup>th</sup> 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 8 <sup>th</sup> 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 9 <sup>th</sup> 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 $\mu$ s]
	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 10 <sup>th</sup> 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 $\mu$ s]
	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	1 <sup>st</sup> mapped object	Unsigned32	rw	N	0x60000108	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	2	2 <sup>nd</sup> mapped object	Unsigned32	rw	N	0x60000208	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	...	...	...	...	...	...	...
	8	8 <sup>th</sup> mapped object	Unsigned32	rw	N	0x60000808	(2 byte index, 1 byte sub-index, 1 byte bit-width)

*continue ...*

---

**... continue**  
**Mapping TxPDO1**

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	1 st mapped object	Unsigned32	rw	N	0x64010110	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	2	2 nd mapped object	Unsigned32	rw	N	0x64010210	(2 byte index, 1 byte sub-index, 1 byte bit-width)
	...	...	...	...	...	...	...
	8	8 th mapped object	Unsigned32	rw	N	0x00000000	(2 byte index, 1 byte sub-index, 1 byte bit-width)

The 2<sup>nd</sup> 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 3 rd to 10 th transmit PDO; sub-index 0: number of mapped objects
	1	1 st 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	8 th mapped object	Unsigned32	rw	N	0x00000000	(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:

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

## PDO control

Index	Sub-index	Name	Type	Attr.	Map.	Default value	Meaning
0x2400	0	Number of Elements	Unsigned8	ro	N	0x0A	Time control for RxPDOs
	1	RxPDO1	Unsigned16	rw	N	0x0000	Timer value [ms]
	2	RxPDO2	Unsigned16	rw	N	0x0000	Timer value [ms]
	...	...	...	...	...	...	...
	10	RxPDO10	Unsigned16	rw	N	0x0000	Timer value [ms]

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

## 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 1 to 4	Unsigned32	rw	N	depending on the components fitted	Parameter bytes 1 to 4
	2	Prm 5 to 8	Unsigned32	rw	N	depending on the components fitted	Parameter bytes 5 to 8
	3	Prm 9 to 12	Unsigned32	rw	N	depending on the components fitted	Parameter bytes 9 to 12
	4	Prm 13 to 16	Unsigned32	rw	N	depending on the components fitted	Parameter bytes 13 to 16

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	1 st mapped object	Unsigned32	rw	N		
	...	...	...	...	...		
	0x40	8 th 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	8-bit digital input block	Unsigned8	ro	N	0x01	Number of available digital 8-bit input blocks
	0x01	1 st input block	Unsigned8	ro	Y		
	...	...	...	...	...		
	0x48	27 st input block	Unsigned8	ro	Y		



### 8bit polarity digital inputs

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6002	0x00	8-bit digital input block	Unsigned8	ro	N	0x01	Number of available digital 8-bit input blocks
	0x01	1 st input block	Unsigned8	rw	N	0x00	1 st polarity digital input block
	...	...	...	...	...	...	...
	0x48	72 nd input block	Unsigned8	rw	N	0x00	72 nd 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	16-bit digital input block	Unsigned8	ro	N	depending on the fitted components	Number of available digital 16-bit input blocks
	0x01	1 st input block	Unsigned16	ro	N		1 st digital input block
	...	...	...	...	...	...	...
	0x24	36 nd input block	Unsigned16	ro	N		36 nd digital input block

### 16bit polarity digital inputs

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6102	0x00	16-bit digital input block	Unsigned8	ro	N	depending on the components fitted	Number of available digital 16-bit input blocks
	0x01	1 st input block	Unsigned16	rw	N	0x0000	1 st polarity digital input block
	...	...	...	...	...	...	...
	0x24	36 input block	Unsigned16	rw	N	0x0000	36 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	32-bit digital input block	Unsigned8	ro	N	depending on the components fitted	Number of available digital 32-bit input blocks
	0x01	1 st input block	Unsigned32	ro	N		1 st digital input block
	...	...	...	...	...	...	...
	0x12	18 input block	Unsigned32	ro	N		18 digital input block

### 32bit polarity digital inputs

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6122	0x00	8-bit digital input block	Unsigned8	ro	N	depending on the components fitted	Number of available digital 32-bit input blocks
	0x01	1 st input block	Unsigned32	rw	N	0x00000000	1 st polarity digital input block
	...	...	...	...	...	...	...
	0x12	18 input block	Unsigned32	rw	N	0x00000000	18 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	8-bit digital output block	Unsigned8	ro	N	0x01	Number of available digital 8-bit output blocks
	0x01	1 st output block	Unsigned8	rw	Y		1 st digital output block
	...	...	...	...	...	...	...
	0x48	72 nd output block	Unsigned8	rw	Y		72 nd digital output block

### 8bit change polarity digital outputs

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6202	0x00	8-bit digital output block	Unsigned8	ro	N	0x01	Number of available digital 8-bit output blocks
	0x01	1 st output block	Unsigned8	rw	N	0x00	1 st polarity digital output block
	...	...	...	...	...	...	...
	0x48	72 nd output block	Unsigned8	rw	N	0x00	72 nd 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	8-bit digital output block	Unsigned8	ro	N	0x01	Number of available digital 8-bit output blocks
	0x01	1 st output block	Unsigned8	rw	N	0xFF	1 st error mode digital output block
	...	...	...	...	...	...	...
	0x48	72 nd output block	Unsigned8	rw	N	0xFF	72 nd 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	8-bit digital output block	Unsigned8	ro	N	0x01	Number of available digital 8-bit output blocks
	0x01	1 st output block	Unsigned8	rw	N	0x00	1 st error value digital output block
	...	...	...	...	...	...	...
	0x48	72 nd output block	Unsigned8	rw	N	0x00	72 nd error value digital output block

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

1 = Set output value to 0 if object 0x6206 is enabled.

0 = Set output value to 1 if object 0x6206 is enabled.

### 16bit digital outputs

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

### 16bit change polarity digital outputs

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

### 16bit error value digital outputs

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

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

1 = Set output value to 0 if object 0x6206 is enabled.

0 = Set output value to 1 if object 0x6206 is enabled.

### 32bit digital outputs

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

### 32bit change polarity digital outputs

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6322	0x00	32-bit digital input block	Unsigned8	ro	N	Depending on the components fitted	Number of available digital 32-bit output blocks
	0x01	1 st output block	Unsigned32	rw	N	0x00000000	1 st polarity digital output block
	...	...	...	...	...	...	...
	0x12	18 output block	Unsigned32	rw	N	0x00000000	18 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	32-bit digital input block	Unsigned8	ro	N	Depending on the components fitted	Number of available digital 32-bit output blocks
	0x01	1 st output block	Unsigned32	rw	N	0xFFFFFFFF	1 st error mode digital output block
	...	...	...	...	...	...	...
	0x48	18 output block	Unsigned32	rw	N	0xFFFFFFFF	18 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



### 32bit error value digital outputs

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

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

1 = Set output value to 0 if object 0x6206 is enabled.

0 = Set output value to 1 if object 0x6206 is enabled.

### Analog inputs

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

### Analog outputs

Index	Sub-Index	Name	Type	Attr.	Map.	Default value	Meaning
0x6411	0x00	2 byte output block	Unsigned8	ro	N	depending on the components fitted	Number of available analog outputs
	0x01	1 st output channel	Unsigned16	ro	Y		1 st analog output channel
	...	...	...	...	...	...	...
	0x24	24 th output channel	Unsigned16	ro	Y		24 th 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 components fitted	Number of available analog inputs
	0x01	Trigger 1 st input channel	Unsigned8	rw	N	0x07	Input interrupt trigger for 1 st analog input channel
	...	...	...	...	...	...	...
	0x24	Trigger 24 th input channel	Unsigned8	rw	N	0x07	Input interrupt trigger for 24 th 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 interruption. Bits set refer to the number of the channel that caused the interruption. The bits are automatically reset, after they have been read by a SDO or send by a PDO.

1 = interruption produced  
0 = interruption 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 by setting an inhibit time (object 0x1800ff, sub-index 3) and/or limit value monitoring (objects 0x6424 + 0x6425) and/or a 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 components fitted	Number of available analog inputs
	0x01	Upper limit 1 st input channel	Unsigned32	rw	N	0x00000000	Upper limit value for 1 st analog input channel
	...	...	...	...	...	...	...
	0x24	Upper limit 24 th input channel	Unsigned32	rw	N	0x00000000	Upper limit value for 24 th 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 components fitted	Number of available analog outputs
	0x01	1 st analog output block	Unsigned8	rw	N	0xFF	1 st error mode analog output block
	...	...	...	...	...	...	...
	0x24	36 th analog output block	Unsigned8	rw	N	0xFF	36 th 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 = reverts to error value 0x6444

### Analog output error value

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

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

## NMT - network management

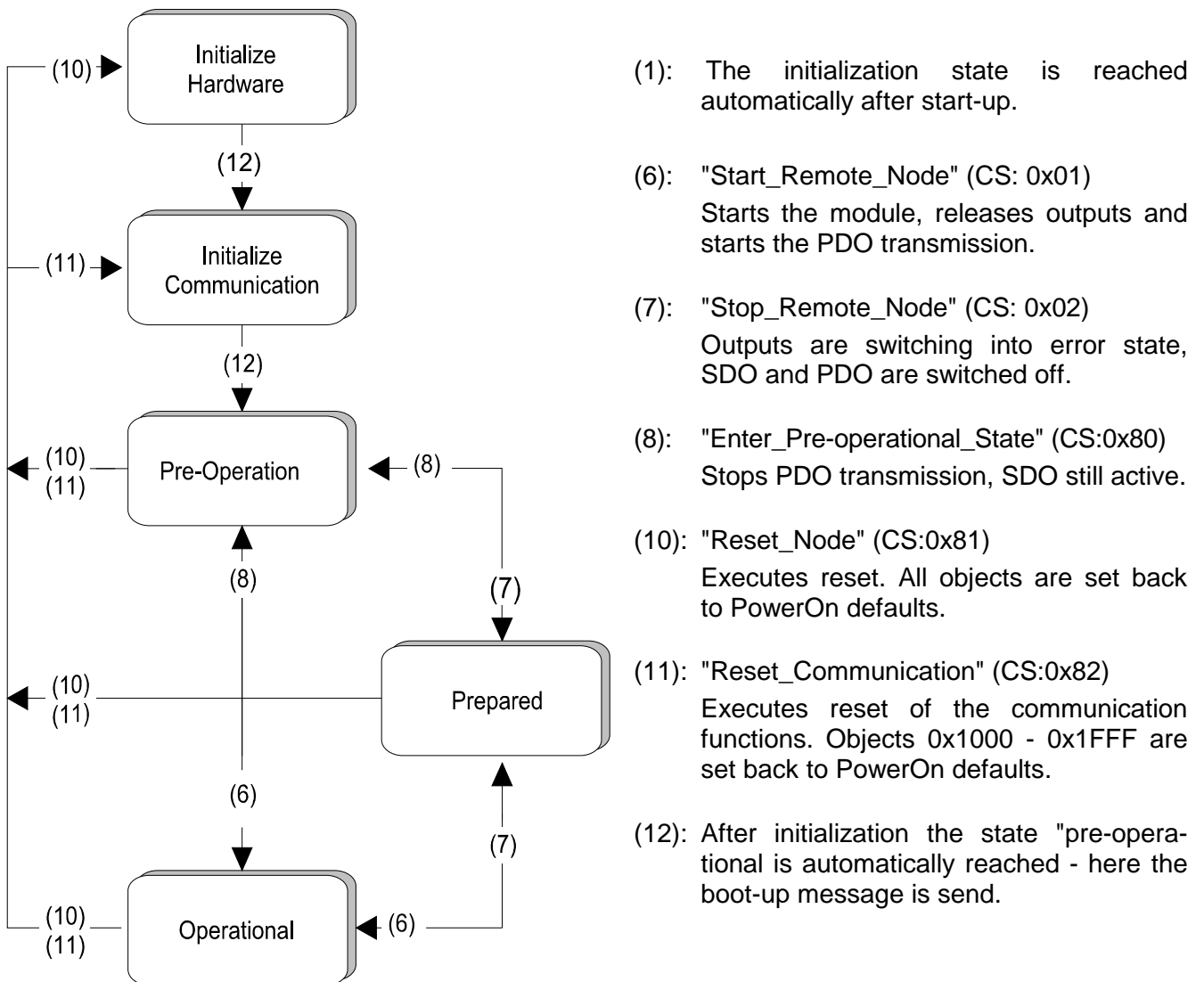
Network management (NMT) provides the global services specifications for network supervision and management. This includes the login and logout of the different network devices, the supervision of these devices as well as the processing of exceptions.

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

The 1<sup>st</sup> data byte contains the NMT command specifier: **CS**.

The 2<sup>nd</sup> 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:





---

**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  $0x700 + \text{module-ID}$  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 ( $0x700 + \text{module-ID}$ ) 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 coupler IM 253CAN

Electrical data	VIPA 253-1CA01
Power supply	DC 24V (20.4 ... 28.8) via front from ext. power supply
Current consumption	max. 700mA
Output current backplane bus	max. 3.5A
Isolation	≥ AC 500V
Status indicator	by means of LEDs located on the front
Connectors/interfaces	9pin D-type (socket) CAN-Bus connection
CAN-Bus interface	
Connection	9pin D-type plug
Network topology	Linear bus, active bus termination at one end, tap lines permitted.
Medium	Screened three-core cable, unshielded 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.
Max. no. of stations	127 stations (depending on the master interface)
Combination with peripheral modules	
max. no. of modules	32 (depending on current consumption)
max. inputs/outputs	80Byte each (80Byte = 10 PDOs à 8Byte)
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	80g

**CANopen coupler**  
**IM 253CAN,**  
**DO 24xDC 24V**

Electrical data	VIPA 253-2CA20
Power supply	DC 24V (20.4 ... 28.8) via front from ext. power supply
Current consumption at L+	max. 800mA
Output current backplane bus	3.5A
Isolation	≥ AC 500V
Status indicator	by means of LEDs located on the front
Connectors/interfaces	9pin D-type (socket) CAN-Bus connection
CAN-Bus interface	
Connection	9pin D-type plug
Network topology	Linear bus, active bus termination at one end, tap lines permitted.
Medium	Screened three-core cable, unshielded 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)
Output unit	
Number of outputs	24
Nominal load voltage	DC 24V (18...35V) internal via CAN coupler
Output current per channel	0.5A (Total current max. 4A)
Status monitor	Power (PW) fuse ok, Error (ER) short circuit, overload
Programming data	
Output data	3Byte
Dimensions and weight	
Dimensions (WxHxD) in mm	50.8x76x76
Weight	150g



## Chapter 5 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, the procedure for connecting the DeviceNet coupler to the Profibus and the technical data conclude the chapter.

Below follows a description of:

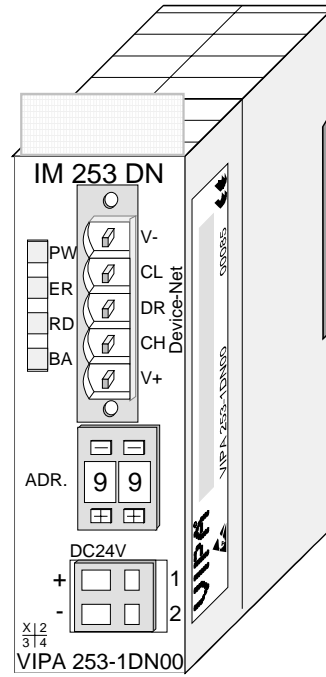
- DeviceNet principles
- Hardware description of the VIPA DeviceNet coupler IM 253DN
- Configuration by means of the *DeviceNet-Manager* inc. examples
- Diagnostics
- Interfacing options for Profibus
- Technical data

### Content

Topic	Page
<b>Chapter 5 DeviceNet .....</b>	<b>5-1</b>
System overview .....	5-2
Principles .....	5-3
IM 253DN - DeviceNet coupler - Construction .....	5-5
Configuration by means of the DeviceNet-Manager.....	5-8
Specifying baudrate and DeviceNet address .....	5-9
Test in conjunction with the DeviceNet .....	5-10
Module configuration in the DeviceNet-Manager .....	5-11
I/O addressing of the DeviceNet scanner .....	5-16
Diagnostics .....	5-17
Profibus interface.....	5-22
Technical data .....	5-23

## System overview

You can use the VIPA DeviceNet coupler to link-up up to 32 modules (of 40Byte each) 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

## Principles

### 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.

<b>Communication medium</b>	<p>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.</p> <p>The length of the tap lines can be up to 6m while the total length of all spur lines depends on the baudrate.</p> <p>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.</p> <p>DeviceNet employs a screened five-core cable as data communication medium.</p> <p>DeviceNet uses voltage differences and for this reason it exhibits less sensitivity to interference than a voltage or current based interface.</p> <p>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.</p>
<b>Bus access method</b>	<p>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).</p> <p>The exchange of messages is message orientated and not station orientated. Each message is provided with a unique and prioritizing identifier. At any time only one station is able to occupy the bus with its messages.</p> <p>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.</p>
<b>Addressing</b>	<p>All stations on the bus must be uniquely identified by means of an ID address. Every DeviceNet device has addressing facilities.</p>
<b>EDS file</b>	<p>The properties of the DeviceNet units are supplied in the form of an EDS file (<b>E</b>lectronic <b>D</b>ata <b>S</b>heet) to configure a slave interface by means of your configuration tool.</p>



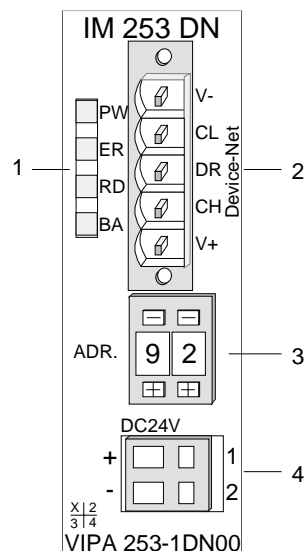
## IM 253DN - DeviceNet coupler - Construction

The DeviceNet coupler IM 253DN provides a simple method of interfacing any decentral peripheral modules by means of the DeviceNet protocol.

### Properties

- 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*
- Profibus–DeviceNet conversion is possible by combining the unit with a IM 208DP

### Front view 253-1DN00



- [1] LED status indicator
- [2] DeviceNet connector
- [3] Address selector
- [4] DC 24V power supply connector

**Components**

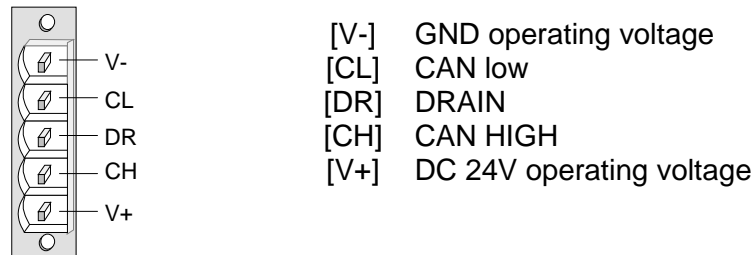
**LEDs**

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	yellow	Power-LED: supply voltage available
ER	red	DeviceNet or backplane bus bus error
RD	green	Backplane bus status
BA	yellow	DeviceNet status

**DeviceNet interfacing**

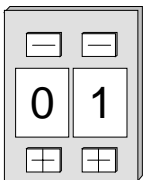
The DeviceNet connection is provided by a 5pin Open Style connector. The pin assignment is imprinted on the front of the module.



**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 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.

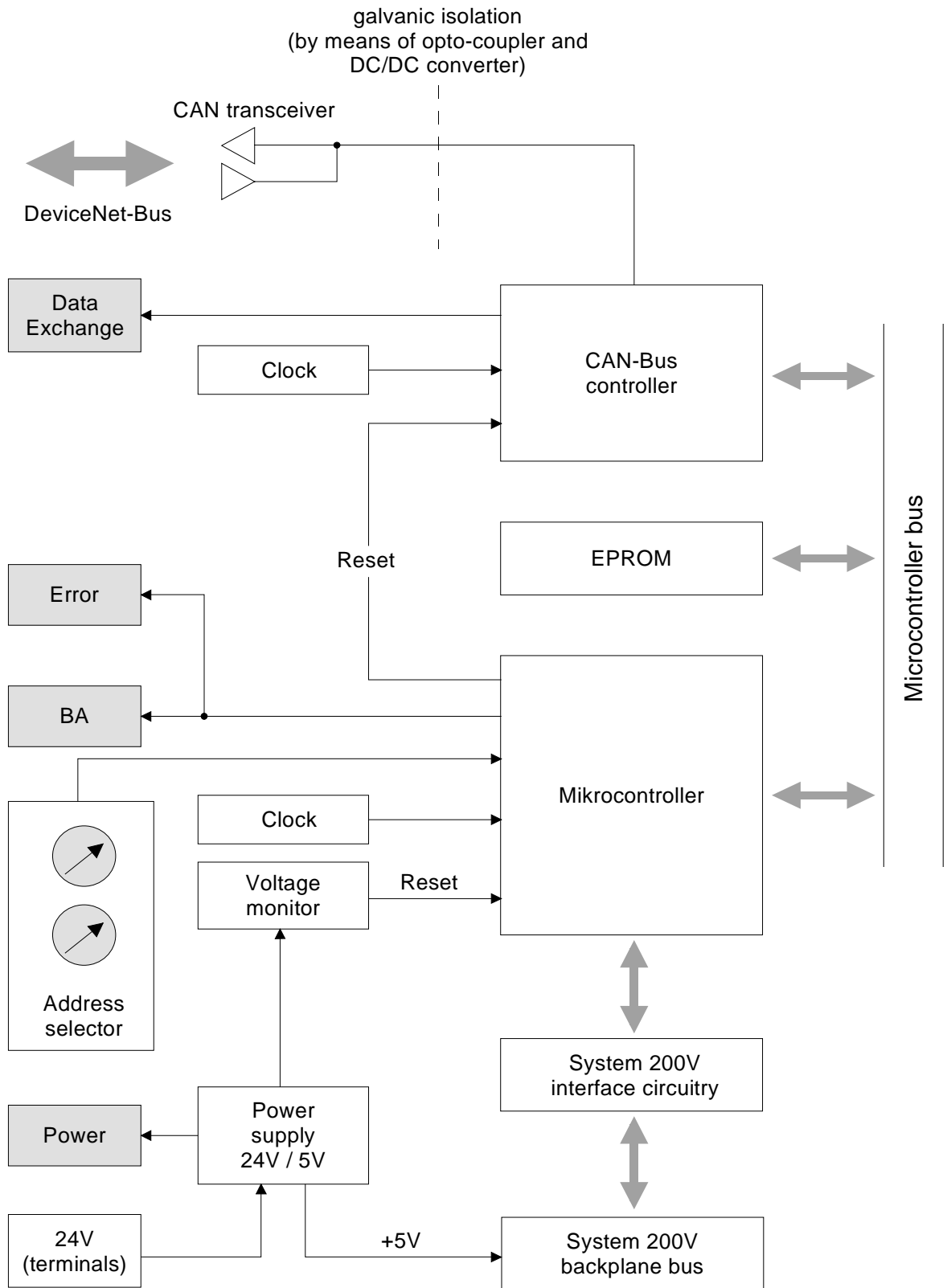


**Note!**

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 address is 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

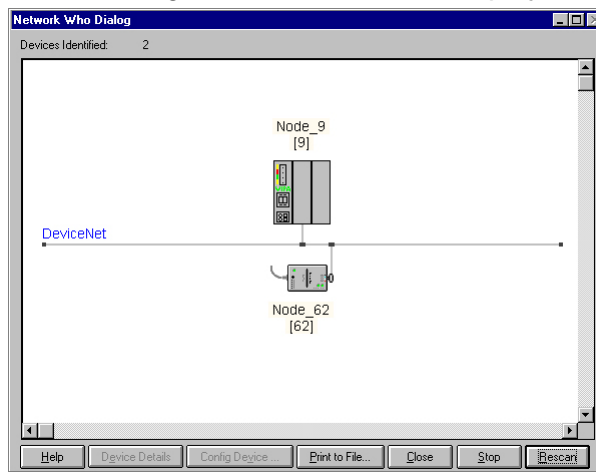
When the address is not valid or if it already exists the ER-LED (red) will be turned on after power on.

## 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

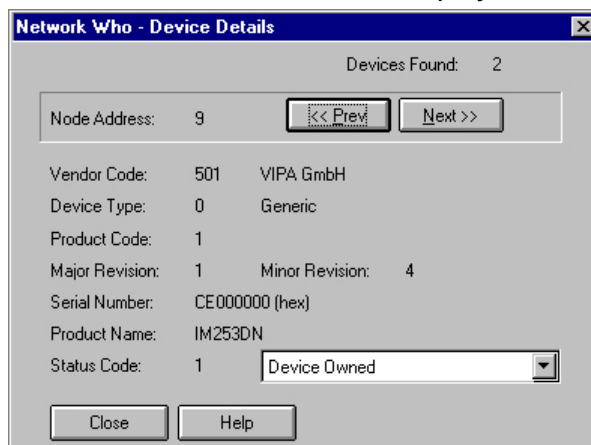
*The following network windows is displayed:*



### 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:

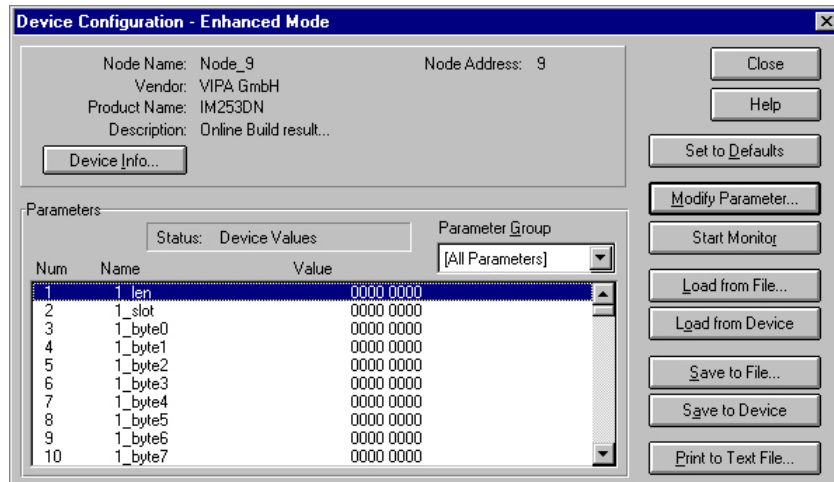
- DeviceNet manages the parameter data in groups.
- Every DeviceNet coupler is able to process and store a maximum of 144Byte of parameter data.
- These 144Byte are divided into 8 groups of 18Byte each.
- Every group can contain the parameter data of 1 module.
- Groups are identified by a prefix-No. (1...8) in the parameter name.
- The number of parameter bytes is defined in the parameter "Len" (1<sup>st</sup> 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<sup>nd</sup> parameter)
- The values may be entered as bit patterns by double-clicking a parameter
- Unused groups are identified by a "Value" 0000 0000.

### Approach

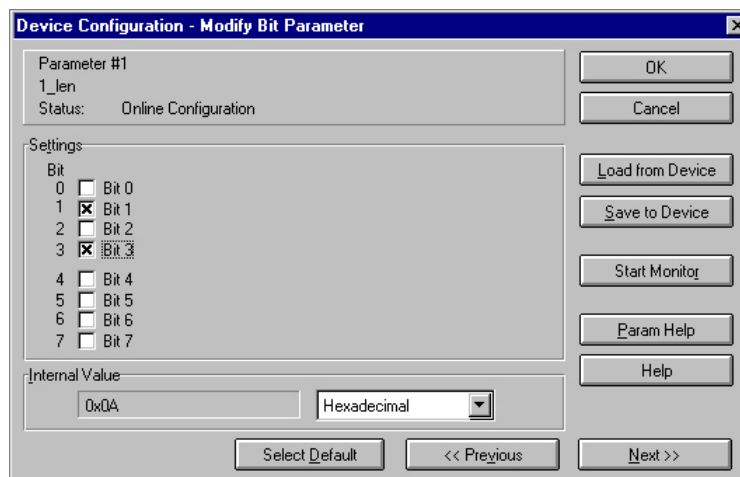
Precondition: The IM 253DN coupler is active on the bus.

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:*

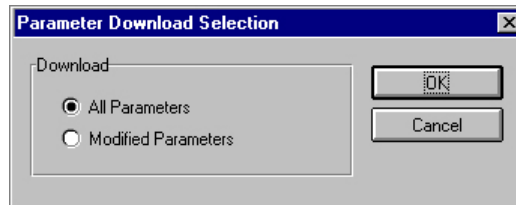


- Enter the number of parameter bytes (bit coded) of the module that you are configuring. You can obtain the number from the documentation for the peripheral module. Set or reset the respective bits by clicking the checkbox.
- Click [OK] to close the mask. The next parameter (slot) of the same group 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.

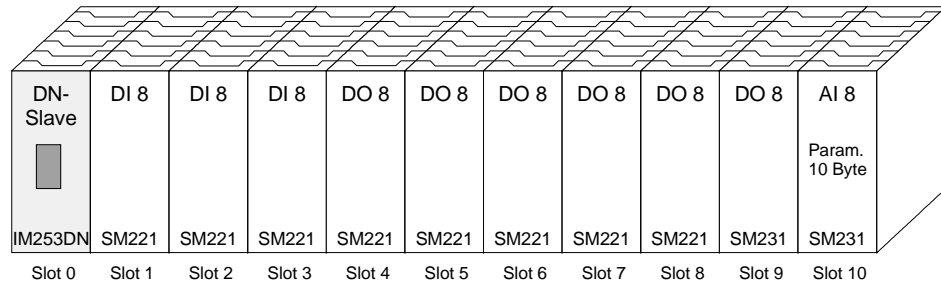
- During the transfer the status text "Status: downloading" is displayed. When the transfer has completed, the status text changes to "Status: Device Values"
- If you were to request the "Device Details", you may see that the bit CONFIGURED is now also included in the status.



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:

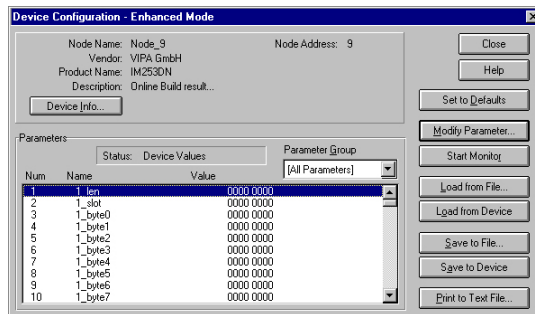


The example shows a DeviceNet coupler with 10 modules; however, the modules installed in plug-in locations 1 to 9 can not be configured.

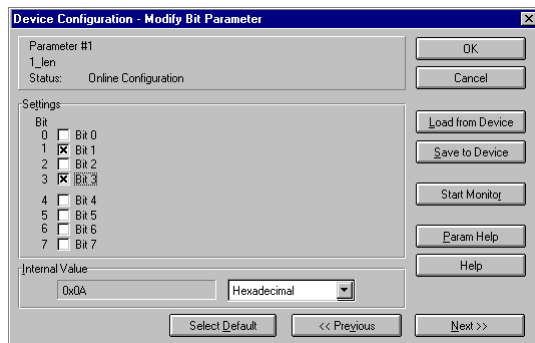
Below follows the description of the configuration of the analog-module in location 10:

- 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 0 ... 5: 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
<input type="checkbox"/> off	LED turned off
<input type="checkbox"/> on	LED is permanently on
<input checked="" type="checkbox"/> 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
<input checked="" type="checkbox"/> PW on <input type="checkbox"/> ER off <input checked="" type="checkbox"/> RD blinks <input type="checkbox"/> 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
<input checked="" type="checkbox"/> PW on <input checked="" type="checkbox"/> ER on <input type="checkbox"/> RD off <input type="checkbox"/> 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
<input checked="" type="checkbox"/> PW on <input type="checkbox"/> ER off <input checked="" type="checkbox"/> RD blinks <input checked="" type="checkbox"/> BA blinks	<p>After POWER ON the PW-LED is turned on.</p> <p>The RD-LED blinks because:</p> <ul style="list-style-type: none"> <li>the backplane bus is operating properly</li> <li>the configuration data was transferred successfully from the EEPROM into the configurable peripheral modules.</li> </ul> <p>The BA-LED blinks because:</p> <ul style="list-style-type: none"> <li>at least one additional device is active on the DeviceNet,</li> <li>and the address set up on the coupler is unique.</li> </ul>
<input checked="" type="checkbox"/> PW on <input checked="" type="checkbox"/> ER on <input type="checkbox"/> RD off <input type="checkbox"/> BA off	<p>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:</p> <ul style="list-style-type: none"> <li>bad address or address occupied by another device</li> <li>data transfer rate is bad.</li> </ul>
<input checked="" type="checkbox"/> PW on <input checked="" type="checkbox"/> ER on <input checked="" type="checkbox"/> RD blinks <input checked="" type="checkbox"/> BA blinks	<p>After POWER ON the PW-LED is on.</p> <p>The ER-LED is turned on when the configuration data could not be transferred into the configurable peripheral module.</p> <p>The RD-LED blinks because</p> <ul style="list-style-type: none"> <li>the backplane bus is operating properly</li> <li>the configuration data was not transferred into the configurable peripheral modules.</li> </ul> <p>The BA-LED blinks because</p> <ul style="list-style-type: none"> <li>at least one other device is active on the DeviceNet,</li> <li>the address set up on the coupler is unique.</li> </ul>

**POWER ON with DeviceNet and master**

LED	Description
<input type="checkbox"/> PW on <input type="checkbox"/> ER on <input checked="" type="checkbox"/> RD blinks <input type="checkbox"/> BA on	<p>After POWER ON the PW-LED is on.</p> <p>The ER-LED is turned on since the configuration data was not transferred into the configurable peripheral modules.</p> <p>The RD-LED blinks because</p> <ul style="list-style-type: none"> <li>• the backplane bus operates properly</li> <li>• the configuration data was not transferred into the configurable peripheral modules.</li> </ul> <p>The BA-LED is turned on</p> <ul style="list-style-type: none"> <li>• because the coupler IM 253DN has established a DeviceNet-connection to a master.</li> </ul> <p>Note!</p> <p>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.</p> <p>It is possible to distinguish between these cases:</p> <ul style="list-style-type: none"> <li>• by interruption of the DeviceNet connection → ER-LED and RD are blinking!</li> <li>• with a network WHO in the <i>DeviceNet-Manager</i> → in case of a hardware error the IM253DN will not appear on the network</li> </ul> <p>Please call the VIPA hotline if a hardware error occurs!</p>

**Proper operation with DeviceNet and master**

LED	Description
<input type="checkbox"/> PW on <input type="checkbox"/> ER off <input checked="" type="checkbox"/> RD on <input type="checkbox"/> BA on	<p>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.</p> <p>The BA-LED is turned on because the coupler IM 253DN established a DeviceNet connection with a master.</p>

**Errors during the operation with DeviceNet and master**

LED	Description
<input checked="" type="checkbox"/> PW on	<p>After POWER ON the PW-LED is on.</p> <p>The ER-LED is turned on because an error was detected on the backplane bus.</p> <p>The BA-LED is turned on because the IM 253DN coupler established a DeviceNet connection with a master.</p> <p>Note! The IM 253DN coupler will execute a reset after 30s.</p>
<input checked="" type="checkbox"/> ER on	
<input type="checkbox"/> RD off	
<input checked="" type="checkbox"/> BA on	

**Change of state from operational to module error status**

LED	Description
<input checked="" type="checkbox"/> PW on	<p>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.</p>
<input checked="" type="checkbox"/> ER on	
<input type="checkbox"/> RD off	
<input type="checkbox"/> BA off	

**Indicators after a re-start and a reset**

LED	Description
<input checked="" type="checkbox"/> PW on	<p>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.</p> <p>All Allen - Bradley scanners will display message #77.</p>
<input checked="" type="checkbox"/> ER on	
<input checked="" type="checkbox"/> RD blinks	
<input checked="" type="checkbox"/> BA on	
<input checked="" type="checkbox"/> PW on	<p>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.</p> <p>All Allen - Bradley scanners will display message #77.</p>
<input type="checkbox"/> ER off	
<input checked="" type="checkbox"/> RD on	
<input checked="" type="checkbox"/> BA on	



**Change of state from operational to connection error status**

LED	Description
<input type="checkbox"/> PW 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.
<input checked="" type="checkbox"/> ER blinks	
<input checked="" type="checkbox"/> RD blinks	
<input type="checkbox"/> BA on	

**Configuration mode****POWER ON in configuration mode**

LED	Description
<input type="checkbox"/> PW on	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.
<input type="checkbox"/> ER off	
<input checked="" type="checkbox"/> RD on	
<input type="checkbox"/> BA off	

**Device error**

LED	Description
<input type="checkbox"/> PW on	The address that was set up on the coupler is not valid. Change the address to a valid setting: <ul style="list-style-type: none"> <li>• 0...63 as DeviceNet address</li> <li>• 90...92 for the definition of the baudrate</li> </ul>
<input checked="" type="checkbox"/> ER on	
<input type="checkbox"/> RD off	
<input type="checkbox"/> BA off	
<input type="checkbox"/> PW 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. <p>Note!</p> Errors that occur during POWER ON with DeviceNet and master display the same combination of LEDs as a hardware error. <p>It is possible to distinguish between these cases:</p> <ul style="list-style-type: none"> <li>• by interruption of the DeviceNet connection → ER-LED and RD are blinking!</li> <li>• with a network WHO in the <i>DeviceNet-Manager</i> → in case of a hardware error the IM 253DN will not appear on the network</li> </ul> Please call the VIPA hotline if a hardware error occurs!
<input checked="" type="checkbox"/> ER on	
<input checked="" type="checkbox"/> RD on	
<input type="checkbox"/> BA on	

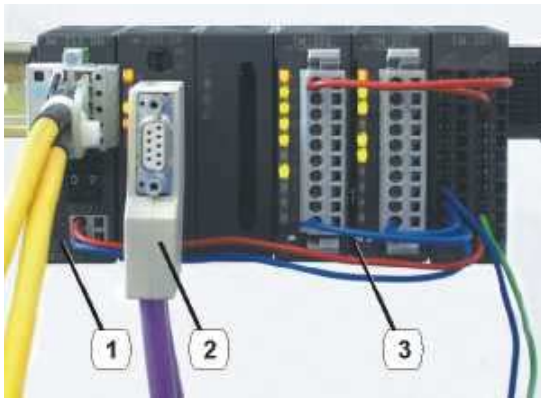
## Profibus interface

### Description

The modular System 200V may be very easily used to establish a DeviceNet–Profibus bridge. The Profibus master is simply installed on the backplane bus together with the DeviceNet coupler.

The connection from the DeviceNet to Profibus-DP is able to transfer 256Byte of input and 256Byte of output data.

In cases where the maximum quantity of data is not used, it is also possible to install peripheral modules in addition to the Profibus master.



- [1] DeviceNet coupler  
IM 253DN
- [2] Profibus master  
IM 208DP(0)
- [3] Additional  
peripheral modules

### Example

You want to provide a link between DeviceNet and Profibus-DP. The following 4 modules were installed into the backplane bus:

Location (hex)	Installed module	I/O-data and addresses
Slot 0	DeviceNet coupler IM 253DN	-
Slot 1	Profibus master IM 208DP	Input as of address 0 Output as of address 2
Slot 2	Digital Out SM 222	1Byte, address 0
Slot 3	Digital Out SM 222	1Byte, address 1

### Approach

- Assemble your system by installing the Profibus master IM 208DP to the right of the DeviceNet coupler, followed by the 2 output modules (see figure).
- Please ensure that the addresses of the directly installed peripheral modules have been reserved in your Profibus configuration tool. For details refer to the documentation on your Profibus master.

The peripheral modules connected via Profibus-DP and the output modules exchange data with the Profibus master. This communicates with the DeviceNet coupler via the backplane bus.

## Technical data

### DeviceNet coupler

#### IM 253DN

Electrical data	VIPA 253-1DN00
Power supply	DC 24V (20.4 ... 28.8) via front from ext. power supply
Current consumption	Bus coupler: 50mA incl. supply to the peripheral modules: 800mA max.
Output current backplane bus	3.5A
Isolation between DeviceNet and backplane bus	500V rms
Function specific data	
Status indicator	by means of LEDs on the front
Physical connection to DeviceNet	5pin Open Style connector
Network topology	Linear bus, tap lines up to 6m length
Communication medium	Screened 5core cable
Communication rate	125, 250, 500kBaud
Overall length of the bus	up to 500m
Number of stations	max. 64
Combination with peripheral modules	
Number of modules	max. 32
Inputs	max. 256Byte
Outputs	max. 256Byte
Dimensions and Weight	
Dimensions (BxHxT)	25.4x76x76mm
Weight	80g



## Chapter 6 SERCOS

### 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 principles
- 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
<b>Chapter 6 SERCOS</b> .....	<b>6-1</b>
System overview .....	6-2
Principles .....	6-3
IM 253Sercos - SERCOS coupler - Construction .....	6-5
Basic parameterization via address adjuster .....	6-8
SERCOS Identifier .....	6-10
Example for the automatic ID assignment .....	6-13
Technical Data .....	6-22



### Note!

For the deployment of the SERCOS coupler in this chapter, a thorough knowledge of SERCOS is required.

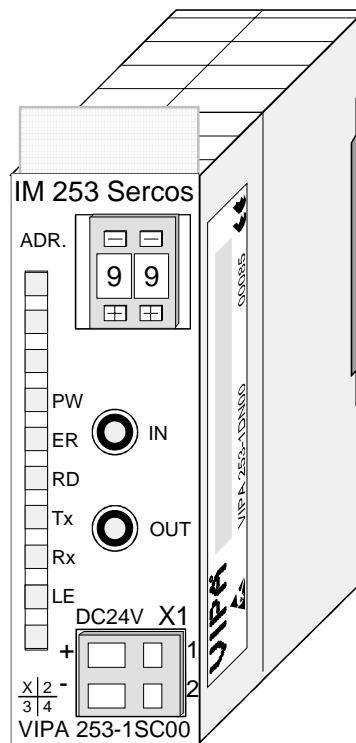
This manual describes exclusively the VIPA specific properties.

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

## Principles

### SERCOS

SERCOS means **S**erial **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 proved 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**

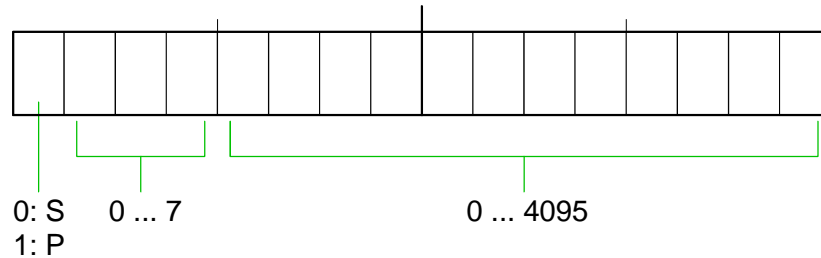
All participants at the bus must be identified by an unique address. Every SERCOS device has the option to fix the address.

**ID number for data exchange**

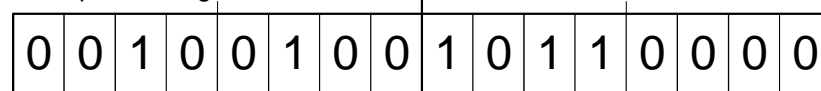
The addressing of the data at the demand control data exchange and the definition of the real-time data happens with SERCOS via ident numbers. For the ID numbers, are value range of  $2^{16}$  is fixed, divided into two areas:

- 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





## IM 253Sercos - SERCOS coupler - Construction

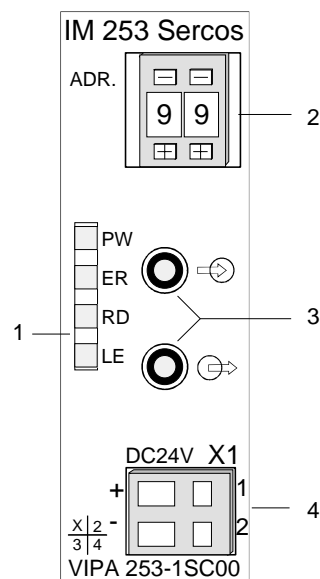
### Properties

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

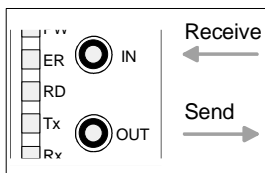
## Components

### 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

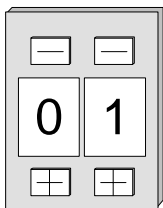


Via this jack you include the SERCOS coupler via FO transmitters into your SERCOS.

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

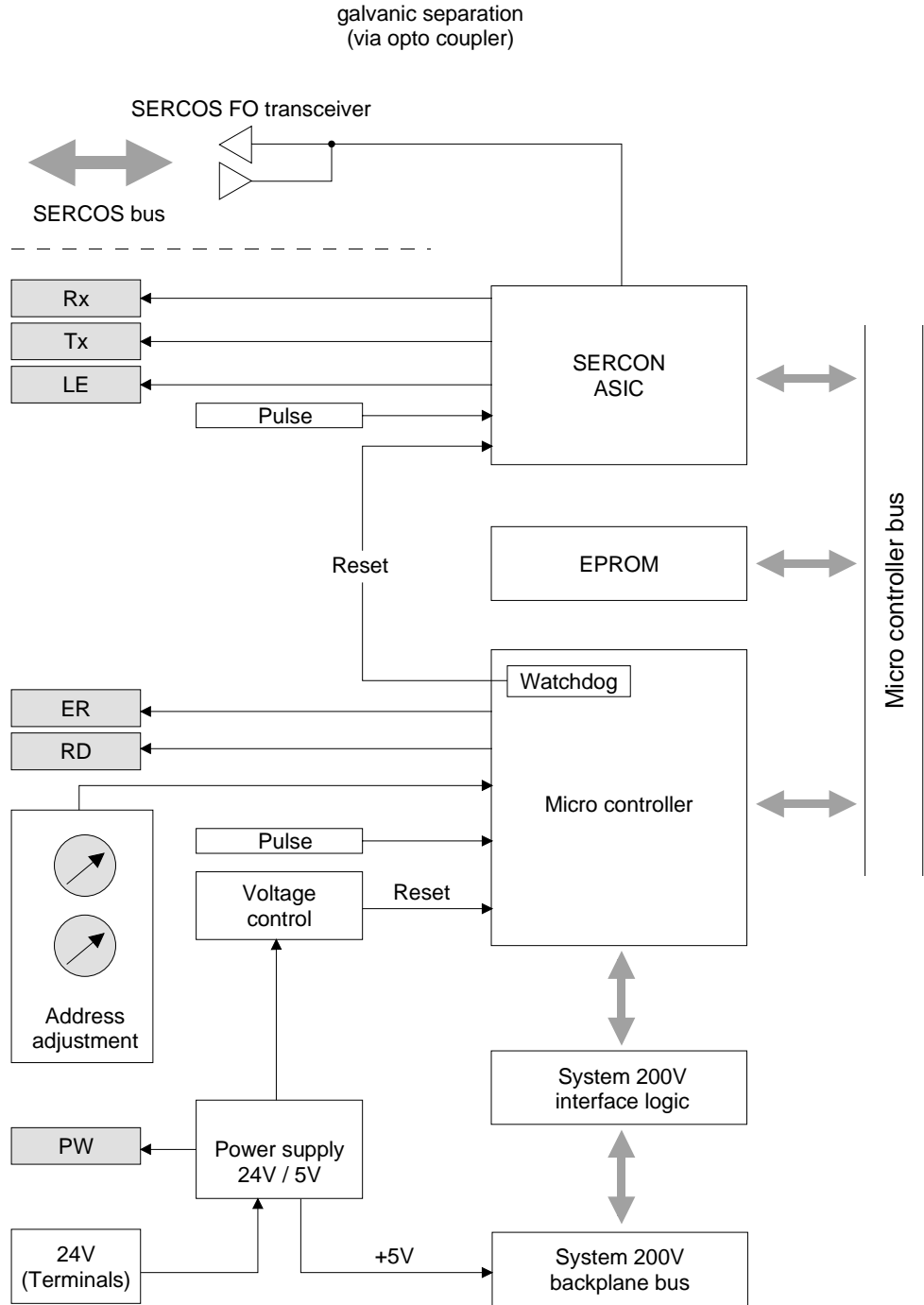
The SERCOS coupler has an integrated power supply, protected against inverse polarity and overcurrent.

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

**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)

The length value and a description of the parameters to transfer are to find in the according chapters of the modules in this manual.

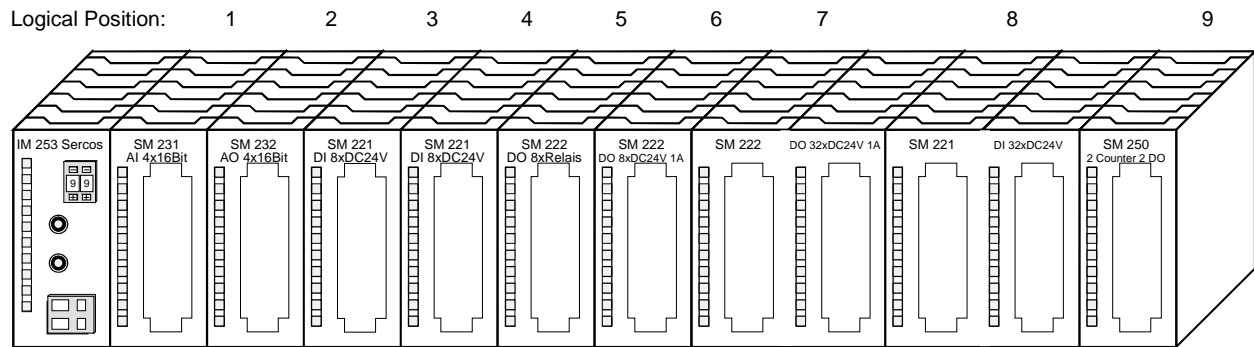


## Example for the automatic ID assignment

### 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 232-1BD50 (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: 10 Byte

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 $\pm 10V$ in the S7 format from Siemens	28h = 40dez
3	Function no. channel 1	Voltage $\pm 10V$ 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	1.PARAMETER.10	are created but not used
...	...	
P-0-0117	1.PARAMETER.15	

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 (232-1BD50) 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 $\pm 10V$ in the S7 format from Siemens	09h = 9dez
3	Function no. channel 1	Voltage $\pm 10V$ 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	2.PARAMETER.6	are created but not used
...	...	
P-0-0217	2.PARAMETER.15	

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	9.PARAMETER.2	are created but not used
...	...	
P-0-0917	9.PARAMETER.15	

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	DC 24V (20.4 ... 28.8V) via front by ext. pow. supply
Current consumption	Bus coupler: 50mA incl. supply of the peripheral modules: max. 3.5A (5V)
Output current backplane bus	max. 3.5A
Potential separ. to the backplane bus	500V eff.
Function specific data	
Status indicator	via LED at the frontside
Physical connection SERCOS	FO jacks
Network topology	Ring
Transfer medium	Fiber optic transmitter, for use with 1mm Plastic Optical Fiber and 200µm Hard Clad Silica HCS®
Transfer rate	2, 4, 8, 16MBaud
Number of participants	max. 89
Combination with peripheral modules	
Module number	max. 32
Inputs	max. 256Byte
Outputs	max. 256Byte
Dimensions and Weight	
Dimensions (WxHxD)	25.4x76x76mm
Weight	75g

## Chapter 7 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 principles. 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
- Principles of the Ethernet communication
- Construction of the Ethernet coupler
- Principles of the automatic address allocation
- (Online-)access to the Ethernet coupler
- Programming sample
- Technical data

### Content

Topic	Page
<b>Chapter 7 Ethernet coupler .....</b>	<b>7-1</b>
System overview .....	7-2
Principles of Ethernet.....	7-3
Planning a network .....	7-7
IM 253NET - Ethernet coupler - Construction .....	7-9
Access to the Ethernet coupler .....	7-11
Principle of the automatic address allocation .....	7-14
Project engineering under WinNCS .....	7-15
Diagnosis and test via Internet Browser.....	7-16
ModbusTCP.....	7-20
Modbus function codes.....	7-21
Siemens S5 Header Protocol.....	7-25
Programming sample.....	7-27
Technical data .....	7-28

## 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.

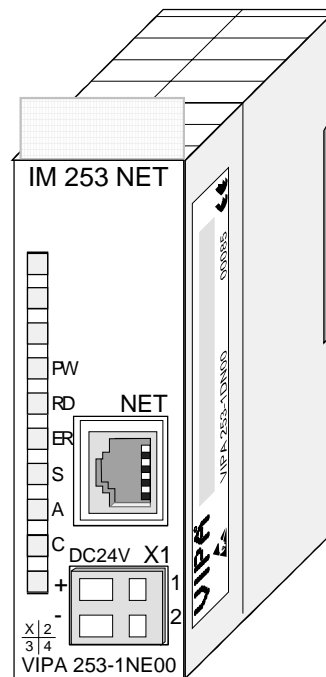
Slave systems on the other hand are "data collectors" that deliver the I/O data of the connected modules to the requesting master.

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

## Principles of Ethernet

### 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 or 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.

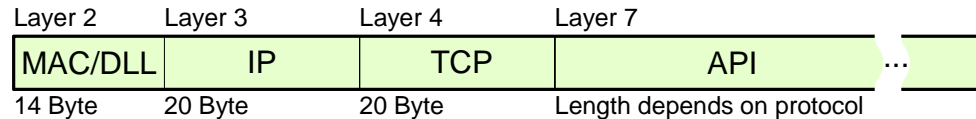
For test and diagnostic purposes the Ethernet slave provides a web server that allows the read and write access to the I/O periphery as well as the parameterization of the modules.

**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****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 (Transmission Control Protocol) 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**

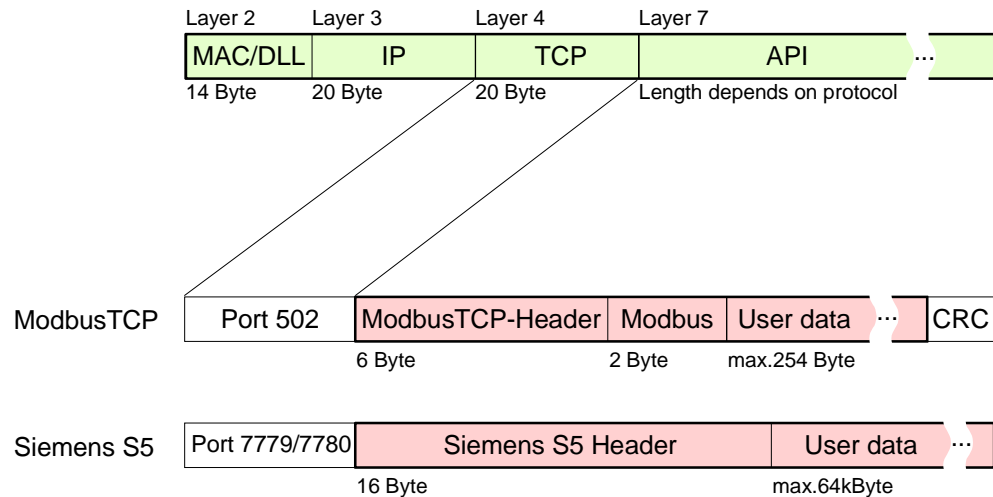
API means **A**pplication **P**rogramming **I**nterface. API covers the conditions of the Application Layer (Layer 7).

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.

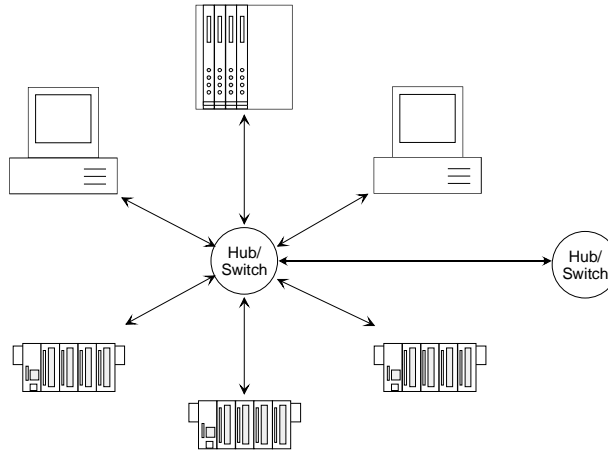
### Standards and norms

The following standards and norms about network technologies have been fixed by international and national committees:

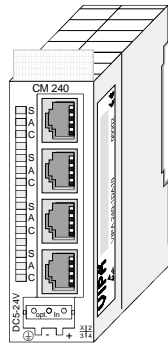
ANSI	American National Standards Institute The ANSI X3T9.5 standard currently defines the provisions for high speed LAN's (100 MB/s) based on fiber optic technology. (FDDI) Fiber Distributed Data Interface.
CCITT	Committee Consultative Internationale de Telephone et Telegraph. Amongst others, this advisory 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 - Construction

### 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.
- Integrated web server for test and diagnosis
- RJ45 jack 100BaseTX, 10BaseT
- Automatic polarity and baud rate 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

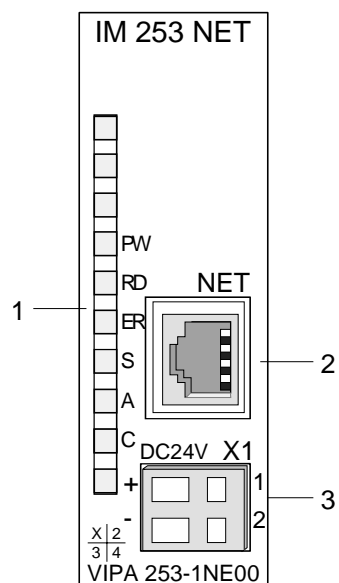


### Attention!

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 208DP



- [1] LED Status monitor
- [2] RJ45 jack for Twisted Pair
- [3] DC 24V voltage supply

## Components

### LEDs

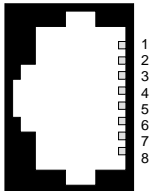
The Ethernet coupler has different LEDs for diagnosis and monitoring the operational state. The usage and meaning of the colors are described in the following table.

Label	Color	Description
PW	Yellow	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	-

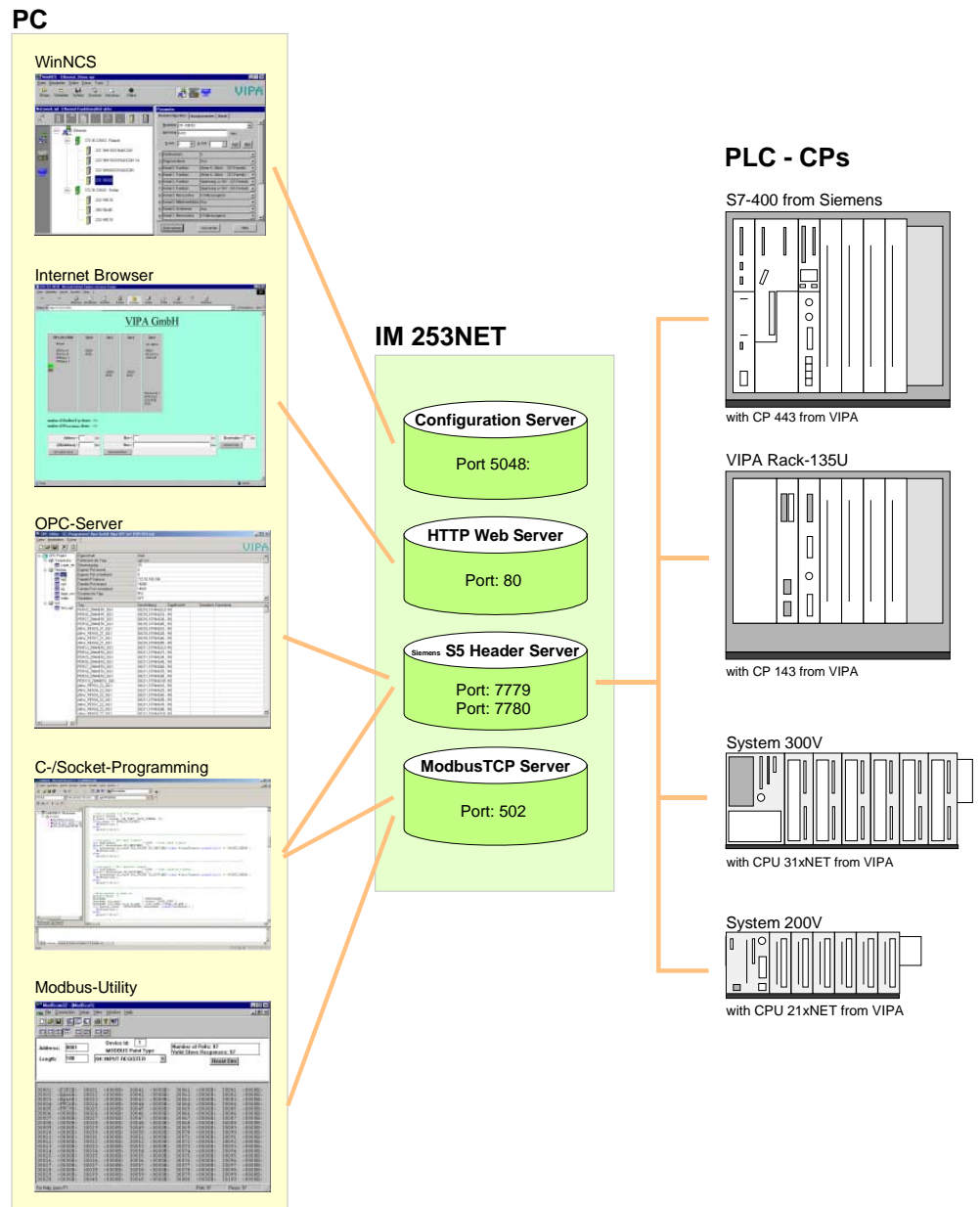
### Power supply

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](http://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.

### 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.



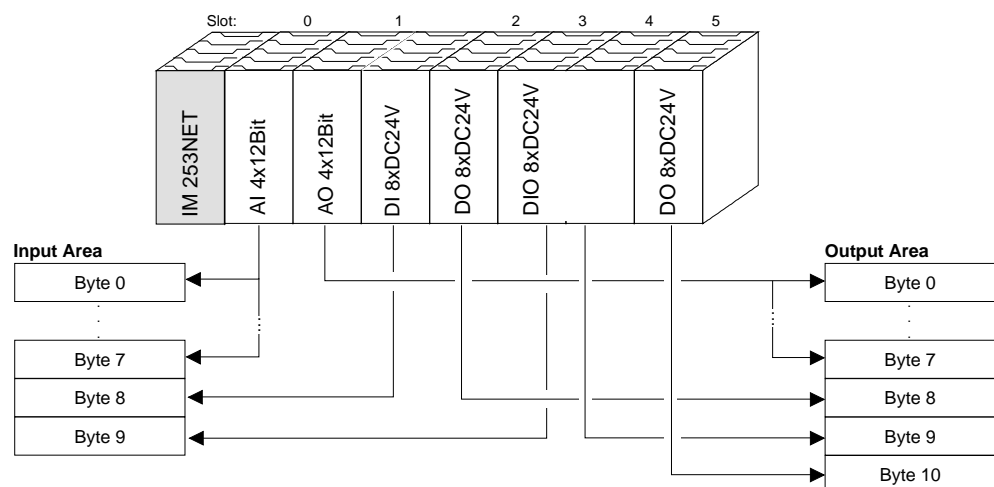
### Note!

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 VIPA2ETH.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: VIPA2ETH.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
- Diagnosis in-/output periphery
- Information about connected clients
- Elements for active access to the Ethernet coupler

Diagnosis Ethernet coupler	Diagnosis In-/Output periphery				
VIPA 253-1NE00	Slot 0	Slot 1	Slot 2	Slot 3	Slot 4
Station A	221-1BH10	222-1BH10	221-1BH10	223-2BL10	231-1BD52
HWVer: 10 PLDVer: 10	IB[0]= 00 00		IB[2]= 00 00	IB[4]= 00 00	IB[6]= 00 00 00 00 00 00 00 00
FWMajor: 1 FWMinor: 3		QB[0]= 00 00		QB[2]= 00 00	
<input type="checkbox"/> RDY <input type="checkbox"/> ERR					Prm(len10)= 00 00 2d 2d 28 28 00 00 00 00

#### Information about connected clients

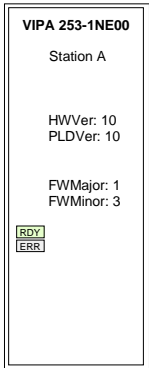
**Number of Modbus/TCP clients:**<2>: [172.16.131.31] [172.16.131.55]  
**Number of S5 from Siemens clients:** <1>: [172.16.131.10]

#### Elements for the active access to the Ethernet coupler

Address = <input type="text"/> <input type="button" value="dec"/>	Slot = <input type="text"/> <input type="button" value="dec"/>	Resetvalue = <input type="text"/> <input type="button" value="dec"/>
QB[Address] = <input type="text"/> <input type="button" value="hex"/>	Prm = <input type="text"/> <input type="button" value="hex"/>	<input type="button" value="Reboot node"/>
<input type="button" value="Set output value"/>	<input type="button" value="Set parameters"/>	

home

**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.

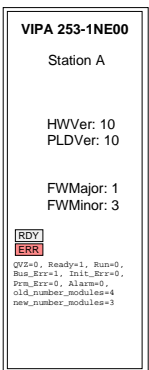
*PLDVer:* The PLD (**P**rogrammable **L**ogic **D**evice) is a programmable logic block for control of the communication between backplane bus and processor.

*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.

*RDY, ERR:* Status monitor of the LEDs RD and ER

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:

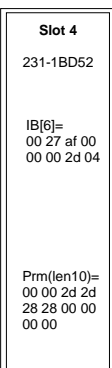
**Error monitor**



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 and parameter bytes.

*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
```

This means: The module at slot 4 occupies 8Byte of the input area, starting with Byte 6 and has a hexadecimal content.

The image is put out in little endian (Intel) format (Low-Byte, High-Byte).

*Parameter bytes:* 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.

### 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.

The following 3 control elements are available:

- Control outputs
- Parameterize module
- Reset the Ethernet coupler

Adress =	<input type="text" value=""/>	dec
QB[Adress] =	<input type="text" value=""/>	hex
Set output value		

#### 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 `Adress`.

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

Slot =	<input type="text" value=""/>	dec
Prm =	<input type="text" value=""/>	hex
Set parameters		

**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 zero 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.

Resetvalue =	<input type="text" value="1"/>	dec
Reboot node		

**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 home.

By presetting a *reset value*, you may additionally to the re-boot of the Ethernet coupler delete the configuration or module parameters.

Permissible *reset 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 the module configuration (module name) and re-boot the coupler
- Reset value= 3 Delete the module parameters and re-boot the coupler

## 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*

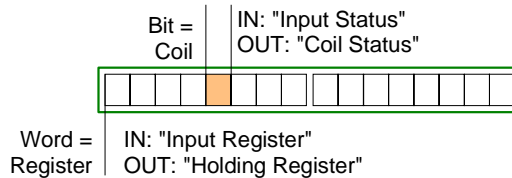
Byte	Name	Description
0	Transaction identifier (High-Byte)	Is sent back by the server (user-defined)
1	Transaction identifier (Low-Byte)	Is sent back by the server (user-defined)
2	Protocol identifier (High-Byte)	Always 0
3	Protocol identifier (Low-Byte)	Always 0
4	Length field (High-Byte)	Always 0 because 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".

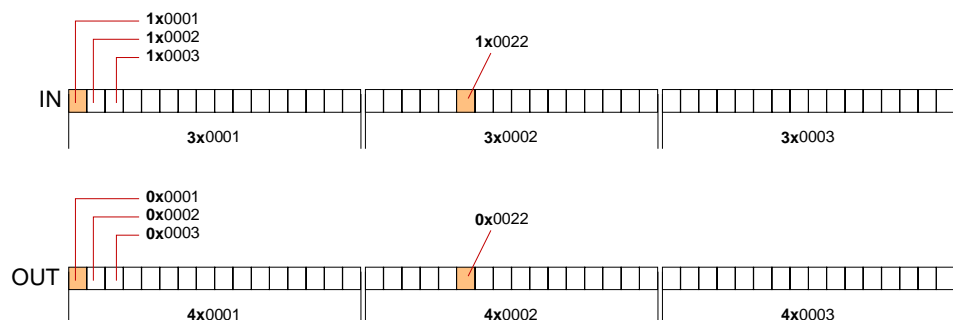
### Range definitions

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 output  
Access via function code 01h, 05h, 0Fh
- 1x: Bit area for input  
Access via function code 02h
- 3x: Word area for input  
Access via function code 04h
- 4x: Word area for output  
Access via function code 03h, 06h, 10h



A description of the function codes follows below.

**Overview**

The following Modbus function codes are implemented:

Code	Command	Description
01h	Read n Bits	Read n Bits of output area 0x
02h	Read n Bits	Read n Bits of input area 1x
03h	Read n Words	Read n Words of output area 4x
04h	Read n Words	Read n Words input area 3x
05h	Write one Bit	Write 1 Bit to output area 0x
06h	Write one Word	Write 1 Word to output area 4x
0Fh	Write n Bits	Write n Bits to area 0x
10h	Write n Words	Write n Words to area 4x

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 1 <sup>st</sup> Bit	Number of Bits
x   x   0   0   0   6				
6Byte	1Byte	1Byte	1Wordt	1Word

Respond telegram

ModbusTCP-Header	Slave address	Function code	Number of read Bytes	Data 1 <sup>st</sup> Byte	Data 2 <sup>nd</sup> 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 Bit	Number of Words
x   x   0   0   0   6				
6Byte	1Byte	1Byte	1Wort	1Wort

Respond telegram

ModbusTCP-Header	Slave address	Function code	Number of read Bytes	Data 1 <sup>st</sup> Word	Data 2 <sup>nd</sup> Word	...
x   x   0   0   0   6						
6Byte	1Byte	1Byte	1Byte	1Word	1Wordt max. 125Words	

max. 253Byte

**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 1 <sup>st</sup> Bit	Number of Bits	Number of Bytes	Data 1 <sup>st</sup> Byte	Data 2 <sup>nd</sup> Byte	...
x x 0 0 0 0								
	1Byte	1Byte	1Word	1Word	1Byte	1Byte	1Byte	1Byte
	max. 255Byte					max. 248Byte		

Respond telegram

ModbusTCP-Header	Slave address	Function code	Address 1 <sup>st</sup> Bit	Number of Bits
x x 0 0 0 6				
	1Byte	1Byte	1Wort	1Wort

**Write n Words 10h**

Via this function you may write n Words to the slave.

Command telegram

ModbusTCP-Header	Slave address	Function code	Address 1 <sup>st</sup> Word	Number of Words	Number of Bytes	Data 1 <sup>st</sup> word	Data 2 <sup>nd</sup> word	...
x x 0 0 0 0								
	1Byte	1Byte	1Word	1Word	1Byte	1Word	1Word	1Word
	max. 255Byte					max. 124Words		

Respond telegram

ModbusTCP-Header	Slave address	Function code	Address 1 <sup>st</sup> Word	Number of Words
x x 0 0 0 6				
	1Byte	1Byte	1Wort	1Wort

## 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

CPU area	MB	EB	AB	PB
ORG specification	02h	03h	04h	05h
Description	Only permitted: Read MB0 with length 4.  The total length of the in- and output areas is calculated and stored in	Source/destination data out/in Process image inputs (PAE).	Source/destination data out/in Process image outputs (PAA).	Source/destination data out/in peripheral module At source data input modules, at destination data output modules.
DBNR	MB0 ... MB3 in this format:	irrelevant	irrelevant	irrelevant
Start address Meaning  Permitted range:	MB0: Length In area MB1: 00 MB2: Length Out area MB3: 00	EB-No. from where on the data is fetched res. written.  0...255	AB-No. from where on the data is fetched res. written.  0...255	PB-No. from where on the data is fetched res. written.  0... 255
Number Meaning  Permitted range:		Length of the source/destination data block in Bytes.  1...256	Length of the source/destination data block in Bytes.  1...256	Length of the source/destination data block in Bytes.  1...256

**Structure  
PLC header**

READ and WRITE are created by the Ethernet coupler header for request res. acknowledgement telegrams. The headers have normally a length of 16Byte and have the following structure:

**at WRITE**

**Client (PLC, PC)**

Request telegram

System spec.	= "S"
	= "5"
Length.Header	=16d
Spec.OP-Code	=01
Length OP-Code	=03
<b>OP-Code</b>	<b>=03</b>
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
<b>OP-Code</b>	<b>=04</b>
Ackn. block	=0Fh
Length ackn. Block	=03
Error No.	=Nr.
Empty block	=FFh
Length empty block	=07
free	

**at READ**

Request telegram

System spec.	= "S"
	= "5"
Length.Header	=16d
Spec.OP-Code	=01
Length OP-Code	=03
<b>OP-Code</b>	<b>=05</b>
ORG-Block	=03
Length ORG-Block	=08
ORG specification	
DBNR	
Start address	H
	L
Length	H
	L
Empty block	=FFh
Length	=02

Acknowledgement telegram

System spec.	= "S"
	= "5"
Length.Header	=16d
Spec.OP-Code	=01
Length OP-Code	=03
<b>OP-Code</b>	<b>=06</b>
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

# 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.

PC IP: 172.16.192.50	Slave IP: 172.16.192.11		
<ul style="list-style-type: none"> <li>Socket System</li> </ul>		<p><b>to 1.</b> Start Microsoft Socket System</p>	<pre>WSAStartup (wVersionRequested, &amp;wsaData);</pre>
<ul style="list-style-type: none"> <li>TCP Socket</li> </ul>		<p><b>to 2.</b> Reserve Socket resources for TCP</p>	<pre>m_lsock = socket (AF_INET, SOCK_STREAM, 0);</pre>
<ul style="list-style-type: none"> <li>TCP Socket IP: 172.16.192.50 Port: 1200</li> </ul>		<p><b>to 3.</b> Link-up the socket to the local PC</p> <p>By calling <code>bind</code> 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)</p>	<pre>SocketAddr.sin_port = htons( 0 ); SocketAddr.sin_addr.S_un.S_addr = inet_addr( "0.0.0.0" ); bind(m_lsock, (LPSOCKADDR) &amp;SocketAddr, sizeof(SocketAddr));</pre>
<ul style="list-style-type: none"> <li>TCP Socket IP: 172.16.192.50 Port: 1200</li> </ul>	<ul style="list-style-type: none"> <li>ModbusTCP Server TCP Socket IP: 172.16.192.11 Port: 502</li> </ul>	<p><b>to 4.</b> Establish connection to external device</p>	<pre>SocketAddr.sin_port = htons (m_wPort); SocketAddr.sin_addr.S_un.S_addr = inet_addr(m_szIpAddress); connect(m_lsock, (LPSOCKADDR) &amp;SocketAddr, sizeof(SocketAddr));</pre>
<ul style="list-style-type: none"> <li>TCP Socket IP: 172.16.192.50 Port: 1200</li> </ul>	<ul style="list-style-type: none"> <li>ModbusTCP Server TCP Socket IP: 172.16.192.11 Port: 502</li> </ul>	<p><b>to 5.</b> For write res. read access you have to build up telegrams according to the protocol and store them in <code>sndBuf</code>. <code>sndBufLen</code> contains the number of Bytes to be sent.</p> <p><i>Read access</i></p> <p>Send <code>sndBuf</code> (Request)</p> <p>Receive telegram in <code>rcvBuf</code> (Response+data)</p> <p><i>Write access</i></p> <p>Send <code>sndBuf</code> (Request+data)</p> <p>Receive telegram in <code>rcvBuf</code> (Response)</p>	<pre>send(m_lsock, (char *)sndBuf, sndBufLen, 0); rcv(m_lsock, (char *)rcvBuf, sizeof(rcvBuf), 0); send(m_lsock, (char *)sndBuf, sndBufLen, 0); rcv(m_lsock, (char *)rcvBuf, sizeof(rcvBuf), 0);</pre>
<ul style="list-style-type: none"> <li>TCP Socket IP: 172.16.192.50 Port: 1200</li> </ul>	<ul style="list-style-type: none"> <li>ModbusTCP Server TCP Socket IP: 172.16.192.11 Port: 502</li> </ul>	<p><b>to 6.</b> Close socket again</p>	<pre>closesocket(m_lsock);</pre>
<ul style="list-style-type: none"> <li><del>TCP Socket IP: 172.16.192.50 Port: 1200</del></li> </ul>			

An easy programming sample can be downloaded under [ftp.vipa.de/support](ftp://ftp.vipa.de/support) Demo Client: Cx000059.

## Technical data

### IM 253NET

Electrical data	VIPA 253-1NE00
Voltage supply	DC 24V (20.4 ... 28.8V) via front from ext. power supply
Current consumption	120mA
Output current backplane bus	3.5A
Potential separation	≥ AC 500V
Status monitor	Via LEDs at the front side
Interfaces	RJ45 for Twisted-Pair-Ethernet
Ethernet Interface	
Connection	RJ45
Network topology	Star topology
Medium	Twisted Pair
Transfer rate	10/100MBit
Total length	max. 100m per segment
Online access	
Test/Diagnosis	http server integrated that graphically displays the configuration via website and supports parameterization and project engineering options for test purposes.
Project engineering	Via WinNCS with online coupler search and engineering
Combination with peripheral modules	
max. number of clients	8 per ModbusTCP res. Siemens S5 protocol
max. number of input byte	256
max. number of output byte	256
Dimensions and Weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	70g

## Chapter 8 PC 288 - CPU

### Overview

This chapter contains a description of operation of the PC 288 in the System 200V. After a summary and an overview of the system we will introduce you to the configuration of a PC-based system.

A sample communication session concludes the chapter.

The following description includes:

- System overview
- Principles
- Construction
- Configuration
- Sample configuration
- Technical data

### Content

Topic	Page
<b>Chapter 8 PC 288 - CPU</b> .....	<b>8-1</b>
System overview.....	8-2
Principles.....	8-3
Properties.....	8-4
PC 288 - CPU - Construction.....	8-4
Components.....	8-5
Storage media applications.....	8-9
Deployment in the System 200V.....	8-10
Using the BIOS setup.....	8-13
Register description.....	8-21
Technical data.....	8-23

## System overview

### PC 288



The PC 288 is a complete 486DX based PC. It is suitable for central and decentral control applications.

The external storage media is provided by CompactFlash cards or hard disks (IBM Microdrive) with a capacity of up to 1GByte.

### Order data PC 288

Type	Order number	Description
PC 288 - CPU	VIPA 288-2BL10	486 PC-LAN; 66MHz
CompactFlash	VIPA 950-1KS00	CompactFlash Type II
HDD	VIPA 950-1KH00	HDD 340MByte, 540MByte or 1GByte IBM Microdrive



## Principles

### General

The PC 288 provides you with a complete PC-AT in a compact package with the performance of a 486DX processor. MS-DOS 6.22 is pre-installed on the internal 8MB Flash-ROM.

The PC 288 has connectors for a mouse, a keyboard, a display monitor or a TFT-display as well as an RJ45 socket for network connection.

External memory is provided by a CompactFlash card (type II), which is inserted directly into the front of the unit. You may install CompactFlash cards resp. hard disks IBM Microdrive with a memory capacity up to 1GB.

### Applications

This PC conforms to System 200V requirements and it can be employed as master in conjunction with System 200V peripherals. You may use this combination to implement the structures that represent machines and plants in stand-alone operation or a Profibus network.

### Configuration

Control applications and simple graphic representations may be programmed in C and C++.

Since the source code for the vbus\_api application program has been placed in the public domain, it is a simple matter to create control applications.

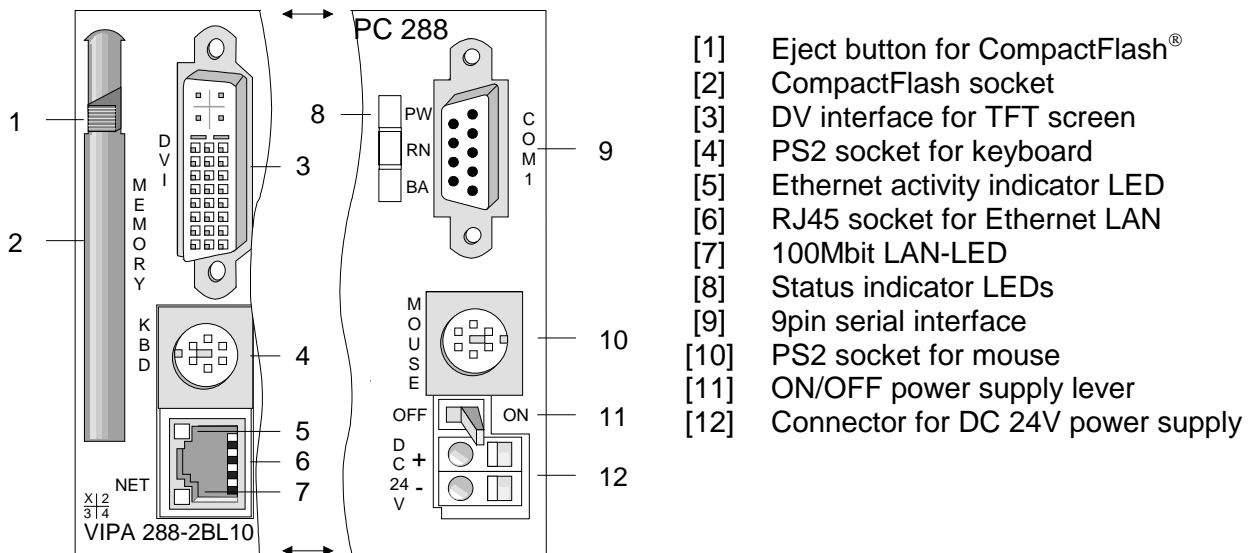
You can also use it to program the drivers for many different operating systems.

## Properties

- PC-AT compatible
- STPC INDUSTRIAL 66MHz
- 32MB RAM
- 8MB DiskOnChip®, bootable
- TYPE II slot for CompactFlash™ memory card
- Serial interface COM1
- Connector for an AT-type keyboard and compatible keyboards (foil keyboard, etc.) via a mini-DIN socket
- Mini-DIN socket for a mouse
- Powered by the 24V supply
- Integrated V-Bus controller for the control of System 200V modules
- Integrated Watchdog timer
- DV interface (Digital Visual Interface)  
Connector for a TFT - LCD via PANEL LINK®

## PC 288 - CPU - Construction

Front view  
PC 288



## Components

### LEDs

The PC 288 is equipped with 3 LEDs that are used as status indicators. These 3 LEDs are on when the power supply is turned on.

The following table shows the purpose and the respective color of these LEDs.

Name	Color	Description
PW	Yellow	Indicates that the PC has been turned on. The PC and the backplane bus (V-Bus) are receiving power.
RN	Green	Is on when the PC status is software RUN and V-Bus communication is active. This LED is not turned on when a V-Bus error occurs.
BA	Red	Is turned on when the output commands have been locked (BASP), i.e. the output modules are not enabled.

### ON/OFF lever

The ON/OFF switch controls the power supplied to the circuitry of the PC and to the backplane bus.

### Power supply

The PC is provided with an internal power unit. Power is connected by means of two terminals located on the front of the unit. The ON/OFF switch controls the power unit. In position OFF, power is removed from the backplane bus and the circuitry of the PC.

The power supply requires DC 24V (20...30V). The supply voltage is used to power the circuitry of the PC as well as the various modules that are connected to the PC via the backplane bus with max. 3.5A.



#### Note!

Verify that the polarity of the power connection is correct!

### Socket for CompactFlash

This socket can accommodate a type II CompactFlash® memory card. The PC includes this card into the system as an additional drive.

The CompactFlash® adapter provides compatibility with the "large" PCMCIA type II format to the memory. This means that you may exchange data with any PC via the PCMCIA slot.



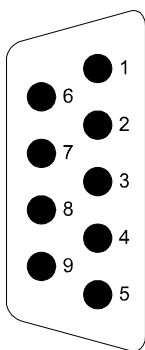
#### Note!

Never eject or insert the memory card when the PC is turned on!

## Sockets and plugs

### Serial interface COM 1

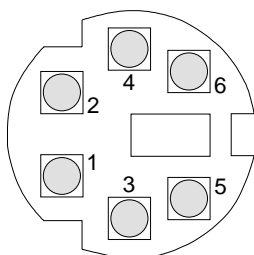
The connector of the serial interface is accessible as COM 1 and it has been designed to cater for a maximum distance of 15m at a communication rate of 38.4kBaoud. Data is communicated by means of data, handshaking and control lines.



Pin	RS232C	RS422/485
1	DCD-	CTS-
2	RXD	RXD-
3	TXD	TXD+
4	DTR-	TXD-
5	GND	GND
6	DSR-	RXD+
7	RTS-	RTS+
8	CTS-	RTS-
9	RI-	CTS+

### PS2 socket KBD/MOUSE

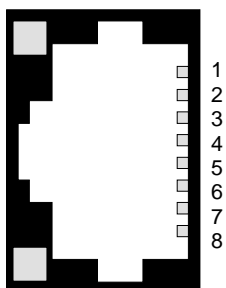
The pin assignment of the two PS2 sockets is identical. Connect your keyboard to the "KBD" socket and your mouse to the "MOUSE" socket.



Pin	Assignment
1	+ KBD-Data (I/O)
2	reserved
3	GND
4	+5V
5	+ KBD-Clock (I/O)
6	reserved

### RJ45 socket

The RJ45 socket provides a twisted-pair connection to your Ethernet LAN. The pin assignment and the purpose of the LEDs is as follows:



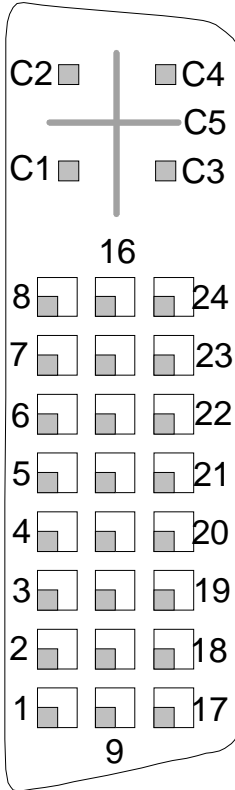
Pin	Signal
1	Transmit +
2	Transmit -
3	Receive +
4	-
5	-
6	Receive -
7	-
8	-

LED	
Ethernet activity (yellow)	On when a active Ethernet connection exists, blinks during data transfers
Rate (green)	Is turned on when 100MBit data transfer is active, else it is off.

**DVI socket**

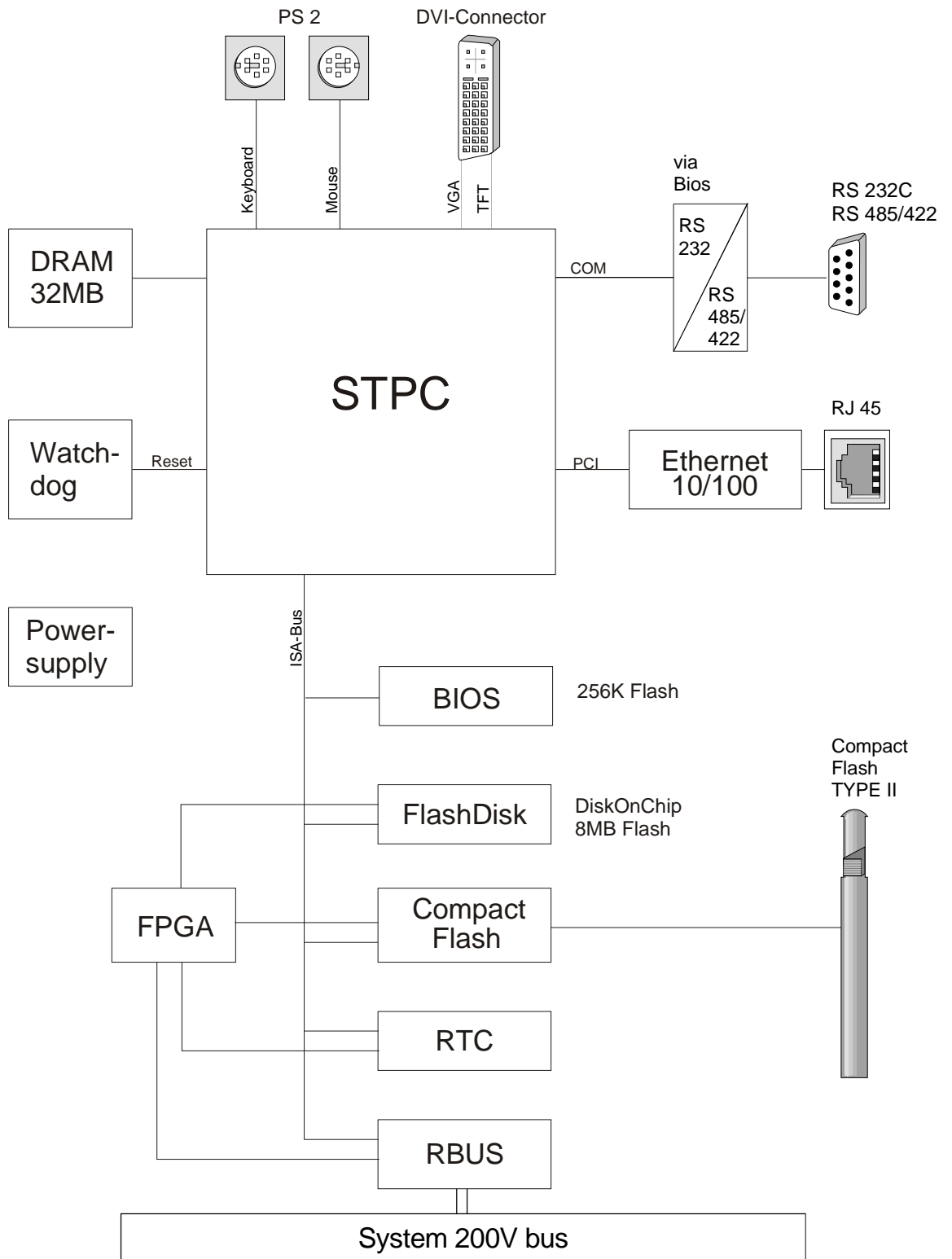
The DVI socket is the interface for analog and digital displays and monitors with a maximum resolution of 1280x1024 pixel.

The pin assignment of the socket is as follows:



Pin	Signal
C1	Analog Red
C2	Analog Green
C3	Analog Blue
C4	Analog Horizontal Sync
C5	Analog RGB Return
1	T.M.D.S Data2-
2	T.M.D.S Data2+
3	T.M.D.S Data2/4 Shield
4	T.M.D.S Data4-
5	T.M.D.S Data4-
6	DDC Clock
7	DDC Data
8	Analog Vertical Sync
9	T.M.D.S Data1-
10	T.M.D.S Data1+
11	T.M.D.S Data1/3 Shield
12	T.M.D.S Data3-
13	T.M.D.S Data3+
14	+5V Power
15	Ground (return for +5V, HSync and VSync)
16	Hot Plug Detect
17	T.M.D.S Data0-
18	T.M.D.S Data0+
19	T.M.D.S Data0/5 Shield
20	T.M.D.S Data5-
21	T.M.D.S Data5+
22	T.M.D.S Clock Shield
23	T.M.D.S Clock+
24	T.M.D.S Clock-

**Block diagram** The following block diagram shows the logical structure of the PC:



## Storage media applications

### Overview

The PC 288 has a Flash-ROM based internal drive providing 8MB of space and a type II CompactFlash® slot.

A CompactFlash® adapter provides the compatibility between the CompactFlash® card and the "large" PCMCIA type II format. This can be used to establish a communication link to PCs with a PCMCIA slot.

The physical drives are assigned via the BIOS-SETUP program.

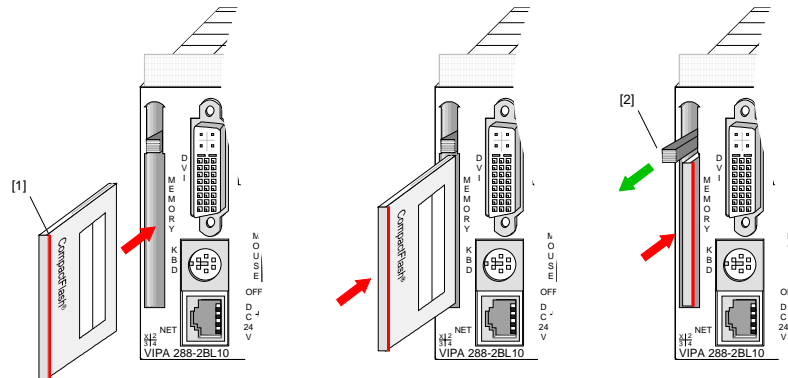
Different settings determine the boot behavior of the PC 288.

### Inserting/ejecting a CompactFlash®

Every CompactFlash® memory module has an extraction lip [1]. Make sure that this extraction lip faces to the right.

Insert the memory module into the PC 288 without force until it locks and the eject lever [2] becomes visible.

If you wish to eject the CompactFlash® adapter you press this eject lever.

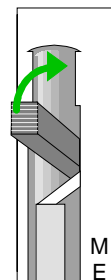


### Note!

Never eject or insert the CompactFlash® memory card when the PC is turned on!

The memory card has to be introduced in the setup!

### Protection against inadvertent ejection



You may fold the eject lever upwards into the enclosure to protect the unit against inadvertent ejection of the CompactFlash® card.

To eject the memory card you fold the eject lever out of the back to the original position or you can press the lever in the fold-away position using a pointed object, e.g. by means of a screwdriver.

## Deployment in the System 200V

### Overview

Applications using the PC 288 require C-language programming knowledge. VIPA supplies the PC together with the open source code of the software interface.

Since this code contains a description of all the functions together with examples of the application of the different functions, we do not include further details of these and the V-Bus organization in this manual.

The **vbus\_api.c** contains all the functions.

The file **vbus\_api.h** contains the respective descriptions.

The file **softsps.c** contains an application example for these functions.

### Automatic address allocation

Certain addresses in the PC must be associated with specific peripheral modules so that the installed modules can be accessed.

The PC 288 has a peripheral area (address 0...255 )and a process image of the inputs and outputs (0...127 per address) that is similar to the memory organization of a CPU.

During the start-up phase the PC automatically assigns peripheral addresses to the digital input/output modules starting from 0.

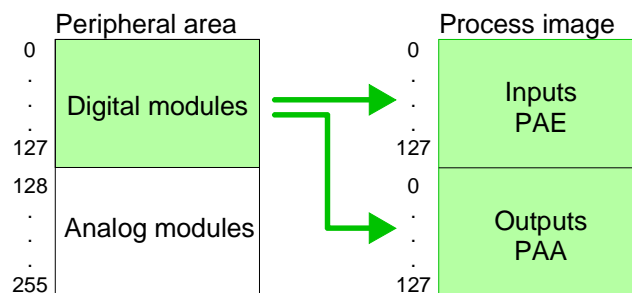
The automatic addressing for analog modules assigns even addresses starting from 128.

### Signal status in the process image

The statuses of the signals at the lower addresses (0...127) are also transferred into a special memory area, the *process image*.

The process image is divided into two parts:

- Process image of inputs (PAE)
- Process image of outputs (PAA)



In contrast to the CPU the process image of the PC 288 is not updated automatically. This facility is provided by the functions *vbus\_read\_pa* and *vbus\_write\_pa*.



**Read and write access**

The modules are accessible by means of read or write accesses to the peripheral bytes or the process image.



**Note!**

Please note that the read and write access to one address can be directed to different modules,

e.g. `vbus_read_pword (128,&w)` reads from the AI at plug-in location 3

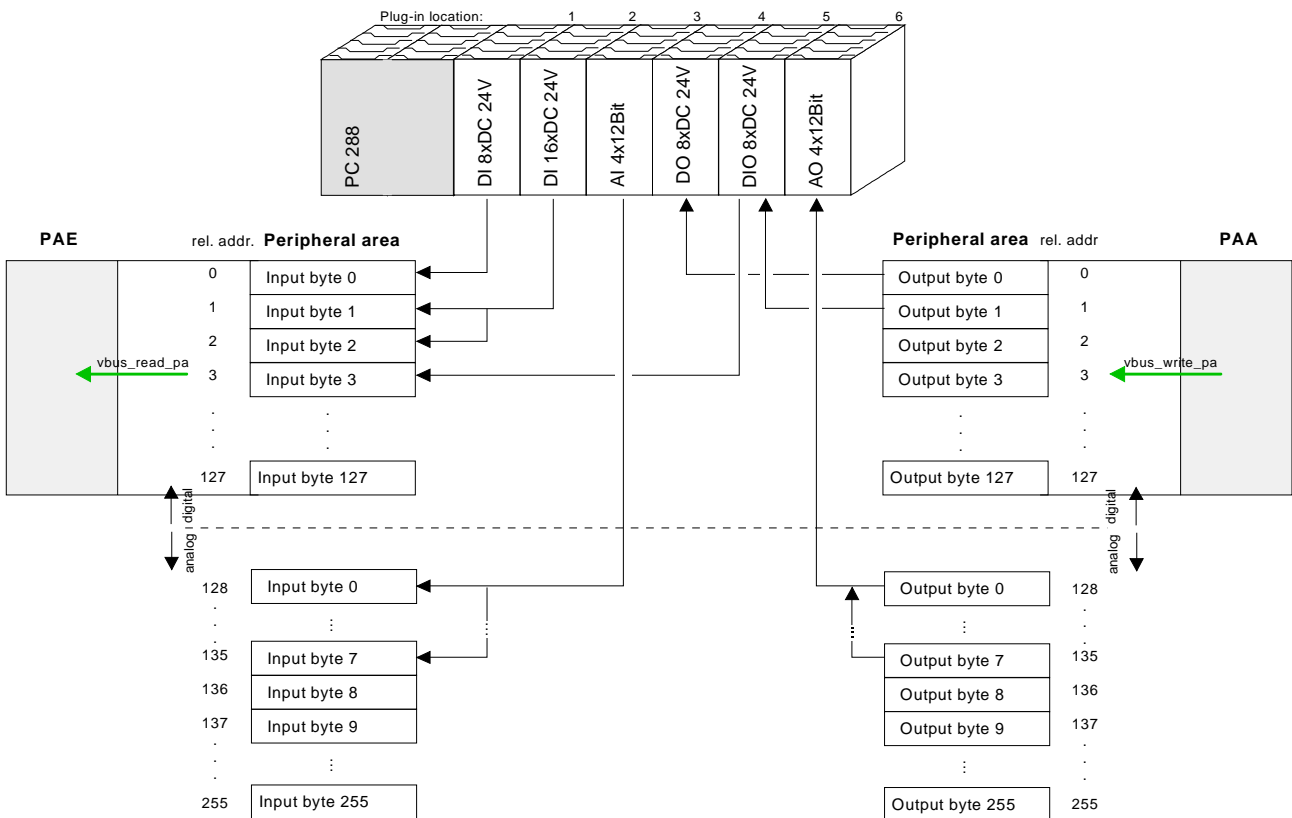
`vbus_write_pword (128,w)` writes to the AO at plug-in location 6

Separate address ranges are assigned to digital and to analog modules during automatic address allocation.

Digital modules: 0...127

Analog modules: 128...255

The following figure explains the process of the automatic address allocation:



**Modifying the allocation by means of `set_address_table`**

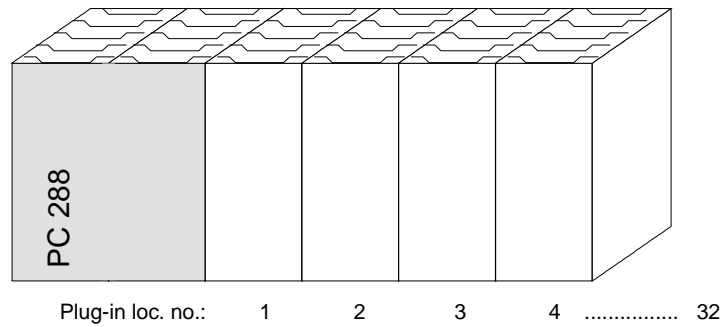
You may use the `set_address_table` function to replace automatically allocated addresses in your program. For this purpose you have to create a custom allocation table and supply this to the `set_address_table` function. In this manner it is possible to locate also analog modules in the process image and digital modules at addresses above 127!

The new allocation is enabled by means of the `vbus_businit` function.

Please refer to the function descriptions in `vbus_api.h`.

**Module configuration by means of *vbus\_set\_param***

System 200V modules, e.g. analog modules, can be configured by means of up to 16Byte of configuration parameters provided by the PC. This operation is provided by the *vbus\_set\_param* function. *vbus\_set\_param* accesses the respective module directly via the plug-in location number (1...32) and saves the parameters in a buffer.



The function *vbus\_businit* transfers and activates the new set of parameters. For more information please refer to the function description contained in ***vbus\_api.h***.



**Note!**

Please remember that the delay time of the peripheral modules in the System 200V is app. 2ms unless specified otherwise!

## Using the BIOS setup

### General

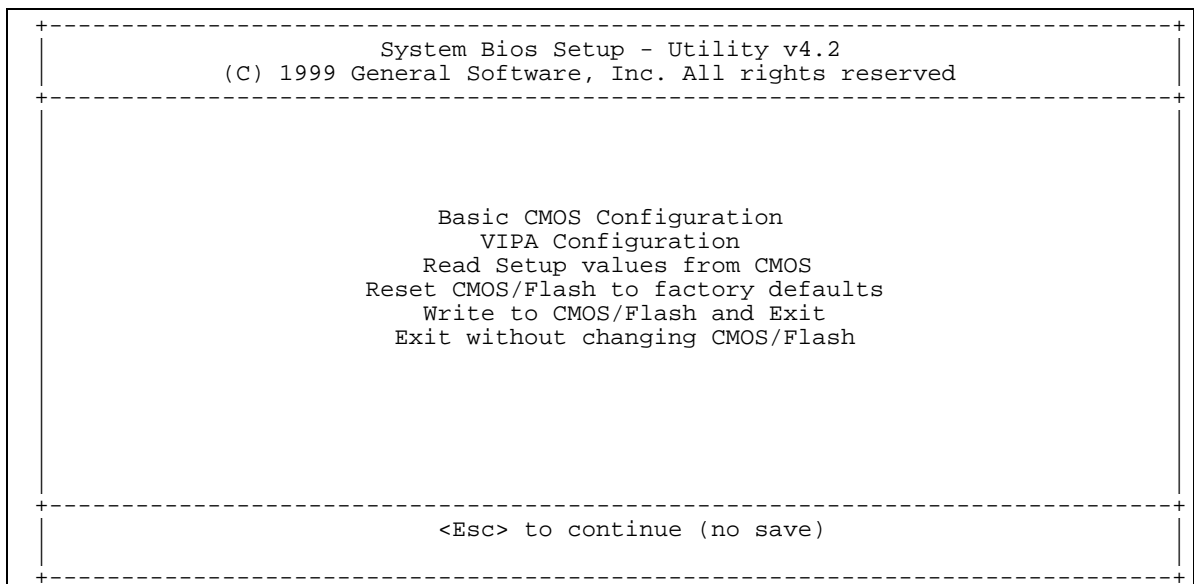
At the start-up of the system, the BIOS version is displayed on the connected monitor. Then the BIOS executes a test of the system components and the memory. At the end of the test the system attempts to boot. Between the start-up and the boot procedure you may access the BIOS setup routine by operating the **[Entf]/[Del]** key.

An appropriate message is displayed on the screen.

Via the setup menu you may configure the hardware of your PC.

### BIOS menu

After hitting the **[Entf]/[Del]** key, the following menu appears:



With the help of the standard cursor keys you may select the single menu options. You activate the selected sub-menu with **[Enter]**.

Via **[ESC]** you leave the setup without saving your entries.

**Control keys in BIOS**

Every dialog box that is accessible via the main menu is controlled by means of the following keys:

**[ESC] key**

With the [ESC] key the dialog window is closed and you return to the main menu. The altered parameters are stored but not written to the CMOS.

**Cursor keys**

With the Cursor keys you choose the parameter you want to modify.

**[PU]/[PD]**

With the keys [PgUp] and [PgDn] or [Bild↑] and [Bild↓] or at the numeric block [+] and [-] you may alter the value of a parameter.

**Note!**

Please regard, that at the setup level there has not been loaded a driver for the country specific keyboard yet. Modified setup values are only valid and written into CMOS by confirming your changes explicitly with "Y". To obtain Y, you have to push the [Z] key.

**Basic CMOS Configuration**

This sub-menu provides access to the main settings for your system. The menu is divided into a number of logical units. You can navigate through these by means of the cursor keys.

System Bios Setup - Basic CMOS Configuration (C) 1999 General Software, Inc. All rights reserved			
<b>DRIVE ASSIGNMENT ORDER:</b>	Date:>Jan 01, 2000	Typematic Delay :	250 ms
Drive A: (None)	Time: 10 : 03 : 25	Typematic Rate :	30 cps
Drive B: (None)	NumLock: Disabled	Seek at Boot :	None
Drive C: CompactFlash		Show "Hit Del" :	Enabled
Drive D: (None)	<b>BOOT ORDER:</b>	Config Box :	Enabled
Drive E: (None)	Boot 1st: Drive C:	F1 Error Wait :	Enabled
Drive F: (None)	Boot 2nd: (None)	Parity Checking :	(Unused)
Drive G: (None)	Boot 3rd: (None)	Memory Test Tick :	Enabled
Drive H: (None)	Boot 4th: (None)	Test Above 1 MB :	Disabled
Drive I: (None)	Boot 5th: (None)	Long Memory Test :	(Unused)
Drive J: (None)	Boot 6th: (None)	Hexadecimal Case :	Upper
Drive K: (None)			
Boot Method: Boot Sector	<b>IDE DRIVE GEOMETRY:</b>	Sect Hds Cyls	Memory
	Ide 0: 2 = AUTOCONFIG, PHYSICAL		Base:
<b>FLOPPY DRIVE TYPES:</b>	Ide 1: Not installed		640KB
Floppy 0: Not installed	Ide 2: Not installed		Ext:
Floppy 1: Not installed	Ide 3: Not installed		30MB
	U/D/L/R/<CR>/<Tab> to select or <PgUp>/<PgDn>/+/- to modify		

**Drive Assignment Order and IDE Drive Geometry**

This section is used to assign logical drive names to the physical drives. The VIPA BIOS only provides support for drive "C". The following settings are valid for "C":

- "None": if you did **not** insert a CompactFlash® card
- "CompactFlash": if you have installed a CompactFlash® card.



**Note!**

All the other drives must be assigned to type "None". Any other setting can cause malfunction of the PC.

Please also note that the settings for "Ide 0" in the section *IDE Drive Geometry* must be changed at the same time as the parameters for drive "C". The following settings are valid:

CompactFlash®	DRIVE ASSIGNMENT ORDER \ Drive C:	IDE DRIVE GEOMETRY \ Ide0:
Installed	CompactFlash®	2 = AUTOCONFIG, PHYSICAL
Not installed	None	Not installed

The integrated DiskOnChip® drive (DOC) is configured by means of the "VIPA Configuration" menu.

**Floppy drive types** Here you would normally define the settings for the floppy disk drive. Since the PC 288 does not support floppy disk drives you have to select the type "Not installed"! Otherwise you will encounter long-term delays between the system test and the start-up of the system.

**Date, Time, NumLock** Here you enter the current date and time.  
The "NumLock" parameter defines the status of the [NumLock] key after the system has booted.



**Note!**

If the real-time clock has stopped it has to be assumed that the backup battery for the CMOS memory is discharged or defective.

Please contact the VIPA Hotline if this battery has still not accepted a charge after one day.

Your CMOS settings are safe even if the battery is empty. You only need to adjust the clock and the date.

**Boot order** This parameter is preset to "Drive C" and it defines the boot sequence.

**IDE Drive Geometry** This section defines the geometry settings of the IDE drives.  
If you are using a CompactFlash<sup>®</sup> card you have to enter "autoconfig, physical" for this parameter. If you did not install a CompactFlash<sup>®</sup> card the parameter must be set to "None".

**Typematic Rate/Delay** This parameter defines the keyboard interface and specifies the repetition rate for the characters. You should not change this setting.

**Seek at Boot** This option specifies which drives should receive a "SEEK" command before booting the system. The default is "NONE" to ensure that the boot procedure is as quick as possible.

**Show "Hit Del"** If this option is active the system will display a message to press [Del] to access the setup menu while the system is booting.

**Config Box** This parameter specifies whether the configuration settings should be displayed on screen when the system boots.

---

<b>F1 Error Wait</b>	If you activate this menu item, the boot process is stopped when an error is detected. You then may decide what action to take. If you press [F1] the system will continue booting. Press [Del] to gain access to the setup menu.
<b>Parity Checking</b>	This menu item is not used.
<b>Memory Test Tick</b>	Turn this option on to sound a test click via the PC's speaker when memory is being tested.
<b>Test Above 1MB</b>	This parameter determines whether the memory above the 1MB limit will be tested by the memory test or not.
<b>Long Memory Test</b>	This menu item is not used.
<b>Hexadecimal Case</b>	This menu item determines the display format that the BIOS uses for hexadecimal numbers. You can either choose "UPPER" (capitals) or "LOWER" (lower case).
<b>Memory Base / Ext.</b>	This menu item displays the memory configuration below 1MB (Base Memory) and above 1MB (Extended memory). These parameters are purely for information purposes and they cannot be changed.

**VIPA Configuration**

This sub-menu is used to define the board-specific settings. You may navigate through the menu by means of the cursor keys.

System BIOS Setup - VIPA Configuration (C) 1999 General Software, Inc. All rights reserved			
VGA Frame Buffer Size	: 1.0 Mb	VGA Palette Snoop	: Enabled
Graphic Clock Speed	: 85 MHz	Watchdog	: Enabled
Drive C	: CF	Com1 Mode	: RS232
L1 Cache	: Enabled		
Bios Version: 1.0.0 Seriennummer: 040002 Ausgabestand: 1 FPGA Version: 10			
U/D/L/R/<CR>/<Tab> to select or <PgUp>/<PgDn>/+/- to modify			

**VGA Frame Buffer Size**

This parameter determines the amount of memory that is used as graphic memory for the internal graphic controller.

The respective memory area is no longer available as main memory for the processor.

**Graphic Clock Speed**

This option determines the clock speed used for the internal graphic controller. This option should always be set to "85MHz".

**Drive C**

This parameters determines the drive from which the operating system is booted:

"CF" CompactFlash® (C:) followed by the internal drive (D:)

"DOC" DiskOnChip® - internal drive (C:) followed by CompactFlash® (D:)

In every case the other drive will be accessible as drive D:.

**L1 Cache**

This option enables and disables the L1 cache of the processor.

**VGA Palette Snoop**

This menu item determines whether access to the VGA palette can only occur within the STPC (disabled) or whether it should also be routed to the external PCI bus (enabled).



**Watchdog** This option is used to enable or disable a Watchdog that issues an automatic reset after a certain time has elapsed (30s).  
If you deactivate the Watchdog in your application program you can ensure that your system has booted without errors. Otherwise the PC is re-booted when the watchdog has expired.  
You may define the watchdog time and the watchdog properties in the system register.

**Com1 Mode** This parameter determines the physical properties of serial interface COM 1. You may select from:  
RS232 RS232C interface  
RS422 RS422/485 interface

**Version data** The left hand section of the "VIPA Configuration" menu displays hardware specific parameters:  
BIOS-Version: The version level of the BIOS  
Serial No.: Every PC is provided with a unique serial number. This serial no. is identical to the serial no. located on the enclosure.  
Revision level: Identical to the revision level located on the enclosure.  
FPGA Version: The version of the FPGA that controls the V-Bus access.



**Note!**

You should include this information when you request information from the service department of VIPA GmbH to allow us to help you more effectively.

**Note!**

The following menu items display a query that is to be answered with "Yes" or "No".

Please note that the keyboard uses the US layout for the BIOS setup, i.e. that you have to press the [Z] key on German keyboards to obtain the letter "Y".

---

**Read Setup values from CMOS**

This option loads the most recent settings that were saved to CMOS memory.

---

**Reset CMOS/Flash to factory defaults**

This option loads the factory default values into CMOS memory.

If you wish to write these values into the CMOS memory you have to use the menu item "Write to CMOS and Exit" to quit from this function.

---

**Write to CMOS/Flash and Exit**

This menu item saves the modified settings to CMOS-RAM.

When the settings have been saved the system is re-started automatically which will reload the modified configuration settings.

**Note!**

This process can require a few seconds since the data is also saved in the EEPROM.

During this time you may **NOT** turn the system off or issue a manual reset!

---

**Exit without changing CMOS/Flash**

When you select this menu item, you quit from the setup menu without saving and activating any settings to CMOS memory or to the EEPROM.

## Register description

**Address range**      The following addresses are occupied by VIPA:

<b>270h - 277h</b>	Watchdog
<b>280h - 28Fh</b>	
280h - 284h	reserved
285h	EEPROM Port
286h - 28Dh	reserved
28Eh	Version
28Fh	Device ID = 84h
<b>290h - 297h</b>	
290h	WD-Timer
291h	RS232/RS422
292h	C165 control register
293h	Enable register
294h - 297h	reserved

**Watchdog**      (I/O address range **270h-277h**)

The Watchdog is turned off when the system has started or has been reset and it may be enabled under software control.

The Watchdog register is controlled via I/O address 270h and the following parameters:

Watchdog on	Enter 40h into address 270h
Watchdog off	Enter 50h into address 270h
Watchdog trigger	Enter 60h into address 270h and then 70h into address 270h

The Watchdog has to be triggered when the power has been turned on. The triggering time is programmable:

**Trigger time**      Parameter for triggering time I/O register **290h** R/W (enabled via **293h**)

time	= content of register <b>290h</b> x 117ms
content	= "0" Watchdog turned off

**Enable** Write access only I/O-Port **293h** for WD-Timer.  
 The following sequence enables I/O-register **290h** and **292h** for write access:

```
WR 293h          03h
WR 293h          06h
WR 293h          03h
WR 293h          01h
```

The following sequence inhibits I/O-register **290h** and **292h** for write access:

```
WR 293h          03h
WR 293h          06h
WR 293h          03h
WR 293h          00h
```



**Note!**  
 After a reset write access to **290h** and **292h** is inhibited.

**Serial number** The serial number is available to the user via registers **271h** and **272h**.

**EEPROM** VIPA will supply you with then relevant details on I/O-Port **285h**.

**RS232/RS422** Changing the operating mode of COM1 RS232/RS422 I/O-Port **291h** R/W

	"0"	"1"
Bit 0	RS232	RS422/485
Bit 1	RX enable	TX disable
Bit 2	TX disable	RX enable
Bit 3...7	reserved	

**C165 control register** The C165 control registers I/O-Port **292h** R/W (enabled via **293h**) are reserved for the download.

## Technical data

### PC-CPU PC 288

Electrical data	VIPA 288-2BL10
Power supply	DC 24V (20.4 ... 28.8) via front from ext. power supply
Current consumption	max. 1.5A
Output current backplane bus	max. 3.5A
Status indicators (LED)	by means of LEDs located on the front
Connectors / interfaces	2pin                    power connector Mini-DIN             AT-keyboard Mini-DIN             mouse 9pin                    COM1: serial interface PCMCIA               CompactFlash card Type II DVI                     interface for display/TFT RJ45                    twisted pair interface for Ethernet ON/Off-switch for power supply
Battery backup for clock and CMOS	Lithium battery, 30 day backup
Combination with peripheral modules	
max. no. of modules	32
max. digital I/Os	32
max. analog I/Os	16
Dimensions and weight	
Dimensions (WxHxD) in mm	50.8x76x76
Weight	170 g



## Chapter 9 Communication processor CP 240

### Overview

This chapter contains a description of the construction and the interfaces of the communication processor CP 240. VIPA distributes the communication processor CP 240 with different communication protocols that are explained in the following. Additionally you will find a description of the standard handler blocks that are supplied with the processor.

The following description includes:

- a system overview
- the protocols ASCII, STX/ETX, 3964(R), RK512 and Modbus
- project engineering and deployment of the communication processor
- Standard handler blocks
- Technical data

### Content

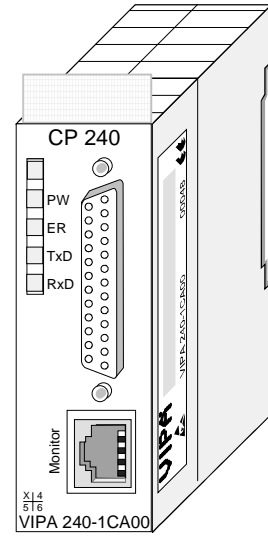
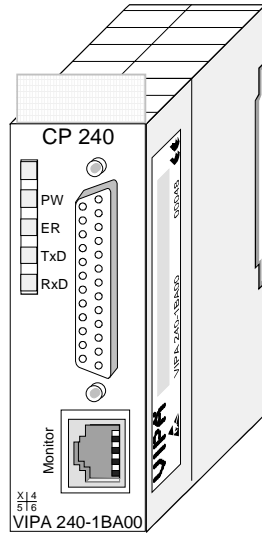
Topic	Page
<b>Chapter 9 Communication processor CP 240</b> .....	<b>9-1</b>
System overview.....	9-2
Principles ASCII, STX/ETX, 3964(R), RK512 .....	9-3
Principles Modbus.....	9-10
CP 240 with 20mA/RS232C interface - Construction .....	9-11
CP 240 with RS422/RS485 interface - Construction .....	9-16
Parameterization.....	9-22
Access to the CP 240 interface under ASCII, STX/ETX, 3964(R).....	9-30
Deployment under Modbus .....	9-32
Modbus function codes .....	9-36
Example for the deployment under Modbus.....	9-39
Communication by means of standard handler blocks .....	9-45
Standard handler blocks for the CPU 24x.....	9-46
Standard handler blocks for the CPU 21x.....	9-61
Technical data .....	9-78

### Order data

Type	Order no.	Description
CP 240	VIPA 240-1BA00	CP 240 with 20mA/RS232C interface Protocols: ASCII, STX/ETX, 3964(R), RK512
CP 240	VIPA 240-1CA00	CP 240 with RS422/485 interface Protocols: ASCII, STX/ETX, 3964(R), RK512
CP 240 Modbus	VIPA 240-1BA10	CP with 20mA/RS232C interface Protocol: Modbus
CP 240 Modbus	VIPA 240-1CA10	CP with RS422/485 interface Protocol: Modbus
Diagnostic tool	UPI-FOX2	Diagnostic cable incl. software

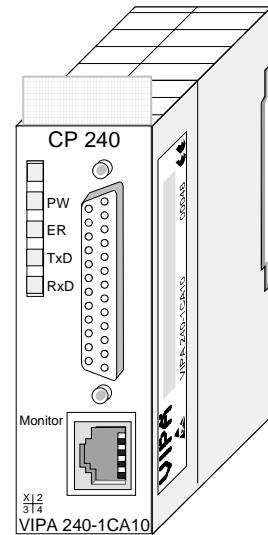
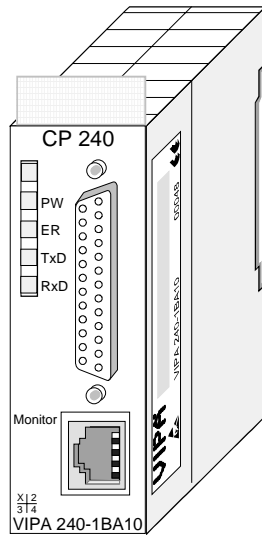
## System overview

**CP 240 with the protocols ASCII, STX/ETX, 3964(R) RK512**



Type	Order no.	Description
CP 240	VIPA 240-1BA00	CP 240 with 20mA/RS232C interface Protocols: ASCII, STX/ETX, 3964(R), RK512
CP 240	VIPA 240-1CA00	CP 240 with RS422/485 interface Protocols: ASCII, STX/ETX, 3964(R), RK512

**CP 240 with Modbus protocol**



Type	Order no.	Description
CP 240 Modbus	VIPA 240-1BA10	CP with 20mA/RS232C interface Protocol: Modbus
CP 240 Modbus	VIPA 240-1CA10	CP with RS422/485 interface Protocol: Modbus



## Principles ASCII, STX/ETX, 3964(R), RK512

- General** The CP 240 modules provide serial interfacing facilities between the processes of different source and destination systems.  
The CP 240 modules have an integrated serial interface that may be configured by means of hardware to operate either as RS232C or 20mA resp. RS422 or RS485 interface.  
The CP 240 modules obtain the required operating power via the back-plane bus.
- Protocols** The communication processor supports the ASCII, STX/ETX, 3964(R) and RK512 or Modbus (VIPA 240-1BA10 resp. VIPA 240-1CA10) protocols and procedures.
- Diagnostic facilities** The front of the unit provides access to a diagnostic interface for troubleshooting and service purposes. This interface carries the signals RxD and TxD at RS232 levels (TTL levels).  
VIPA may supply you with a diagnostic adapter (Order no.: UPI-FOX2) that contains the serial interface between the diagnostic interface and a PC. The software allows analysis of the signal stream and operational tests of your interface.
- Parameterization** The CP can be configured by means of 16Byte of configuration data that contain the parameters required by the selected protocol.
- Communication** Handler blocks that are supplied with the modules of the CPU-families 21x and 24x by VIPA control the serial communication.  
Data transmitted between the CP 240 and a communication partner is transferred via the serial interface in a 9 or a 12Bit character frame. Three different formats are available for every character frame. These formats differ in the number of data bits, with or without parity bit and the number of stop bits.

Character frame	Start bit	Data bits	Parity bit	Stop bit
9Bit	1	7	-	1
10Bit	1	7	-	2
10Bit	1	7	1	1
10Bit	1	8	-	1
11Bit	1	7	1	2
11Bit	1	8	1	1
11Bit	1	8	-	2
12Bit	1	8	1	2

**Protocols and procedures**

The data transfer between any two communication partners is controlled by means of protocols or procedures as for instance:

- ASCII communication
- STX/ETX
- 3964(R)
- RK512

**ASCII**

ASCII data communication is one of the simple forms of data exchange that can be compared to a multicast/broadcast function.

Individual messages are separated by means of 2 windows in time. The sending station has to transmit data messages within the character delay time (ZVZ) or receive window that was defined in the receiving station.

The receiving station must acknowledge the receipt of the message within the "time delay after command" (ZNA) or command window that was defined in the sending station.

These time stamps can be used to establish a simple serial communication link between PLC and PLC.

A send command is only flagged as "command completed without errors" (AFOF) when the data has been transferred and the ZNA has expired.

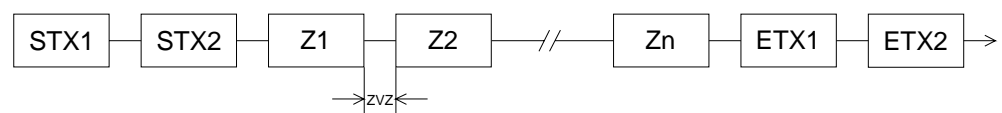
**STX/ETX**

STX/ETX is a simple protocol employing headers and trailers. The STX/ETX procedure is suitable for the transfer of ASCII characters (20h...7Fh). It does not use block checks (BCC). Any data transferred from the periphery must be preceded by an STX (Start of Text) followed by the data characters. An ETX (End of Text) must be inserted as the terminating character.

The effective data which includes all the characters between STX and ETX are transferred to the CPU when the ETX has been received.

When data is sent from the CPU to a peripheral device, any user data is handed to the CP 240 where it is enclosed with an STX start character and an ETX termination character and transferred to the communication partner.

Message structure:



You may define up to 2 start and end characters. It is also possible to specify a ZNA for the sending station.

**3964(R)**

The 3964(R) procedure controls the data transfer of a point-to-point link between the CP 240 and a communication partner. The procedure adds control characters to the message data during data transfer. These control characters may be used by the communication partner to verify the complete and error free receipt.

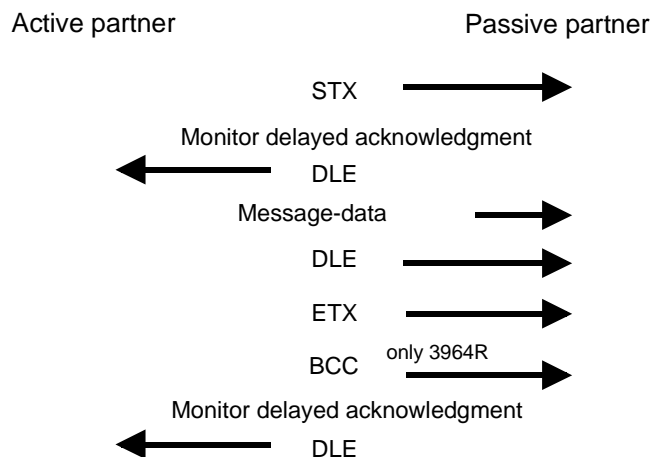
The procedure employs the following control characters:

- STX        Start of Text
- DLE        Data Link Escape
- ETX        End of Text
- BCC        Block Check Character (only for 3964R)
- NAK        Negative Acknowledge

**Note!**

When a DLE is transferred as part of the information it is repeated to distinguish between data characters and DLE control characters that are used to establish and to terminate the connection (DLE duplication). The DLE duplication is reversed in the receiving station.

The 3964(R) procedure requires that a lower priority is assigned to the communication partner. When communication partners issue simultaneous send commands the station with the lower priority will delay its send command.

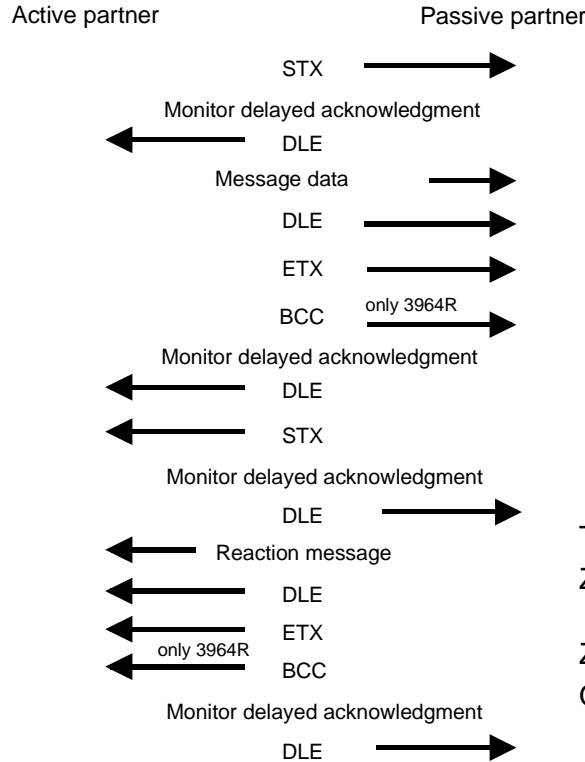
**Procedure**

You may transfer a maximum of 250Byte per message.

**3964(R)  
with RK512**

The RK512 is an extended form of the 3964(R) procedure. The difference is that a message header is sent ahead of the message data. The header contains data about the size, type and length of the message data.

**Procedure**



Timeout times:  
 ZNA (time delay after command)  
 ZVZ (character delay time)  
 QVZ (acknowledgment delay time)

**Timeout times**

The QVZ is monitored between STX and DLE and between BCC and DLE. ZVZ is monitored for the entire period of receiving the message.

When the QVZ expires after an STX, the STX is repeated. This process is repeated 5 times after which the attempt to establish a connection is terminated by the transmission of a NAK. The same sequence is completed when a NAK or any other character follows an STX.

When the QVZ expires after a message (following the BCC-byte) or when a character other than DLE is received the attempt to establish the connection and the message are repeated. This process is also repeated 5 times after which a NAK is transmitted and the attempt is terminated.

**Passive operation**

When the procedure driver is expecting a connection request and it receives a character that is not equal to STX it will transmit a NAK. The driver does not respond with an answer to the reception of a NAK.

When ZVZ expires during the reception, the driver will send a NAK and wait for another connection request.

The driver also sends a NAK when it receives an STX while it is not ready.

**Block check character (BCC-Byte)**

The 3964R procedure appends a Block check character to safeguard the transmitted data. The BCC-Byte is calculated by means of an XOR function over the entire data of the message, including the DLE/ETX.

When a BCC-Byte is received that differs from the calculated BCC, a NAK is transmitted instead of the DLE.

**Initialization conflict**

If two stations should simultaneously attempt to issue a connection request within the QVZ then the station with the lower priority will transmit the DLE and change to receive mode.

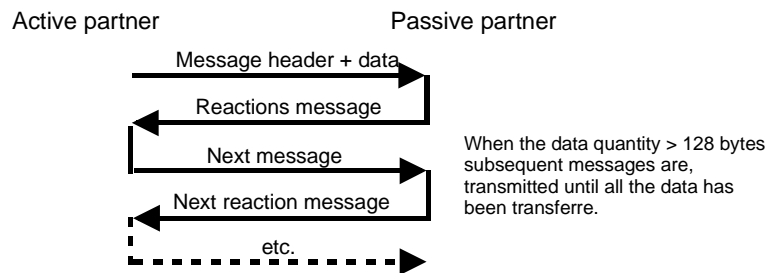
**Data Link Escape (DLE-character)**

The driver duplicates any DLE-character that is contained in a message, i.e. the sequence DLE/DLE is sent. During the reception, the duplicated DLEs are saved as a single DLE in the buffer. The message always terminates with the sequence DLE/ETX/BCC (only for 3964R).

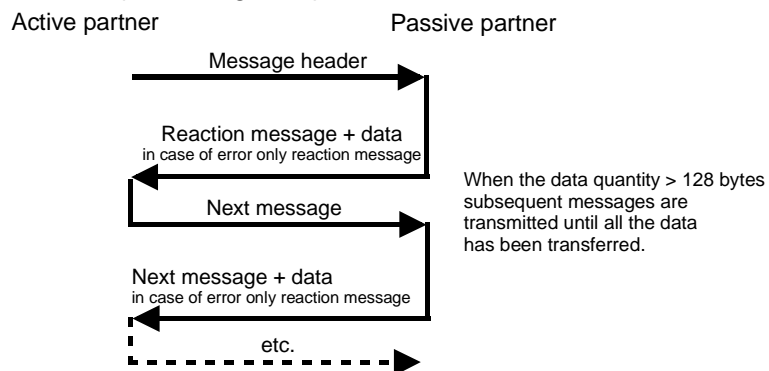
The control codes :           02h = STX  
                                   03h = ETX  
                                   10h = DLE  
                                   15h = NAK

**Logical message sequence**

*SEND (transmission of data)*



*FETCH (retrieving data)*

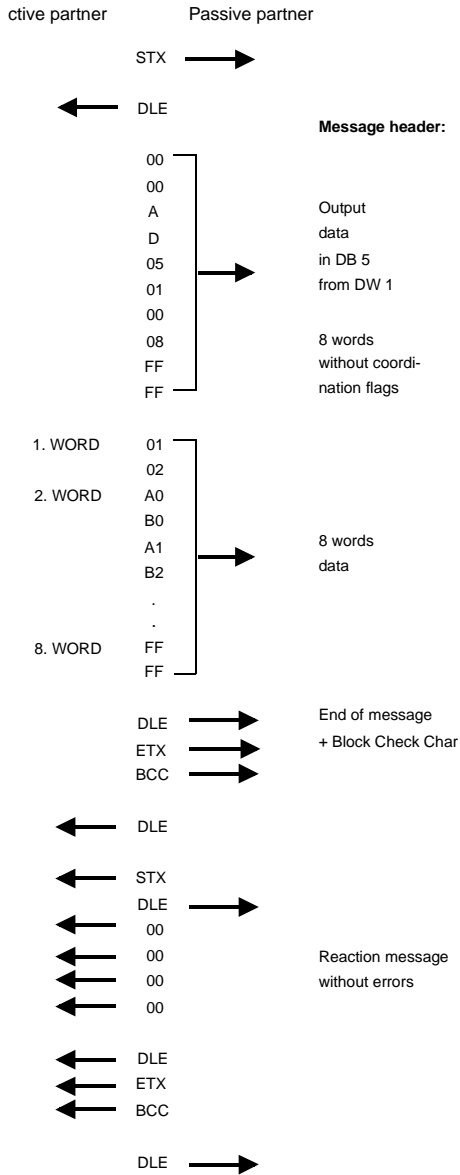


In both cases the procedure will time out after a maximum period of 5s during which a reaction must be received, else the reception is terminated.

**Message contents** Every message has a header. Depending on the history of the message traffic, this header will contain all the required information.

**Structure of the output message**

*Sample output message*



Normal message

Byte	Hex	Description
0	00	Message flag
1	00	flag
2	A	Output command
3	X	type of data
4	xx	Parameter 1
5	xx	Destination
6	yy	Parameter 2
7	yy	Quantity
8	zz	Parameter 3
9	zz	Coordination flag
10 - N	aa bb xy	Data

with N = 10 ... 127

Reaction message

Byte	Hex	Description
0	00	Flag for reaction message
1	00	reaction message
2	00	message
3	xx	Error code

When the data exceeds 128Byte, additional messages will be sent.

*Structure of additional messages*

Next message

Byte	Hex	Description
0	FF	Flag for next message
1	00	next message
2	A	Output command
3	X	Data type
4 - N	aa bb xy	Data

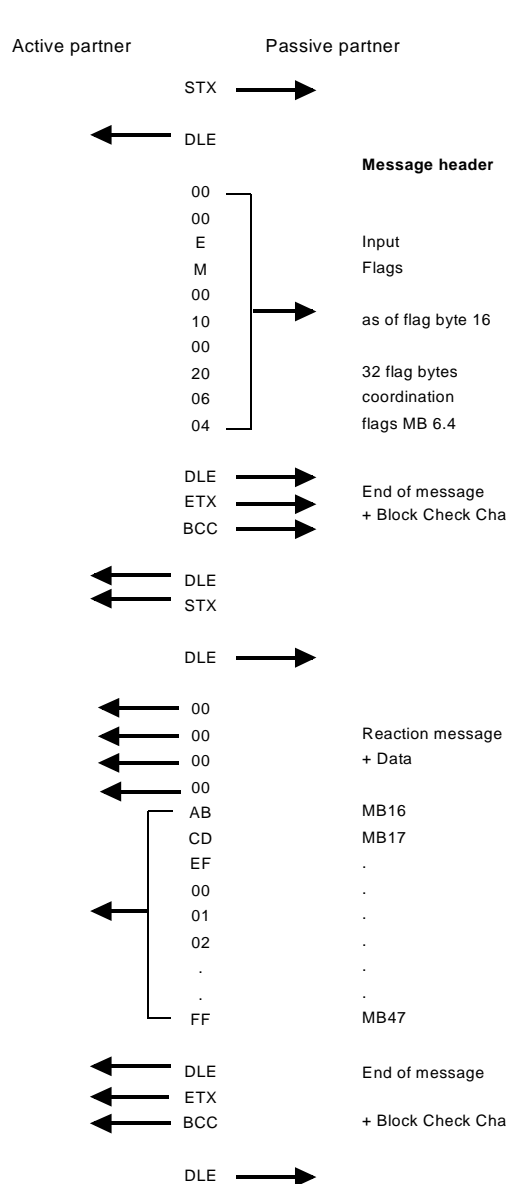
with N = 4 ... 127

Next reaction message

Byte	Hex	Description
0	FF	Flag for next reaction message
1	00	next reaction message
2	00	message
3	xx	Error code

**Structure of the input message**

*Sample input message*



**Normal message**

Byte	Hex	Description
0	00	Message identifier
1	00	Message identifier
2	E	Input command
3	X	Data type
4	xx	Parameter 1
5	xx	Destination
6	yy	Parameter 2
7	yy	Quantity
8	zz	Parameter 3
9	zz	Coordinnation flag

**Reaction message**

Byte	Hex	Description
0	00	Reaction message flag
1	00	Reaction message flag
2	00	Reaction message flag
3	xx	Error code
4	aa	Data
-	bb	
N	xy	

with N = 4 ... 127

When the data exceeds 128Byte, additional messages will be sent.

*Structure of additional messages*

**Next message**

Byte	Hex	Description
0	FF	Flag for next message
1	00	Flag for next message
2	E	Input command
3	X	data type

**Next reaction message**

Byte	Hex	Description
0	FF	Flag for next reaction message
1	00	Flag for next reaction message
2	00	Flag for next reaction message
3	xx	Error code
4	aa	Data
-	bb	
N	xy	

with N = 4 ... 127

**Coordination flags**

The coordination flag is set in the partner PLC in active-mode when a message is being received. This occurs for input as well as for output commands. When the coordination flag has been set and a message with this flag is received, then the respective data is not accepted (or transferred) and a reject message is sent (error code 32h). In this case the user has to reset the coordination flag in the partner PLC.

## Principles Modbus

**Overview** The Modbus protocol is a communication protocol that defines a hierarchic structure between a master and several slaves.  
Physically, Modbus transmits via a serial half-duplex core as point-to-point connection under RS232 or as multi-point connection under RS485.

**Master– slave communication** There are no bus conflicts for the master is only able to communicate with one slave at a time. After the master requested a message, it waits for an answer until a adjustable wait period has expired. During waiting is no other communication possible.

**Telegram structure** The request telegrams of the master and the respond telegrams of a slave have the same structure:

Start ID	slave address	Function code	Data	Flow control	End ID
----------	---------------	---------------	------	--------------	--------

**Broadcast with slave address = 0** A request may be addressed to a certain slave or send as broadcast message to all slaves. For identifying a broadcast message, the slave address 0 is set.  
Only write commands may be send as broadcast.

**ASCII, RTU mode** Modbus supports two different transmission modes:

- ASCII mode: Every Byte is transferred in 2 character ASCII code. The data is marked by a start and an end ID. This enables high control at the transmission but needs time.
- RTU mode: Every Byte is transferred as character. Thus enables a higher data through-put than the ASCII mode. Instead of start and end ID, RTU uses a time watcher.

The mode selection is at parameterization.

**Modbus at CP 240 from VIPA** The CP 240 module is a communication processor that occupies 16Byte each for in- and output data at a chooseable area in the CPU.  
The CP has to be included via the hardware configuration by means of the included GSD file.  
The Modbus-CP from VIPA may be deployed as master or as intelligent slave. In master operation the communication takes place via your PLC user application, that requires the SEND and RECEIVE handler blocks.



## CP 240 with 20mA/RS232C interface - Construction

### Properties

- The order number of the communication processor is: VIPA 240-1BA00 and VIPA 240-1BA10 with Modbus protocol.
- RS232C or 20mA interfacing depending on external connections.
- Interface compatible with SSM BG41-43 with MD26 (20mA/RS232) from VIPA and the Siemens CP525.
- Protocols supported: ASCII, STX/ETX, 3964(R) and RK512 or Modbus
- Active or passive 20mA interface
- 16Byte parameter data
- 8 receive buffers of 256Byte each and 1 send buffer with 256Byte.
- Diagnostic functions via backplane bus.
- Diagnostic interface with TTL levels.
- The serial interface is not isolated from the backplane bus
- Power supply via backplane bus

### Properties of the 20mA interface

- Logical levels represented by currents
- Data transfers over distances of up to 1000m depending on baudrate
- Data transfer rate up to 19.2kBaud
- Point-to-point connections (active/passive)
- Bus connection

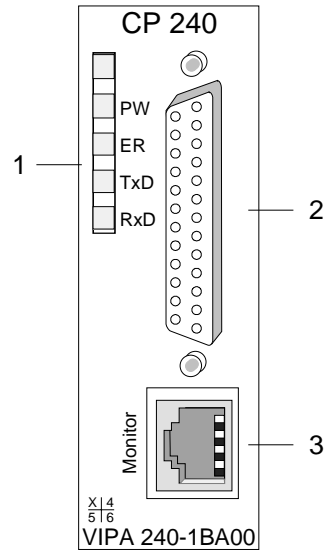
Options of the 20mA mode:

- Module is active station and supplies line current
- Module is passive station; partner station supplies the line current

### Properties of the RS232C interface

- Logical signals as voltage levels
- Point-to-point links with serial full-duplex transfer in 2-wire technology
- Data transfers over distances of up to 15m
- Data transfer rate up to 115kBaud

**Construction**



- [1] LED Status indicators
- [2] 25pin serial D-type socket (RS232C or 20mA)
- [3] Diagnostic socket for troubleshooting purposes

**Components**

**Power supply**

The communication processor receives power via the backplane bus. Please note that the module requires external DC 24V via the D-type socket if it should operate as an active 20mA interface. Please refer to "Deployment as active 20mA interface" for more detailed information.

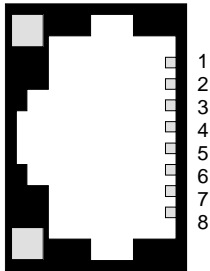
**LEDs**

The communication processor is provided with 4 LEDs for the purpose of displaying the operating status. The following table depicts the description and the color of these LEDs.

Name	Color	Description
PW	yellow	Indicates that power is available
ER	red	For Modbus this signalizes an internal error other protocols: error indicator for open circuit lines, overflow, parity or framing errors. The error LED is reset automatically after 4s. If diagnostics are enabled the error causes transmission of diagnostic bytes.
TxD	yellow	Transmit data
RxD	yellow	Receive data

**Diagnostic interface**

The RJ45 socket provides access to the RxD and TxD signals of the serial interface. The signals have already been converted to RS232 levels. The diagnostic interface has the following pin assignment:

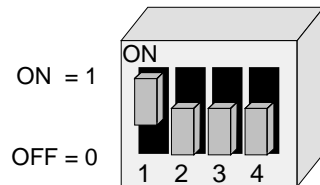


Pin	Assignment
1	reserved
2	RxD
3	TxD
4	reserved
5	GND
6	VCC (5V)
7	channel selection
8	channel selection

You may display and analyze the signals by means of a diagnostic cable and software that is available from VIPA (order no.: VIPA-UPI-FOX2).

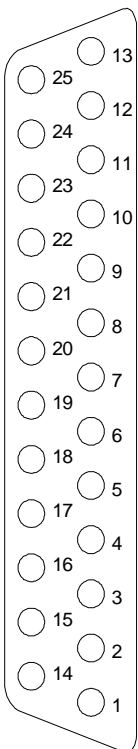


**Note!**



Please remember that you are only able to use the diagnostic facilities when the shown combination is set up on the DIP-switch of the diagnostic cable.

**25pin D-type socket**



Pin	RS232C	20mA	Pin	RS232C	20mA
25	not connected	not connected	13	reserved	RxD+
24	reserved	Current sink 20mA -Tx	12	reserved	Current srce 20mA +Tx
23	RI	reserved	11	reserved	+24V
22	reserved	GND 24V	10	reserved	TxD+
21	reserved	Current sink 20mA -Rx	9	not conn.	not connected
20	Selection RS232C / 20mA		8	DTR	reserved
19	reserved	TxD-	7	GND	GND
18	CD	reserved	6	DSR	reserved
17	GND	GND	5	CTS	reserved
16	reserved	Current srce 20mA +Rx	4	RTS	reserved
15	+5V	+5V	3	RxD	reserved
14	reserved	RxD-	2	TxD	reserved
			1	shield	shield

**Wiring**

The communication processor is equipped with a single interface as shown above. This interface can be set to operate either as RS232C or as 20mA interface. The selection of the type of operating mode is controlled by means of pin 20 on the 25-pin socket:

- pin 20 at ground → RS232C interface
- pin 20 open → 20mA interface

In the 20mA mode the interface can be operated in "active" or "passive" mode.

**20mA current source**

The CP 240 is provided with a 20mA current source for the active mode. This requires a DC 24V power supply via pins 11 (+24V) and 22 (24V ground).

**RS232C interface**

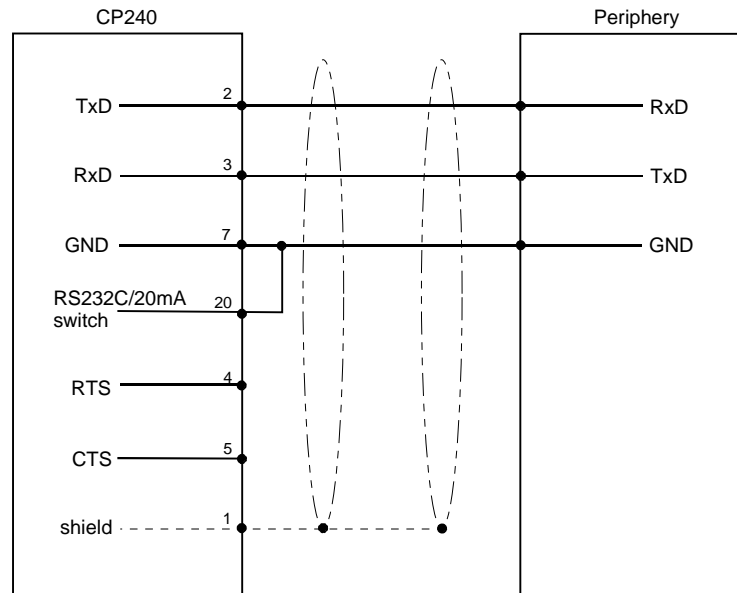
The CP 240 currently supports the following RS232C signals:

**TxD Transmit Data**

The transmit data is transferred via the TxD line. When the transmit line is not used the CP 240 holds it at a logical "1".

**RxD Receive Data**

The receive data arrives via the RxD line. When the receive line is not in use, it must be held at a logical "1" by the transmitting station.



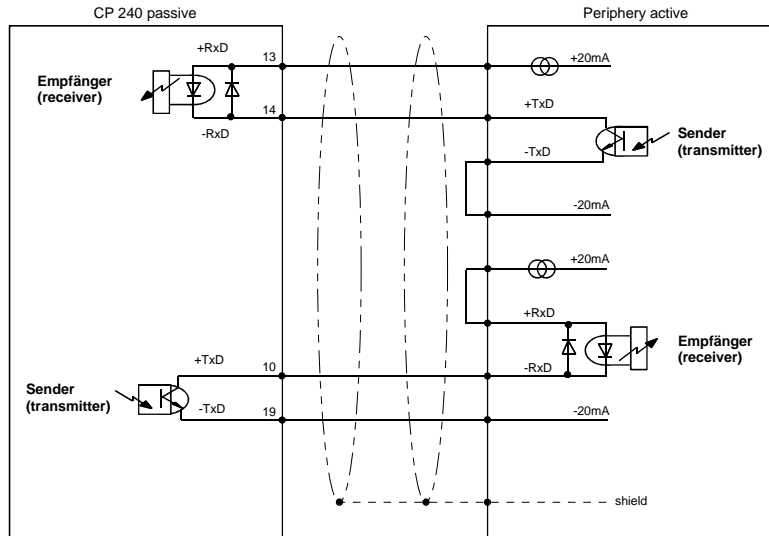
**20mA interface**

*Options of the 20mA interface*

- Module as passive station, the partner station supplies the line current.
- Module as active station, it supplies the line current.

*Passive 20mA interface*

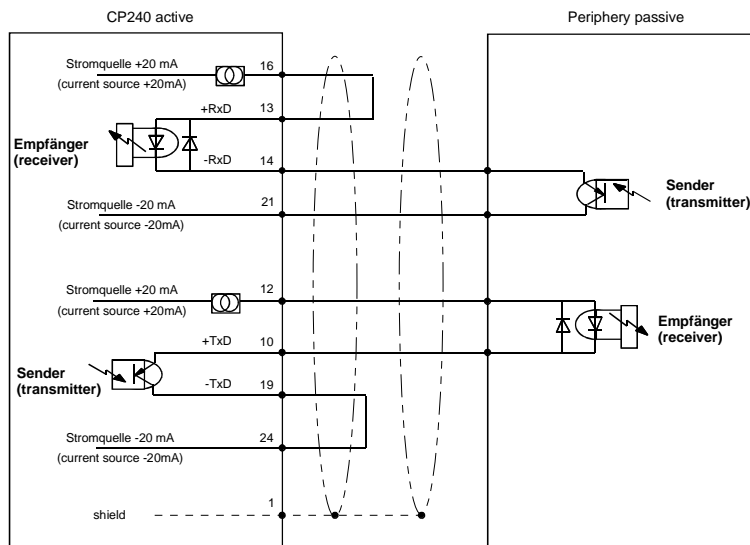
When the CP 240 is used with a "passive" 20mA interface, the required current is supplied by the periphery. In this case the CP 240 does not require an external power source.



*Active 20mA interface*

When the CP 240 is used as the "active" 20mA interface, the internal current source supplies the required 20mA. In this case the connected periphery represents the passive station.

Please note that the current source in the CP 240 requires an external source of DC 24V power connected to pin 11 and 22.



## CP 240 with RS422/RS485 interface - Construction

### Properties

- The communication processor has the order no.: VIPA 240-1CA00 and VIPA 240-1CA10 with Modbus protocol
- RS422 or RS485 interface
- Interface compatible with SSM BG41-43 with MD21 (RS422/RS485) from VIPA and CP525 from Siemens
- Supports the protocols ASCII, STX/ETX, 3964(R), RK512 or Modbus
- 16Byte parameter data
- 8 receive buffers of 256Byte each and 1 transmit buffer of 256Byte
- Diagnostic function via the backplane bus
- Diagnostic interface employing TTL levels.
- Isolation with respect to the backplane bus
- Power supplied via the backplane bus

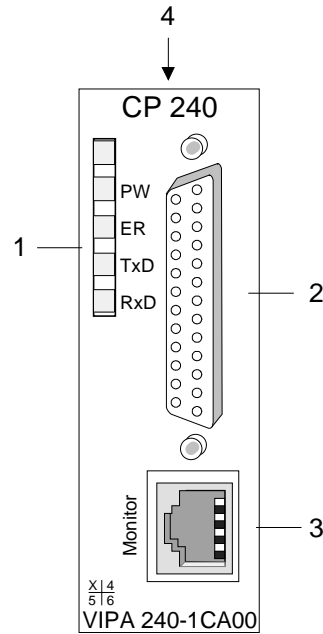
### RS422 interface

- Logical states are represented by differential voltage levels on two twisted cores
- Point-to-point link with a serial full-duplex transfer in 4-wire technology
- Multidrop connection
- High interference immunity
- Up to 16 partner stations
- Data transfer over distances of up to 1000m
- Data transfer rate up to 115kBaud

### RS485 interface

- Logical states are represented by differential voltage levels on two twisted cores
- Serial bus connections by means of a two wire half-duplex link
- Multidrop connections
- High noise immunity
- Up to 32 partner stations
- Data transfer over distances of up to 500m
- Data transfer rate up to 115kBaud

**Construction**



- [1] LED status indicators
- [2] 25pin serial D-type socket (RS422/485)
- [3] Diagnostic socket for troubleshooting purposes
- [4] Switchable terminating resistor

**Components**

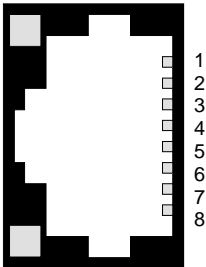
**Power supply** The communication processor receives power via the backplane bus.

**LEDs** The communication processor is provided with 4 LEDs for the purpose of displaying the operating status. The description and the color of these LEDs is depicted by the following table.

Name	Color	Description
PW	yellow	Indicates that power is available
ER	red	Indicates errors: open circuit lines, overflow, parity or framing errors. The error LED is reset automatically after 4s. If diagnostics are enabled the error causes transmission of diagnostic bytes.
TxD	yellow	Transmit data
RxD	yellow	Receive data

**Diagnostic interface**

The RJ45 socket provides access to the RxD and TxD signals of the serial interface. The signals have already been converted to RS232 levels. The diagnostic interface has the following pin assignment:

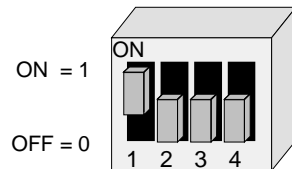


Pin	Assignment
1	reserved
2	RxD
3	TxD
4	reserved
5	GND
6	VCC (5V)
7	channel selection
8	channel selection

You may display and analyze the signals by means of a diagnostic cable and software that is available from VIPA (order no.: VIPA-UPI-FOX2).

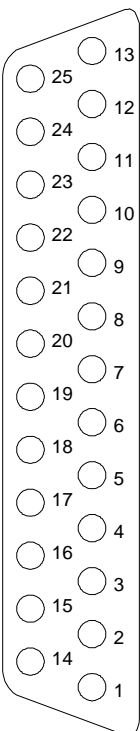


**Note!**



Please remember that you are only able to use the diagnostic facilities when the shown combination is set on the DIP-switch of the diagnostic cable.

**25pin D-type socket**



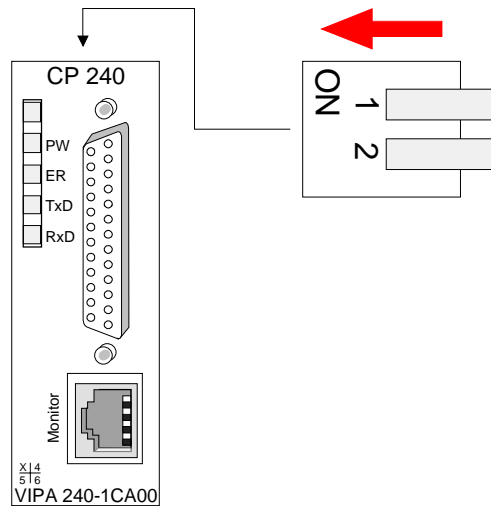
Pin	RS422/485	Pin	RS422/485
25	not connected	13	CTS-
24	not connected	12	CTS+
23	not connected	11	not connected
22	not connected	10	RTS+
21	not connected	9	RTS-
20	not connected	8	RxD+
19	not connected	7	GND
18	not connected	6	RxD-
17	GND iso	5	TxD-
16	not connected	4	TxD+
15	+5V	3	5V iso
14	not connected	2	not connected
		1	shield



**Terminating resistor**

A 2pin DIP-switch is accessible at the top of the module. This switch connects a 100Ω terminating resistor between the RxD and the TxD lines respectively.

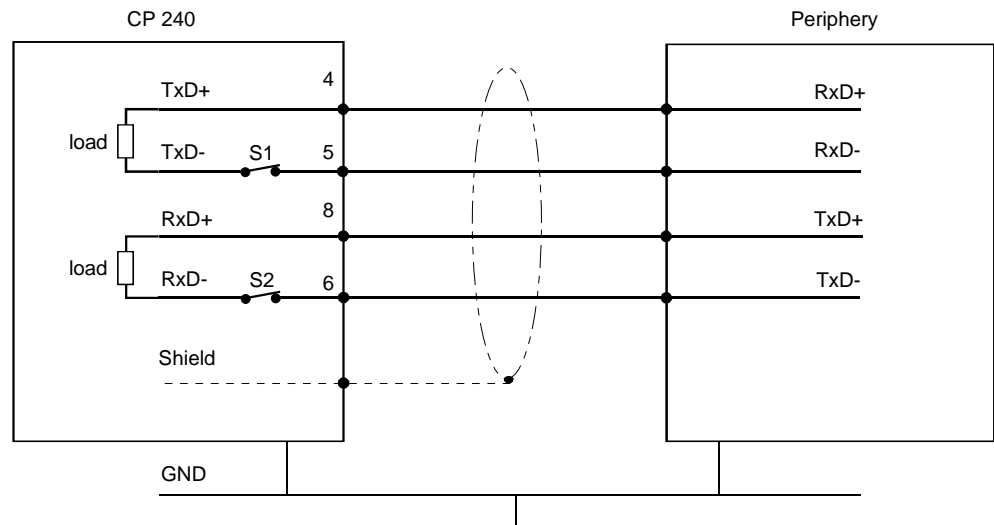
This is necessary in case of longer communication lines or higher data communication rates when the module is located at the physical end of the bus.



**Wiring**

The interface may be used for point-to-point links (RS422) or for a bus system where the transmission and reception is carried by the same line (RS485). In this case a bus master controls the operating mode by means of the SEL signals.

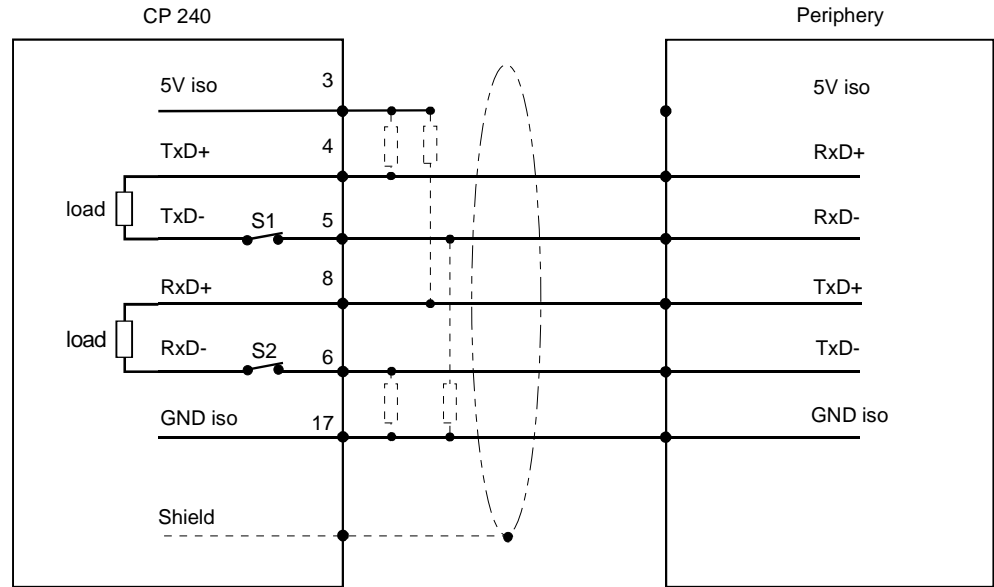
**RS422 dc coupled**



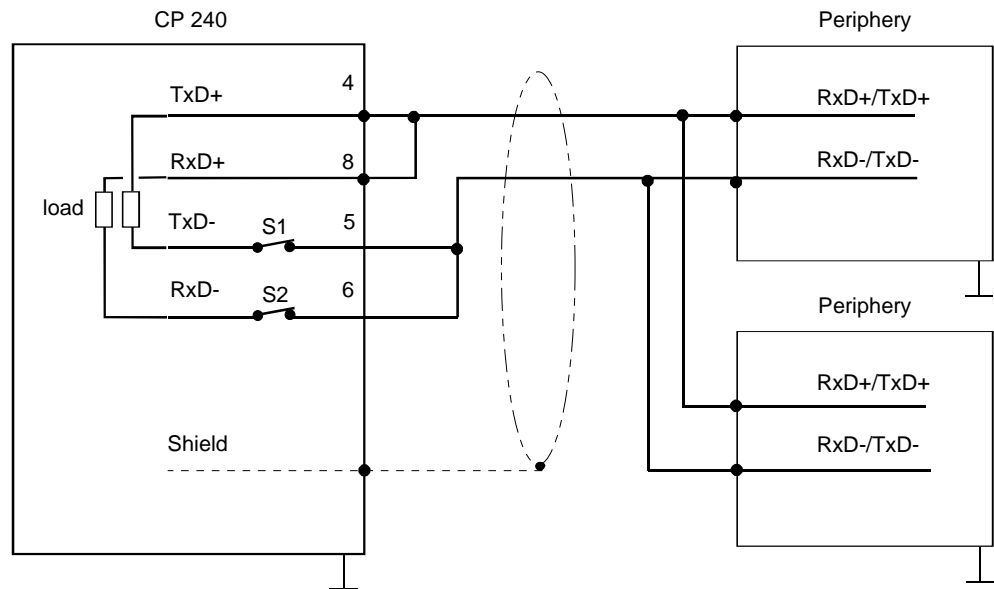
**Attention!**

The grounds of the device interfaces must be connected.

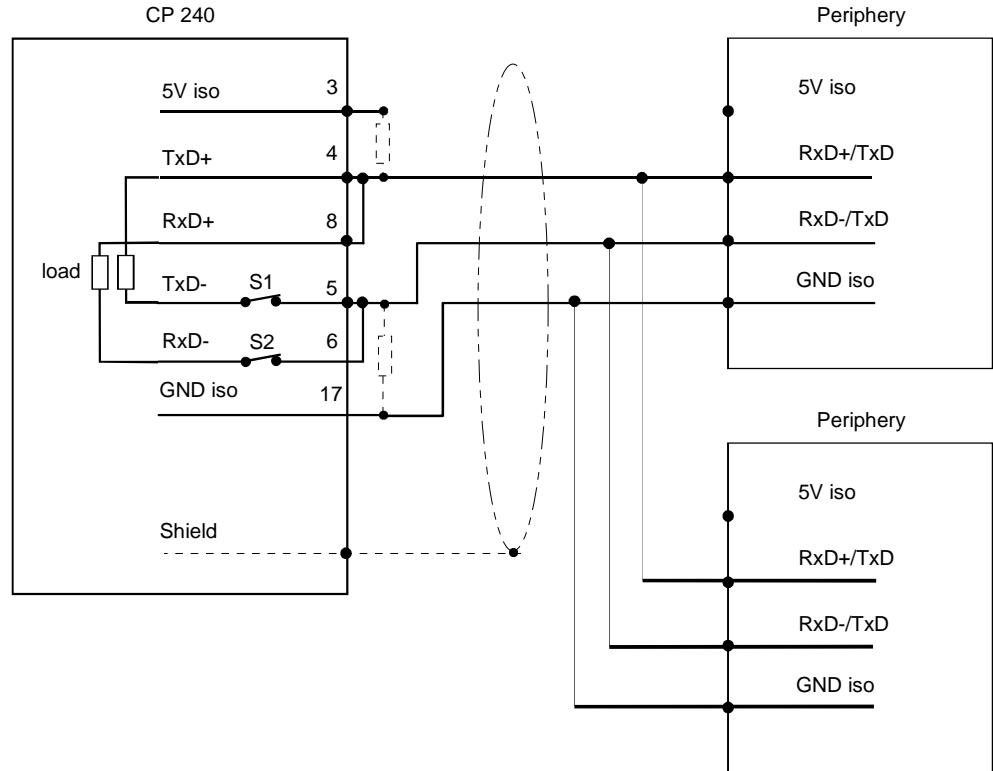
**RS422 isolated**



**RS485 dc coupled**



**RS485 isolated**



**Defined static levels by means of resistors**

Pin 3 of the isolated interfaces carries the isolated 5V supply with the respective ground GND on pin 17. You may use this isolated voltage to provide defined static voltage levels on the signaling lines by means of resistors and ensure that reflections are reduced to a minimum.

## Parameterization

### General

You may configure the CP 240 by means of 16Byte of configuration data. The structure of the parameter data depends on the selected protocol or procedure. Below follows a list of the parameter bytes with the respective default values.

### Structure of the parameter bytes for ASCII

Byte	Function	Range	Default parameter
0	Baudrate	0h: Default (9600Baud) 1h: 150Baud 2h: 300Baud 3h: 600Baud 4h: 1200Baud 5h: 1800Baud 6h: 2400Baud 7h: 4800Baud 8h: 7200Baud 9h: 9600Baud Ah: 14400Baud Bh: 19200Baud Ch: 38400Baud Dh: 57600Baud Eh: 115200Baud	0: 9600Baud
1	Protocol	1: ASCII	1 (ASCII)
2	Bit 1/0 data bits	00b: 5 data bits 01b: 6 data bits 10b: 7 data bits 11b: 8 data bits	11b: 8 data bits
	Bit 3/2 Parity	00b: none 01b: odd 11b: even	00b: none
	Bit 5/4 Stop bits	01b: 1 10b: 1.5 11b: 2	01b: 1 Stop bit
	Bit 7/6 Flow control	00b: none 01b: Hardware 10b: XON/XOFF	00b: none
3	Reserved	0	0
4	ZNA (*20ms)	0..255	0
5	ZVZ (*20ms)	0..255	10
6	no. of receive buffers	1..8	1
7...15	reserved		

**Structure of  
parameter bytes  
for STX/ETX**

Byte	Function	Range of values	Default parameters
0	Baudrate	0h:.. Default (9600Baud) 1h:.. 150Baud 2h:.. 300Baud 3h:.. 600Baud 4h:.. 1200Baud 5h:.. 1800Baud 6h:.. 2400Baud 7h:.. 4800Baud 8h:.. 7200Baud 9h:.. 9600Baud Ah: . 14400Baud Bh: . 19200Baud Ch: . 38400Baud Dh: . 57600Baud Eh: . 115200Baud	0: 9600Baud
1	Protocol	2:.... STX/ETX	2 (STX/ETX)
2	Bit 1/0 Data bits	00b: 5 Data bits 01b: 6 Data bits 10b: 7 Data bits 11b: 8 Data bits	11b: 8 Data bits
	Bit 3/2 Parity	00b: none 01b: odd 11b: even	00b: none
	Bit 5/4 Stop bits	01b: 1 10b: 1.5 11b: 2	01b: 1 Stop bit
	Bit 7/6 Flow control	00b: none 01b: Hardware 10b: XON/XOFF	00b: none
3	Reserved	0	0
4	ZNA (*20ms)	0..255	0
5	TMO (*20ms)	0..255	10
6	Number of start flags	0..2	01
7	Start flag 1	0..255	02
8	Start flag 2	0..255	0
9	Number of end flags	0..2	01
10	End flag 1	0..255	03
11	End flag 2	0..255	0
12	not used		
13	not used		
14	not used		
15	not used		

**Structure of the parameter bytes for 3964(R) / 3964(R) with RK512**

Byte	Function	Range	Default parameters
0	Baudrate	0h:... Default (9600Baud) 1h:... 150Baud 2h:... 300Baud 3h:... 600Baud 4h:... 1200Baud 5h:... 1800Baud 6h:... 2400Baud 7h:... 4800Baud 8h:... 7200Baud 9h:... 9600Baud Ah: . 14400Baud Bh: . 19200Baud Ch: . 38400Baud Dh: . 57600Baud Eh: . 115200Baud	0: 9600Baud
1	Protocol	3:.... 3964 4:.... 3964R 5:.... 3964 + RK512 6:.... 3964R + RK512	3: 3964
2	Bit 1/0 Data bits	00b: 5 Data bits 01b: 6 Data bits 10b: 7 Data bits 11b: 8 Data bits	11b: 8 Data bits
	Bit 3/2 Parity	00b: none 01b: odd 11b: even	00b: none
	Bit 5/4 Stop bits	01b: 1 10b: 1.5 11b: 2	01b: 1 Stop bit
	Bit 7/6 Flow control	00b: none 01b: Hardware 10b: XON/XOFF	00b: none
3	Reserved	0	0
4	ZNA (*20ms)	0..255	0
5	ZVZ (*20ms)	0..255	10
6	QVZ (*20ms)	0..255	25
7	BWZ (*100ms)	0..255	100
8	STX repetitions	0..255	3
9	DBL	0..255	6
10	Priority	0:.... low 1:.... high	0: low
11	reserved	0	0
12	QVZ (*100ms) user acknowledgment RK512	0..255	50
13	not used		
14	not used		
15	not used		

**Parameter structure at Modbus**

Byte	Function	Range	Default parameter
0	Baudrate	0h: 9600Baud 6h: 2400Baud 7h: 4800Baud 9h: 9600Baud Ah: 14400Baud Bh: 19200Baud Ch: 38400Baud	0h: 9600Baud
1	Protocol	Ah: Modbus master ASCII Bh: Modbus master RTU Ch: Modbus slave ASCII Dh: Modbus slave RTU	Bh: Modbus master RTU
2	Bit 1/0 Data bits	00b: 5 Data bits 01b: 6 Data bits 10b: 7 Data bits 11b: 8 Data bits	11b: 8 Data bits
	Bit 3/2 Parity	00b: none 01b: odd 11b: even	00b: none
	Bit 5/4 Stop bits	01b: 1 10b: 1.5 11b: 2	01b: 1 Stop bit
	Bit 7/6 Flow control	00b: none 01b: Hardware 10b: XON/XOFF	00b: none
3	Reserved	0	0
4	Address	1...255	1
5	Debug	0: Debug off 1: Debug on	0
6...7	Wait period	0: automatic calculation 1 ... 60000: Time in ms	0
8	not connected		
9	not connected		
10	not connected		
11	not connected		
12	not connected		
13	not connected		
14	not connected		
15	not connected		

**Note!**

If no parameterization is present and the CP 240 is linked-up via auto addressing, the CP has the following default parameters:

Baudrate: 9600Baud, Protocol: ASCII, data bits: 8, **Parity: even**, Stopbits: 1, Flow control: no, ZNA: 0, ZVZ: 200ms, Receivebuffer: 1

---

**Parameter description**

**Baudrate (for all protocols)** The data communication rate in bit/s (Baud).  
You may select one of the following values:

0h:	Default (9600Baud)
1h:	150Baud
2h:	300Baud
3h:	600Baud
4h:	1200Baud
5h:	1800Baud
6h:	2400Baud
7h:	4800Baud
8h:	7200Baud
9h:	9600Baud
Ah:	14400Baud
Bh:	19200Baud
Ch:	38400Baud
Dh:	57600Baud
Eh:	115200Baud

*Default: 0 (9600Baud)*

**Protocol** The protocol that you wish to employ. This setting determines the further structure of the parameter data. Depending on the chosen CP 240 module the following options are available:

only VIPA 240-1BA00 and VIPA 240-1CA00:

1:	ASCII
2:	STX/ETX
3:	3964
4:	3964R
5:	3964 and RK512
6:	3964R and RK512

only VIPA 240-1BA10 and VIPA 240-1CA10:

Ah:	Modbus master with ASCII
Bh:	Modbus master with RTU
Ch:	Modbus slave with ASCII
Dh:	Modbus slave with RTU



**Data communication parameter byte (for all protocols)**

Here you define the physical data transfer parameters. The structure of the Byte is as follows:

Byte	Function	Range	Default parameter
2	Bit 1/0 Data bits	00b: 5 Data bits 01b: 6 Data bits 10b: 7 Data bits 11b: 8 Data bits	11b: 8 Data bits
	Bit 3/2 Parity	00b: none 01b: odd 11b: even	00b: none
	Bit 5/4 Stop bits	01b: 1 10b: 1.5 11b: 2	01b: 1 Stop bit
	Bit 7/6 Flow control	00b: none 01b: Hardware 10b: XON/XOFF	00b: none

*Data bits*

Quantity of *data bits* that represent a character.

*Parity*

The parity depends on the value and may be even or odd. For the purposes of the parity check, the information bits are expanded by the parity bit. The value of the parity bit ("0" or "1") completes the value of all the bits to obtain a pre-arranged state. If the parity was not specified, the parity bit is set to "1" but it is not included in the assessment.

*Stop bits*

The stop bits are appended to each character and signify the end of the character.

*Flow control*

This is a mechanism that synchronizes the data transfer when the transmitting station sends the data faster than it can be processed by the receiving station. Flow control can be hardware- or software-based (XON/XOFF). Hardware flow control employs the RTS and CTS lines and these must therefore be wired accordingly.

Software flow control employs the control characters XON=11h and XOFF=13h. Please remember that your data must not contain these control characters.

*Default: 13h (data bits: 8, parity: none, stop bits: 1, flow control: none)*

<b>Time delay after command (ZNA)</b> (for all protocols except Modbus)	The delay time that must expire before a command is executed. The ZNA is specified in units of 20ms. <i>Range: 0 ... 255</i>	<i>Default: 0</i>
<b>Character delay time (ZVZ)</b> (for ASCII, 3964(R) and RK512)	The character delay time defines the maximum time that may expire between two characters of a single messages during the reception of the message. The ZVZ is defined in units of 20ms. When the ZVZ=0 the character delay time (ZVZ) will be calculated automatically (about double character time). <i>Range: 0 ... 255</i>	<i>Default: 10</i>
<b>Number of receive buffers</b> (only for ASCII)	Defines the number of receive buffers. When only 1 receive buffer is available no more data can be received while the receive buffer is occupied. The received data can be redirected into an unused receive buffer when you chain up to a maximum of 8 receive buffers. <i>Range: 1 ... 8</i>	<i>Default: 1</i>
<b>Timeout (TMO)</b> (only for STX/ETX)	TMO defines the maximum time between two messages. TMO is specified in units of 20ms. <i>Range: 0 ... 255</i>	<i>Default: 10</i>
<b>Number of start flags</b> (only for STX/ETX)	You can select 1 or 2 start flags. When you select "1" as start flag, the contents of the 2 <sup>nd</sup> start flag (byte 8) is ignored. <i>Range: 0 ... 2</i>	<i>Default: 1</i>
<b>Start flag 1 and 2 (STX)</b> (only for STX/ETX)	The ASCII value of the start character that precedes a message to signify the start of a data transfer. You may select 1 or 2 start characters. When you are using 2 start characters you have to specify "2" at "Number of start flags". <i>Number of start flags: Range: 0 ... 2</i> <i>Start character 1, 2: Range: 0 ... 255</i>	<i>Default: 1</i> <i>Default: 2 (flag 1)</i> <i>0 (flag 2)</i>
<b>End flag 1 and 2 (ETX)</b> (only for STX/ETX)	The ASCII value of the end character that follows a message to signify the end of the data transfer. You may specify 1 or 2 end characters. When you are using 2 end characters you have to enter a "2" for "number of end flags". <i>Number of end flags: Range: 0 ... 2</i> <i>End character 1, 2: Range: 0 ... 255</i>	<i>Default: 1</i> <i>Default: 3 (char. 1)</i> <i>0 (char. 2)</i>

<b>Delayed acknowledgment time (QVZ)</b> (for 3964(R), RK512)	The delayed acknowledgment time defines the maximum time for the acknowledgment from the partner when the connection is being established. The QVZ is specified in units of 20ms. <i>Range: 0 ... 255</i> <i>Default: 25</i>
<b>Block delay time (BWZ)</b> (for 3964(R), RK512)	The BWZ is specified in units of 100ms. <i>Range: 0 ... 255</i> <i>Default: 100</i>
<b>STX repetitions</b> (for 3964(R), RK512)	Maximum number of allowed attempts for a CP 240 to establish a connection. <i>Range: 0 ... 255</i> <i>Default: 3</i>
<b>Repetitions of data blocks (DBL)</b> (for 3964(R), RK512)	Maximum number of message repetitions (incl. the 1 <sup>st</sup> telegram) if an error occurs. <i>Range: 0 ... 255</i> <i>Default: 6</i>
<b>Priority</b> (for 3964(R), RK512)	A communication partner has a high priority when its transmit request supersedes the transmit request of a partner. When the priority is lower, it must take second place after the transmit request of the partner. The priorities of the two partners must be different for the 3964(R) protocols. You may select one of the following settings: 0: low 1: high <i>Default: 0 (low)</i>
<b>QVZ user acknowledgment RK512</b> (only for RK512)	This is the delay time during which an acknowledgment must be received from the partner to indicate that the data was received and processed (reaction message). The QVZ is specified in units of 100ms. <i>Range: 0 ... 255</i> <i>Default: 50</i>
<b>Address</b> (only for Modbus)	If the module has slave character, you adjust the Modbus slave address here. <i>Range: 1...255</i> <i>Default: 1</i>
<b>Debug</b> (only for Modbus)	This mode is reserved for internal tests. This function should always be deactivated. <i>Range: 0, 1</i> <i>Default: 0</i>

## Access to the CP 240 interface under ASCII, STX/ETX, 3964(R)

The access methods described here are **not** valid for the **Modbus protocol!**

### Send and receive data

Data that is written from the CP 240 to the concerning data channel via the backplane bus, is written to the according send buffer (256Byte) and from there put out via the interface.

When the communication processor receives data via the interface, the data is stored in a ring buffer (8x256Byte). The received data may be read from the CPU via the data channel.

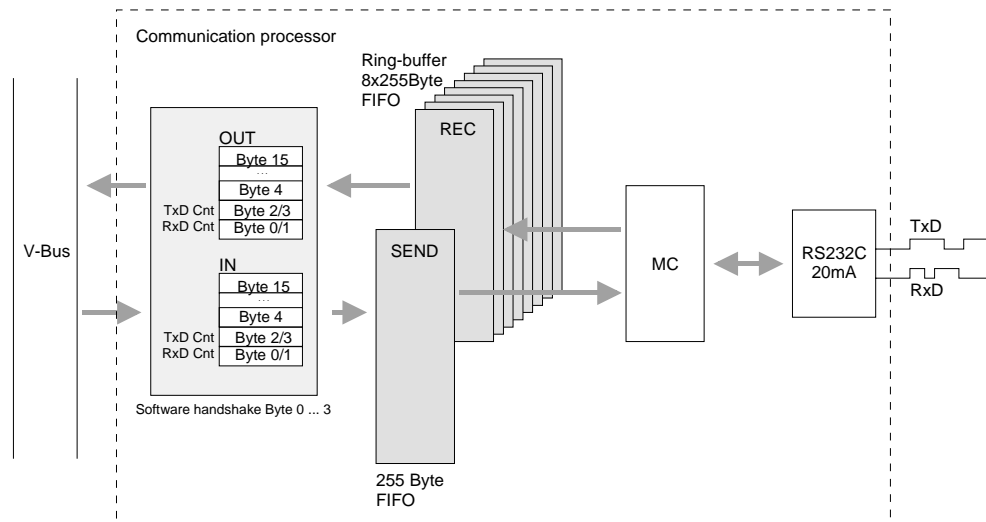
### Communication via backplane bus

The exchange of received telegrams via the backplane bus happens asynchronously. When a complete telegram has arrived via the serial interface (expiration of the ZVZ), this is stored in a ring buffer of 8x256Byte. The length of the ring buffer determines the max. length of a telegram. There may be stored 8 independent telegrams of 256Byte each. When the buffer is filled up, new telegrams are rejected. A complete telegram is divided into blocks of 12Byte and transferred to the backplane bus. The reassembly of the data blocks has to take place inside the CPU.

### Tasks of the CPU

The CPU has to split the telegram to send into blocks of 12Byte and transfer them via the backplane bus to the CP 240. In the CP 240 these blocks are assembled in the send buffer, proofed for completeness and then sent to the serial interface. For the data transfer via the backplane bus is asynchronous, a "software handshake" is used between the CP 240 and the CPU. The register for the data transfer from the CP 240 have a width of 16Byte. The bytes 0 to 3 (word 0 and 2) are reserved for the handshake.

The following picture illustrates this:



---

**Software handshake**

For the deployment of the CP 240 together with a System 200V CPU VIPA offers you a series of standard handler blocks that provide the software handshake comfortable and easy.

At deployment of the CP 240 without handler blocks, the functionality is elucidated with an example of data send and receive.

**Example SEND data**

For example, a telegram with 30Byte length is to send. The CPU writes the first 12Byte user data of the telegram into the Bytes 4 to 15. Byte 2/3 contain the telegram length, i.e. "30". The CP 240 receives the data via the backplane bus and copies the 12Byte user data into the send buffer. For the acknowledgement of the telegram the CP 240 writes the value "30" back to Byte 2/3 (length of the telegram).

At reception of the "30", the CPU can send further 12Byte user data to Byte 4 to 15 and the rest length of the telegram ("18" Byte) to Byte 2/3 to the CP 240. Again, this stores the user data in the send buffer and sends back the length information ("18") in Byte 2/3 to the CPU.

The CPU receives the "18" and sends the remaining 6Byte user data in the Bytes 4 to 9 and the according rest length ("6") in Byte 2/3 to the CP 240. The user data is stored in the send buffer and the value "6" is send back to the CPU via Byte 2/3.

The CPU receives the "6" and sends back a "0" via Byte 2/3. The CP 240 now initializes the sending of the telegram via the serial interface. After data transfer is completed, the CP 240 sends back a "0" to the CPU via Byte 2/3.

At reception of the "0", the CPU is able to send a new telegram to the CP 240.

**Example RECEIVE data**

The interface of the CP 240 has e.g. received a telegram with a length of 18Byte via the serial interface. The CP 240 writes the 12Byte user data into the Bytes 4 to 15 of the receive buffer and the telegram length (i.e. "18") into Byte 0/1. The data is transferred to the CPU via the backplane bus. The CPU stores the 12Byte user data and sends back the length value "18" to the CP 240.

At reception of the "18", the CP 240 writes the remaining 6Byte user data into the Bytes 4 to 9 of the receive buffer and the received length of user data ("6") in Bytes 0/1.

Having received the "6", the CP 240 returns the value "0" via Byte 0/1, i.e. the telegram has been completed. The CPU acknowledges with another "0" in Byte 0/1 to the CP 240.

Now the CP 240 may send another telegram to the CPU.

## Deployment under Modbus

### Overview

You may deploy the CP 240 Modbus as well in master as in slave mode. Independently from this, the module occupies each 16Byte for input and output data at a chooseable area in the CPU.

In master mode, the communication takes place via data blocks by using the SEND/RECEIVE handler blocks. At deployment of blocks you may transfer up to 250Byte user data.

In slave mode, the length of the user data for in- and output is limited to 16Byte. For the deployment of a slave, no handler blocks are required. It has only to be parameterized.

The parameterization is executed during the hardware configuration of the CPU by using the delivered GSD file.

After every start-up of the CPU, this transfers the parameters to the Modbus module. If the parameters are valid, they are adopted. Now the Modbus module is ready for communication.

### Operation requirements

The following components are necessary for the deployment of the System 200V Modbus modules:

- VIPA 240-1BA10 resp. VIPA 240-1CA10 (CP 240 Modbus)
- VIPA CPU 21x (programmable in Step<sup>®</sup>7 from Siemens) or VIPA CPU 24x (programmable in Step<sup>®</sup>5 from Siemens).
- Configuration tool VIPA MC5 or Step<sup>®</sup>7 Manager (V5.1) from Siemens
- Programming cable for MPI connection (Green Cable from VIPA)
- GSD file **VIPA04D5.gsd** (V1.31 or higher)
- VIPA handler blocks Fx000002\_V109.zip
- Modbus cable

### Installation

The installation of the Modbus module has the following approach:

1. Install the GSD file **VIPA04D5.gsd** in your configuration tool.
2. Install the VIPA Library **Fx000002\_V109.zip** with the handler blocks in the configuration tool.
3. Fix the parameters for the CP 240 and transfer your configuration into your PLC.
4. Connect the Modbus cable.

### Configuration via GSD

The address assignment and the parameterization of the Modbus module happens via the STEP<sup>®</sup>7 Manager from Siemens by means of a virtual Profibus system. For the Profibus interface is standardized through software, the functionality of the module can be guaranteed when you include the GSD file in the STEP<sup>®</sup>7 Manager from Siemens. You transfer your project into the CPU via MPI.

**Include GSD**

The following steps are required for including the GSD:

- Copy the delivered VIPA GSD file **VIPA04D5.GSD** (V.1.31 or higher) into your GSD directory ...\\siemens\\step7\\s7data\\gsd
- Start the hardware configurator from Siemens
- Close all projects
- Choose **Options** > *Install new GSD*
- Select **VIPA04D5.gsd**

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

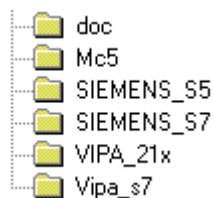
**Include  
VIPA library**

The VIPA specific SFCs are included to the consignment of the CPUs and the CPs in form of libraries. These are self-extracting exe-files.

When you want to employ VIPA specific SFCs, you have to import them into your project.

The following steps are required:

- Extract the file, execute FX00000z\_Vxxx.exe:  
Double-click at FX00000z\_Vxxx.exe to start the integrated extracting program. Choose the destination directory and click at <Extract>.
- "Dearchive" library:  
Start the STEP<sup>®</sup>7 Manager from Siemens. Via **File** > *Dearchive* you open a dialog window where you may choose the according archive. You will find the VIPA SFC library in the directory VIPA\_S7. The file name is VIPA.ZIP.

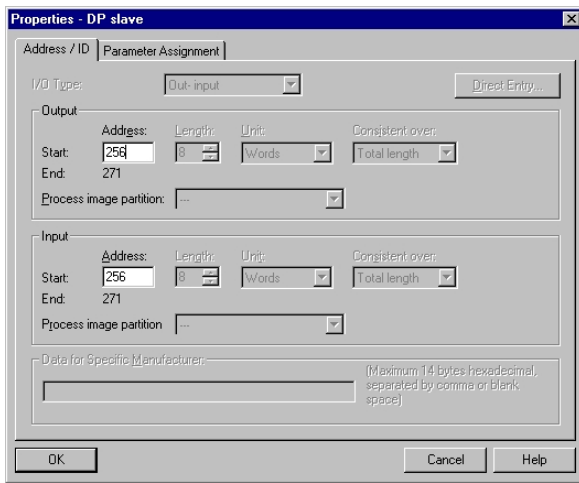


Choose VIPA.ZIP and click at [Open].

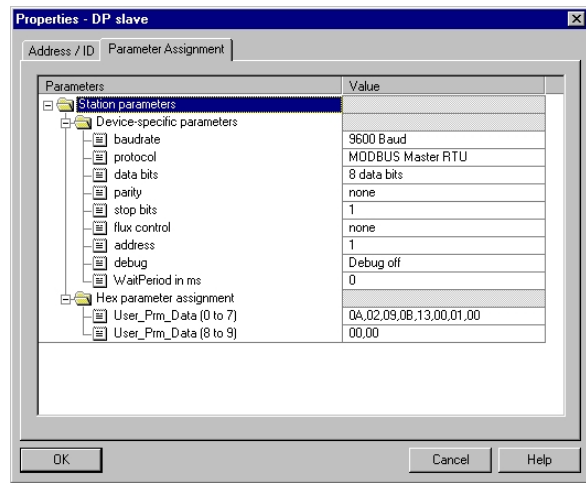
- Choose the destination directory where the handling block shall be stored and start the extraction with [OK].
- Open the library and transfer the SFCs into the project:  
After the extraction, open the library. Open your project and copy the required SFCs from the library into the directory "library" of your project. Now your user application allows you access to the VIPA specific handler blocks.

**Parameterization** the parameterization takes place via the hardware configuration. The following parameters are at your disposal:

Dialog for address input



Dialog for Modbus parameters



A closer description is to find further above in this chapter.

**Master mode**

The master mode is selected during parameterization. By means of the SEND and RECEIVE handler blocks from VIPA you may send resp. receive any Modbus telegram. The telegram structure is the same for the user in ASCII and RTU mode, for telegram start and end flag, check sum and protocol monitoring is handled by the module.

**Slave mode**

The slave mode is selected during parameterization. Using the slave mode, the module works as intelligent slave, processing up to 16Byte in- and output data. At deployment as slave, you have to assign a slave address during parameterization.

**Modbus slave function codes**

Running in slave mode, the CP 240 Modbus from VIPA is able to process the following function codes:

Code	Command	Description
01h, 02h	Read n Bits	read n bits
03h, 04h	Read n Words Status	read n words
05h	Write one Bit	write one output bit
06h	Write one Word	write one output word
0Fh	Write n Bits	write n bits
10h	Write n Words	write n words



**Commissioning**

After starting the power supply, the LEDs ER, TxD and RxD at the Modbus module are on. The module signalizes that there are no valid parameters from the CPU. As soon as you switch the CPU to RUN, the Modbus parameters are transferred to the module. If the parameters are valid, the LEDs ER, TxD and RxD are extinguishing. Now the Modbus module is ready for communication.

At deployment in master mode you may execute according write/read commands via your user application.

If the ER-LED remains flashing, an internal error occurred. If this is only a momentary lapse, you may clear it with a STOP-RUN switch of the CPU.

**Access to several slaves**

When deploying several slaves, bus conflicts are not possible because the master is only able to communicate with one slave at a time. The master sends a command telegram to the slave specified via the address and waits a defined time for the respond telegram of the slave. During the waiting period no other communication is possible.

For the communication with several slaves, a SEND data block for the command telegram and a RECEIVE data block for the respond telegram is required for every slave.

An application with more than one slave would exist of an according number of data blocks with the concerning commands.

These are processed in line:

1<sup>st</sup> slave:      Send command telegram to slave address 1<sup>st</sup> slave  
                  Receive respond telegram from slave address 1<sup>st</sup> slave  
                  Analyze respond telegram

2<sup>nd</sup> slave:      Send command telegram to slave address 2<sup>nd</sup> slave  
                  Receive respond telegram from slave address 2<sup>nd</sup> slave  
                  Analyze respond telegram

... etc.

A request may be directed to a certain slave or be sent as broadcast message to all slaves. To mark a broadcast message, the slave address "0" is set.

Only write commands may be sent as broadcast.

**Note!**

After a broadcast, the master does not wait for a respond telegram.

## Modbus function codes

### Outline

The following Modbus function codes are implemented in slave mode:

Code	Command	Description
01h, 02h	Read n Bits	read n bits
03h, 04h	Read n Words Status	read n words
05h	Write one Bit	write one output bit
06h	Write one Word	write one output word
0Fh	Write n Bits	write n bits
10h	Write n Words	write n words



### Note!

The shown check sums CRC at RTU and LRC at ASCII mode are automatically added to every telegram. They are not shown in the data block.

For the Byte sequence in a word is always valid:

1 word	
High Byte	Low Byte

### Read n Bits 01h, 02h

This function enables the reading from the slave in bits.

#### Command telegram

Slave address	Function code	Address 1 <sup>st</sup> Bit	Number of Bits	Check sum CRC/LRC
1 Byte	1 Byte	1 word	1 word	1 word

#### Respond telegram

Slave address	Function code	Number of read Bytes	Data 1 <sup>st</sup> Byte	Data 2 <sup>nd</sup> Byte	...	Check sum CRC/LRC
1 Byte	1 Byte	1 Byte	1 Byte	1 Byte max. 250 Byte		1 word

**Read n Words**      This function enables the reading from the slave in words.  
**03h, 04h**

Command telegram

Slave address	Function code	Address 1. Bit	Number of Words	Check sum CRC/LRC
1 Byte	1 Byte	1 word	1 word	1 word

Respond telegram

Slave address	Function code	Number of read Bytes	Data 1 <sup>st</sup> word	Data 2 <sup>nd</sup> word	...	Check sum CRC/LRC
1 Byte	1 Byte	1 Byte	1 word	1 word		1 word
				max. 125 words		

**Write one Bit**      This function allows you to change one bit in the slave. A status change is  
**05h**      via "Status Bit" with following values:

"Status Bit" = 0000h → Bit = 0

"Status Bit" = FF00h → Bit = 1

Command telegram

Slave address	Function code	Address Bit	Status Bit	Check sum CRC/LRC
1 Byte	1 Byte	1 word	1 word	1 word

Respond telegram

Slave address	Function code	Address Bit	Status Bit	Check sum CRC/LRC
1 Byte	1 Byte	1 word	1 word	1 word

**Write one word**      This function sends one word to the slave. This allows you to overwrite a  
**06h**      register in the slave.

Command telegram

Slave address	Function code	Address word	Value word	Check sum CRC/LRC
1 Byte	1 Byte	1 word	1 word	1 word

Respond telegram

Slave address	Function code	Address word	Value word	Check sum CRC/LRC
1 Byte	1 Byte	1 word	1 word	1 word

**Write n Bits 0Fh** This function writes n Bits to the slave. Please regard that the number of Bits has additionally to be set in Byte.

## Command telegram

Slave address	Function code	Address 1. Bit	Number of Bits	Number of Bytes	Data 1 <sup>st</sup> Byte	Data 2 <sup>nd</sup> Byte	...	Check sum CRC/LRC
1 Byte	1 Byte	1 word	1 word	1 Byte	1 Byte	1 Byte	1 Byte	1 word
					max. 250 Byte			

## Respond telegram

Slave address	Function code	Address 1. Bit	Number of Bits	Check sum CRC/LRC
1 Byte	1 Byte	1 word	1 word	1 word

**Write n Words 10h** This function sends n words to the slave.

## Command telegram

Slave address	Function code	Address 1 <sup>st</sup> word	Number of words	Number of Bytes	Data 1 <sup>st</sup> word	Data 2 <sup>nd</sup> word	...	Check sum CRC/LRC
1 Byte	1 Byte	1 word	1 word	1 Byte	1 word	1 word	1 word	1 word
					max. 125 words			

## Respond telegram

Slave address	Function code	Address 1. word	Number of words	Check sum CRC/LRC
1 Byte	1 Byte	1 word	1 word	1 word

## Example for the deployment under Modbus

### Outline

After the installation you may use this sample project for your first steps.

When needed, you may obtain the sample project from VIPA.

During this example, a communication between a master and a slave via Modbus is established. Further we will show how to get easily control over the communication tasks by using the handler blocks.

### Requirements

The following components are required for the example:

1 CPU 21x with CP 240 Modbus as master system

Configuration tool STEP<sup>®</sup>7 Manager from Siemens with transfer cable

1 Modbus slave (like CPU 21x and CP 240 Modbus)

Modbus cable connection

### Approach

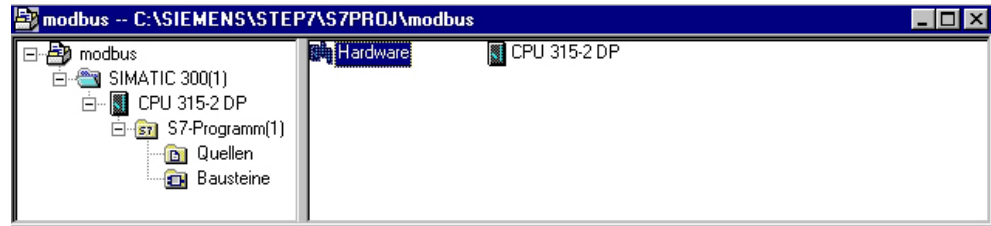
- Assemble a Modbus system, existing of master system, slave system and Modbus cable.
- Engineer the master side! For this you open the sample project using your configuration tool. Adjust the transfer parameters accordingly. Under *Protocol* you select "Modbus Master RTU". Edit the OB1 and coordinate the module addresses with the addresses of the parameterization. Transfer your project into the CPU 21x.
- Engineer the slave side. For this you open the sample project using your configuration tool. Adjust the parameters of the CP 240 accordingly. Under *Protocol* you select "Modbus Slave RTU". Type a slave address in *Address*. For the communication under Modbus, the slave doesn't need a PLC application, because source and destination are transferred by the master.

### Start project

To start the sample in your configuration tool, execute the following steps:

- Start the STEP<sup>®</sup>7 Manager from Siemens
- To extract "Modbus.zip", choose **File** > *dearchive*
- Select the sample file "Modbus.zip" and extract it to destination "s7proj"
- Open the project

**Project structure** The project has the following structure:



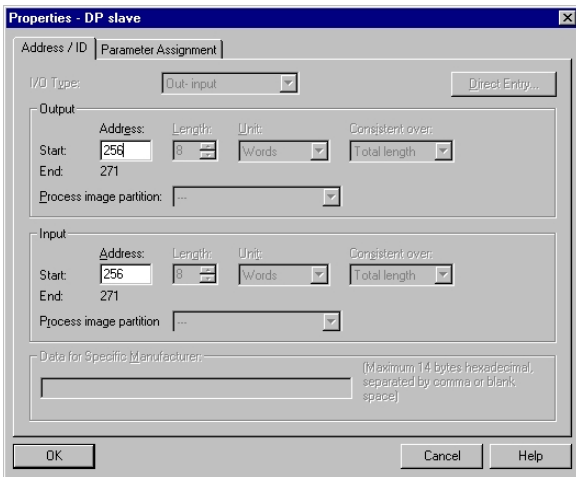
**Master project engineering**

The example already contains PLC application and the parameters for the Modbus master. You only have to adjust the Modbus parameters.

**Parameterization**

Start the hardware configurator from Siemens and choose the module 240-1CA10. Via double-click you reach the parameterization:

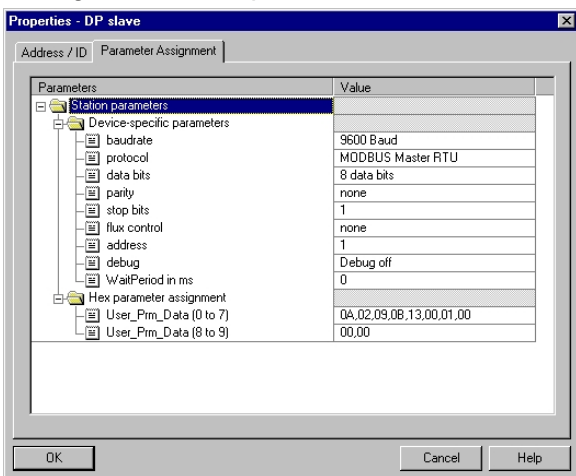
**Dialog for address input**



Here you may fix the start address for the 16 input and output Byte in the CPU.

Please regard, that you have to change addresses not only here but also in the according SEND and RECEIVE blocks.

**Dialog for Modbus parameters**



In this part of the parameterization you fix the Modbus parameters.

The following parameters have to be identical for all bus participants: baudrate, data bits, parity, stop bits and flow control.

Under *Protocol* choose "Modbus Master RTU".

Only the slave side needs an address. At the master parameterization the address is ignored.

**PLC application**      The wanted Modbus commands are defined via your PLC application. The recent example shows the deployment of SEND and RECEIVE in the OB1.

OB 1:

```

CALL "SEND_ACII_STX_3964"
  ADR      :=256           //Start address of the module
  _DB      :=DB10         //In this data block you create the telegram to send
  ABD      :=W#16#0       //From this Byte offset the telegram starts in _DB
  ANZ      :=MW12         //Telegram length (length to send) in Byte
  PAFE     :=MB14         //Error byte
  FRG      :=M1.0        //Send initialization (1=start, switches
                          //to 0 at end of sending)
  GESE     :=MW16         //internal value
  ANZ_INT  :=MW18         //internal value
  ENDE_KOM :=M2.0        //internal value
  LETZTER_BLOCK:=M2.1    //internal value
  SENDEN_LAEUFT:=M2.2    //internal value
  FEHLER_KOM :=M2.3      //internal value

CALL "RECEIVE_ACII_STX_3964"
  ADR      :=256           //Start address of the module
  _DB      :=DB11         //In this data block the
                          //received telegram is stored
  ABD      :=W#16#0       //From this Byte offset the telegram starts in _DB
  ANZ      :=MW22         //Telegram length (received length) in Byte
  EMFR     :=M1.1        //Receive state (1=Telegram received completely)
  PAFE     :=MB34         //Error byte
  GEEM     :=MW36         //internal value
  ANZ_INT  :=MW38         //internal value
  EMPF_LAEUFT :=M3.0     //internal value
  LETZTER_BLOCK:=M3.1    //internal value
  FEHL_EMPF :=M3.2      //internal value

U    M    1.1           //As long as receive state=1, no new telegram is send
R    M    1.1           //thus the receive state has to be acknowledged with 0

```

Now adjust the addresses that the CP occupies in the CPU if needed in the parameterization and transfer the hardware configuration into your CPU 21x of the master system.

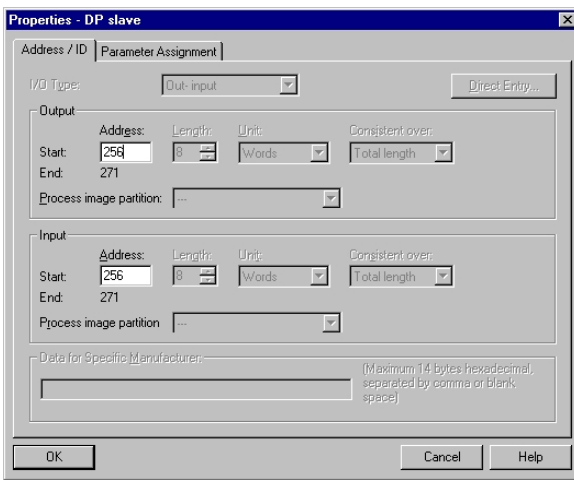
## Slave project engineering

For the project engineering of the slave you only have to adjust the Modbus parameters. A PLC application is not required because the source and destination information is supplied in the master telegram.

## Parameterization

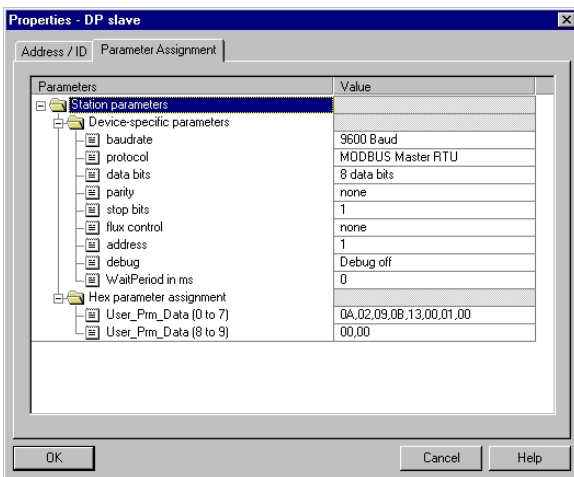
To parameterize the slave module, open the sample project in your hardware configurator. Select the module 240-1CA10. Via double-click you reach the parameterization.

### Dialog for address input



Here you may fix from which address on the 16Byte for in- and output data are stored in the CPU.

### Dialog for Modbus parameters



In this part of the parameterization you adjust the Modbus parameters.

The following parameters have to be identical for all bus participants: baudrate, data bits, parity, stop bits and flow control.

Under *Address* type a valid Modbus address for the slave.

Transfer the parameterization into your CPU of the slave system.



**Send and receive telegrams**

Open the variable table **Table1** of the sample project and go online.

	Operand	Anzei	Statuswert	Steuerwert
1	PEW 256	HEX	W#16#0000	
2	PEW 258	HEX	W#16#0000	
3	MW 12	DEZ	23	
4	M 1.0	BOOL	false	true
5	MB 2	BIN	2#0000_0000	2#0000_0000
6	MW 22	DEZ	6	
7				
8	DB10.DBD 0	HEX	DW#16#05100000	DW#16#05100000
9	DB10.DBD 4	HEX	DW#16#000810A0	DW#16#000810A0
10	DB10.DBD 8	HEX	DW#16#A1A2A3A4	DW#16#A1A2A3A4
11	DB10.DBD 12	HEX	DW#16#A5A6A7A8	DW#16#A5A6A7A8
12	DB10.DBD 16	HEX	DW#16#A9AABAC	DW#16#A9AABAC
13	DB10.DBD 20	HEX	DW#16#ADAEAF00	DW#16#ADAEAF00
14				
15	DB11.DBD 0	HEX	DW#16#05100000	DW#16#00000000
16	DB11.DBD 4	HEX	DW#16#000810A0	DW#16#00000000
17	DB11.DBD 8	HEX	DW#16#00000000	DW#16#00000000
18	DB11.DBD 12	HEX	DW#16#00000000	DW#16#00000000
19	DB11.DBD 16	HEX	DW#16#00000000	DW#16#00000000
20				

**Send block DB10**

DB10.DBD 0	DW#16#05100000 with 05 → 10 → 0000 →	<b>Command telegram</b> slave address 05h function code 10h (write n words) Offset 0000h
DB10.DBD 4	DW#16#000810A0 with 0008 → 10 → A0 →	<b>Command telegram + 1 data byte</b> Word count 0008h Byte count 10h Start 16Byte data with A0h
DB10.DBD 8	DW#16#A1A2A3A4	<b>data byte 2 ... 5</b>
DB10.DBD 12	DW#16#A5A6A7A8	<b>data byte 6 ... 9</b>
DB10.DBD 16	DW#16#A9AABAC	<b>data byte 10 ... 13</b>
DB10.DBD 20	DW#16#ADAEAF00 with ADAEAF → 00 →	<b>data byte 14 ... 16 + 1Byte not used</b> data byte 14 ... 16 not used by the module

**Receive block DB11**

DB11.DBD 0	DW#16#05100000 with 05 → 10 → 0000 →	<b>Respond telegram</b> slave address 05h function code 10h (no error) Offset 0000h
DB11.DBD 4	DW#16#000810A0 with 0008 → 10 → 00 →	<b>Respond telegram + 1 Data byte</b> Word count 0008h Byte count 10h Start 16Byte data with 00h (irrelevant for write command)
DB11.DBD 8	DW#16#00000000	<b>data byte 2 ... 5</b>
DB11.DBD 12	DW#16#00000000	<b>data byte 6 ... 9</b>
DB11.DBD 16	DW#16#00000000	<b>data byte 10 ... 13</b>
DB11.DBD 20	DW#16#00000000 with ADAEAF → 00 →	<b>data byte 14 ... 16 + 1 Byte not used</b> data byte 14 ... 16 not used by the module

**Receive block with error respond**

The communication under Modbus knows 2 kinds of error:

- Slave doesn't respond to a master command

When the slave is not reacting within the defined timeout period, the master writes the following error message into the receive block:

ERROR01 NO\_DATA. In hex notation the following values are shown:

DB11.DBD 0	<b>DW#16#4552524F</b> with 45 → 52 → 52 → 4F →	<b>Respond telegram</b> 45h: E 52h: R 52h: R → 4Fh: O
DB11.DBD 4	<b>DW#16#52000120</b> with 52 → 0001 → 20 →	<b>Respond telegram</b> 52h: R 0001h:1 (as word) 52h: " "
DB11.DBD 8	<b>DW#16#4E4F2044</b> with 4E → 4F → 20 → 44 →	<b>Respond telegram</b> 45h: N 52h: O 52h: " " 4Fh: D
DB11.DBD 12	<b>DW#16#41544100</b> with 41 → 54 → 41 → 00 →	<b>Respond telegram</b> 45h: A 52h: T 52h: A 00h: empty

•  
•  
•

- Slave answers with an error message

If the slave replies an error, the function code with 80h is send back marked with an OR.

DB11.DBD 0	<b>DW#16#05900000</b> with 05 → 90 → 0000 →	<b>Respond telegram</b> slave address 05h function code 90h (error message because 10h OR 80h = 90h) rest data is irrelevant because an error was announced
------------	--	--

## Communication by means of standard handler blocks

Data communication is controlled by means of the handler blocks that are supplied with the hardware.



### Note!

Modules with firmware revision level V1.06 or higher require that the handler block SYNCHRON is executed in the respective program. It is not possible to communicate with the CP 240 before the handler block has been executed since it enters a SYNCHRON identifier into the write and the read pointer, which must be acknowledged by the CPU.

Under Modbus the SYNCHRON block is not necessary.

The following blocks are supplied:

#### for CPU 24x

Name	FBs	Short description
SCP240	FB3	Send block for ASCII, STX/ETX, 3964(R) and Modbus
RCP240	FB4	Receive block for ASCII, STX/ETX, 3964(R) and Modbus
FETCH	FB20	Fetch block only for RK512
SEND	FB22	Send only for RK512
S/R_ALL	FB23	Send - Receive all only for RK512
SYNCHRON	FB25	Synchronization of the CP 240 from firmware rev. V1.06 (not req. For Modbus)

#### for CPU 21x

Name	FCs	Short description
SEND_ASCII_STX_3964	FC0	Send block for ASCII, STX/ETX, 3964(R) and Modbus
RECEIVE_ASCII_3964	FC1	Receive block for ASCII, STX/ETX, 3964(R) and Modbus
FETCH_RK512	FC2	Fetch block only for RK512
SEND_RK512	FC3	Send only for RK512
S/R_ALL_RK512	FC4	Send - Receive all only for RK512
SYNCHRON_RESET	FC9	Synchronization of the CP 240 from firmware rev. V1.06 (not req. for Modbus)

## Standard handler blocks for the CPU 24x

### ASCII, STX/ETX or 3964(R) communication

This type of communication procedure is always active, i.e. both partners must handle data transmission and reception in active mode. When data is transmitted neither the destination (transmission) nor the source (reception) are transferred.

Valid commands under ASCII, STX/ETX or 3964(R) are:

SCP240	FB3	(data transmission)
RCP240	FB4	(data reception)

### RK512 communication

The linkage procedure RK512 employs a master slave scheme. The master can request data from the slave by means of a FETCH and it can transfer data to the partner by means of a SEND. In this case the message header contains the destination or the source.

To enable the slave to react to jobs issued by the master, a S/R\_ALL (Send/Receive\_All) must be executed cyclically within the slave.

Valid RK512 commands are:

FETCH	FB20	(request data)
SEND	FB22	(transmit data)
S/R_ALL	FB23	(slave reaction to requests)

### Modbus communication

Depending on the chosen operating mode, the Modbus CP may be deployed as master or as slave.

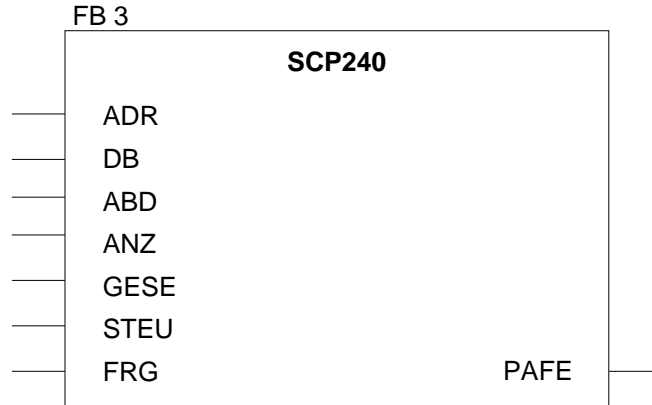
In slave mode no handler blocks are required. In master mode the communication with the slave takes place via SEND and RECEIVE commands. In both cases the CP occupies 16Byte in the CPU for input and output.

Valid commands under Modbus are:

SCP240	FB3	(send data)
RCP240	FB4	(receive data)

**SCP240  
SEND (FB3)**

This FB is used to transfer data to a peripheral unit in ASCII, STX/ETX, 3964(R) and Modbus.



- ADR** Peripheral address for access to the CP 240 module. You specify the peripheral address for the CP 240 system when you configure the DB1. This address must be located in the range from PY000 ... PY240. For details please refer to the VIPA CPU 24x Manual HB99.
- DB** The number of the data block that contains the transmit data.
- ABD** Word variable containing the number of the data word where the transmit data starts.
- ANZ** Word variable containing the number of bytes that will be transmitted.
- GESE** This is an internal variable that controls the transmission of data. Here you specify a flag word that can be used by the handler block to store internal data.

**STEU** Used by the handler block to store internal control bits. You specify a flag byte where the handler block can store its control bits.

**FRG** When you set this flag to "1", the data quantity specified in ANZ is transmitted once. Upon completion of the transmission, the bit is set to "0". The block is not executed if this bit contains a "0" when FB 3 is accessed!

**PAFE** The bits in this flag byte are set to "0" when a function completes without error. If an error occurs, an error code is entered here. This error code is cleared to "0" automatically when the error cause is removed.

The following errors may occur:

1 = Data block does not exist

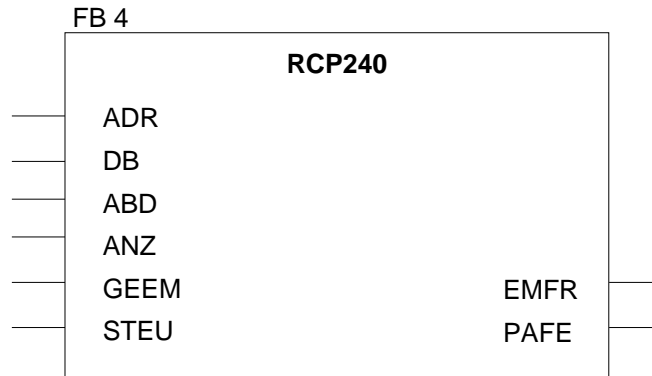
2 = Data block too short

3 = Data block number not within the valid range

---

**RCP240  
RECEIVE (FB4)**

This FB is used to receive data from a peripheral unit in ASCII, STX/ETX, 3964(R) and Modbus mode.



**ADR** The peripheral address to access the CP 240 module. You specify the peripheral address of the CP 240 system when you configure DB1. This address is located in the range from PY000 ... PY240.

For details please refer to the VIPA CPU 24x Manual HB99.

**DB** The number of the data module where the receive data is stored.

**ABD** Word variable containing the number of the data word from where the receive data should be saved.

**ANZ** Word variable containing the number of bytes that will be transferred.

**GEEM** This is an internal variable that controls the reception of data. Here you specify a flag word that can be used by the handler block to store internal data.

**STEU** Used by the handler block to store internal control bits. You specify a flag byte where the handler block can store its control bits.

**EMFR** The flag bit EMFR (receive complete) is set when a message has been received completely and when it has been saved in the RECEIVE-DB. This bit is not reset automatically.

**PAFE** All the bits in this flag byte are set to "0" when a function completes without error. If an error occurs, an error code is entered here. This error code is cleared to "0" automatically when the error cause is removed.

The following errors can occur:

1 = Data block does not exist

2 = Data block too short

3 = Data block number not within the valid range



### FETCH request data via RK512 (FB20)

This FB is used by a peripheral unit to request data using RK512. This FB is only valid in conjunction with RK512.



#### Note!

To enable the slave to react to requests issued by the master, a S/R\_ALL (Send/Receive\_All) must be executed cyclically within the slave.

**ADR** Peripheral address for accessing the CP 240 module. You specify the peripheral address for the CP 240 system when you configure the DB1. This address is located in the range from PY000 ... PY240. For details please refer to the VIPA CPU 24x Manual HB99.

**QDB** The number of the data block that contains the transmit data.

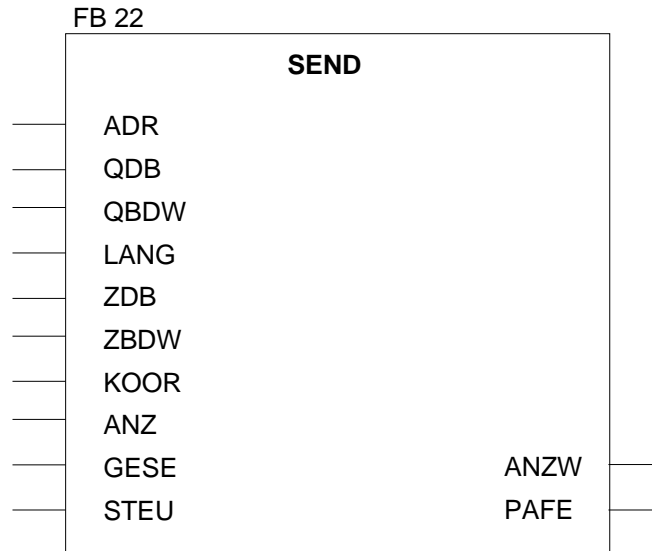
**QBDW** Word variable containing the number of the data word where the transmit data starts.

**LANG** Word variable containing the number of bytes that will be transferred.

<b>ZDB</b>	The number of the data block where the transmitted data will be stored.
<b>ZBDW</b>	Word variable containing the number of the data word, from where on the received data should be saved.
<b>KOOR</b>	<p>This provides the configuration for the use of a coordination flag. The High Byte contains the byte number, the Low Byte the bit number of the coordination flag. If the coordination flag should not be used, both the High Byte and the Low Byte must be set to "255".</p> <p>The coordination flag controls the access to the source area:</p> <p>This flag protects your transmitted data in the partner PLC from being overwritten. When the flag has been reset, the data may be overwritten again.</p>
<b>ANZW</b>	Indicator word. The indicator word occupies a flag word. Status bits are stored in the right-hand byte. When the right-hand byte contains the flag "ready with error", the left-hand byte contains an error number.
<b>STEU</b>	Used by the handler block to store internal control bits. You specify a flag byte where the handler block can store its control bits.
<b>ANZ</b>	Word variable containing the number of bytes that will be transferred.
<b>GESE</b>	This is an internal variable that controls the transmission of data. Here you must specify a flag word that can be used by the handler block to store internal data.
<b>PAFE</b>	<p>All the bits in this flag byte are set to "0" when a function completes without errors. If an error occurs, an error code is entered here. This error code is cleared to "0" automatically when the error cause is removed.</p> <p>The following errors can occur:</p> <ul style="list-style-type: none"><li>1 = Data block does not exist</li><li>2 = Data block too short</li><li>3 = Data block number not within the valid range</li></ul>

**SEND**  
**send data**  
**via RK512 (FB22)**

This FB is used for data output to a peripheral device by means of RK512.  
 This FB is only valid under RK512.



**Note!**

In order to enable the slave to react on requests from the master, the slave must execute a S/R\_ALL (Send/Receive\_All) in every cycle.

**ADR** Peripheral address for accessing the CP 240 module. You specify the peripheral address for the CP 240 system when you configure the DB1. This address is located in the range from PY000 ... PY240. For details please refer to the VIPA CPU 24x Manual HB99.

**QDB** The number of the data block that contains the transmit data.

**QBDW** Word variable containing the number of the data word where the transmit data starts.

**LANG** Word variable containing the number of bytes that will be transferred.

**ZDB** The number of the data block where the transmitted data will be stored.

**ZBDW** Word variable containing the number of the data word from where on the received data should be saved.

**KOOR** This provides the configuration of a coordination flag. The High Byte contains the byte number, the Low Byte the bit number of the coordination flag. If the coordination flag should not be used, both, the High Byte and the Low Byte must be set to "255".

The coordination flag controls access to the source area:

When the flag is set, your transmit data in the partner PLC is protected from being overwritten. When the flag is reset, the data may again be overwritten.

**STEU** Used by the handler block to store internal control bits. You specify a flag byte where the handler block can store its control bits.

**PAFE** All the bits in this flag byte are set to "0" when a function completes without errors. If an error occurs, an error code is entered here. This error code is cleared to "0" automatically when the error cause is removed.

The following errors can occur:

1 = Data block does not exist

2 = Data block too short

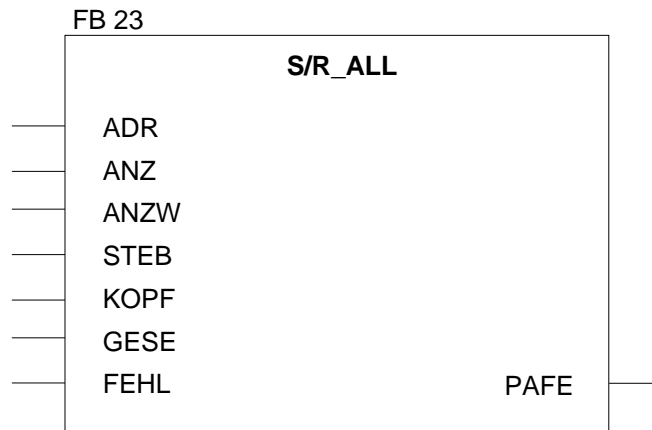
3 = Data block number not within the valid range

---

**S/R\_ALL**  
**reaction to master**  
**request**  
**via RK512 (FB23)**

When the system containing the CP 240 is used as slave, this FB must be executed by the slave CPU cyclically. The CP 240 can only react to the requests issued by the master if this is true. When a FETCH is received, the data is transferred to the master. Data that was received from the master by means of a SEND is accepted and stored, followed by an acknowledgment.

This FB is only valid for the RK512.



- ADR** Peripheral address for accessing the CP 240 module. You specify the peripheral address for the CP 240 system when you configure the DB1. This address is located in the range from PY000 ... PY240. For details please refer to the VIPA CPU 24x Manual HB99.
- ANZ** Word variable containing the number of bytes that will be transferred.
- ANZW** Indicator word. The indicator word occupies a flag word. Status bits are stored in the right-hand byte. When the right-hand byte contains the flag "ready with error", the left-hand byte contains an error number.
- STEB** Internal control byte.
- KOPF** The start of the 10Byte flag area where the RK512 message header is stored.

**GESE** This is an internal variable that controls the transmission of data. Here you specify a flag word that can be used by the handler block to store internal data.

**FEHL** Internal control byte.

**PAFE** All the bits in this flag byte are set to "0" when a function completes without errors. If an error occurs, the respective error code is stored in this location. This error code is cleared to "0" automatically when the error cause is removed.

The following errors can occur:

1 = Data block does not exist

2 = Data block too short

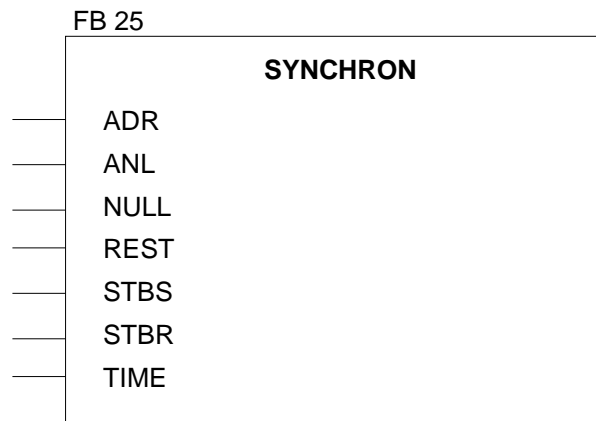
3 = Data block number not within the valid range

---

**SYNCHRON**  
**Synchronization**  
**(FB25)**

This block must be executed in the cyclic portion of the program. This function acknowledges the start-up flag of the CP 240 and is in this way synchronizing CPU and CP. Furthermore it is possible to reset the CP when communication should be interrupted, to ensure that the start-up procedure is synchronized properly.

The SYNCHRON block is not required for Modbus.



**ADR** Peripheral address for accessing the CP 240 module. You specify the peripheral address for the CP 240 system when you configure the DB1. This address is located in the range from PY000 ... PY240. For details please refer to the VIPA CPU 24x Manual HB99.

**ANL** The start-up was completed.  
 This bit informs the HTB that the CPU has executed a STOP/START or POWER-OFF/POWER-ON and that the synchronization is required.  
 The bit is cleared by the HTB when the synchronization procedure has completed.

**NULL** A bit that is used internally for data exchanges with the CP.

**REST** Reset of the CP 240.  
 The CP 240 is reset when the user sets this bit in the PLC program. When this bit is set, the handler block enters the reset flag in the CP and waits until this is acknowledged. The process continues as with the start-up.

**STBS** Control bit SEND  
Here you specify the flag byte where the control bits for the SEND-FB are saved.

**STBR** Control bit RECEIVE  
Here you specify the flag byte where the control bits for the RECEIVE-FB are saved.

**TIME** Timer for the delay time until the reset has been acknowledged.



**Sample FB25  
SYNCHRON:**

In OB21 and OB22, you have to set the bit that was specified in descriptor ANL to "1". When the block detects that a start-up has occurred, it will acknowledge the SYNCHRON flag and clear the control bits of the handler blocks for SEND and RECEIVE resp. FETCH and WRITE. When the synchronization has completed, the block will reset the ANL bit to "0".

FB25 may also be used to reset the CP 240. For this purpose the bit specified for the label REST has to be set. The result is that the FB issues the reset flag to the CP 240 and waits until this is acknowledged.

**Parameter  
description FB25**

```

BAUSTEIN#FB25
BSTNAME #SYNCHRON
BIB      #102
BEZ      #ADR   D:KF   MODULE ADDRESS
BEZ      #ANL   E:BI   START-UP WAS EXECUTED
BEZ      #NULL  E:BI   WRITE 0 INTO SZ/LZ
BEZ      #REST  E:BI   ISSUE RESET
BEZ      #STBS  E:BY   CONTROL BITS FOR SEND
BEZ      #STBR  E:BY   CONTROL BITS FOR RECEIVE
BEZ      #TIME  T      DELAY TIME FOR START-UP ACKN.

```

**Programming  
example**

```

BAUSTEIN#OB21
BIB      #6100
00000    :
00002    :UN  M 101.0   Start-up bit is 0
00004    :S   M 101.0   set it to 1
00006    :U   M 101.1   Send zero is 1
00008    :R   M 101.1   and reset it
0000A    :
0000C    :
0000E    :BE

```

```

BAUSTEIN#OB22
BIB      #6100
00000    :
00002    :UN  M 101.0   Start-up bit is 0
00004    :S   M 101.0   set it to 1
00006    :U   M 101.1   Send zero is 1
00008    :R   M 101.1   and reset it
0000A    :
0000C    :
0000E    :BE

```

```

BAUSTEIN#FB223 (cyclic processing FB)
BSTNAME #P3964
BIB      #17100
0000A    :
0000C    :SPA FB 25
          NAME #SYNCRON
          ADR  =KF +128           Module address
          ANL  =M 101.0           Start-up was executed
          NULL =M 101.1           Send zero to CP (used
                                internally)
          REST =M 101.2           Reset the CP
          STBS =MB 107            Control bits for Send-FB
          STBR =MB 109            Control bits for Receive-FB
          TIME =T 19              Wait time acknowledgment
0001E    :
00020    :
00022    :U   M 101.0           While Synchron was not
                                acknowledged
00024    :BEB                    End of program
00026    :                       After Synchron it is possible
                                to communicate
00028    :L   KB 0
0002A    :T   MW 102
0002C    :
0002E    :SPA FB 3
          NAME #SCP240
          ADR  =KF +128
          DB   =DB 10
          ABD  =MW 102
          ANZ  =MW 104
          GESE =MW 106
          STEU =MB 107
          FRG  =M 101.7
          PAFE =MB 108
00042    :
00044    :
00046    :SPA FB 4
          NAME #RCP240
          ADR  =KF +128
          DB   =DB 11
          ABD  =MW 102
          ANZ  =MW 114
          GEEM =MW 116
          STEU =MB 109
          EMFR =M 101.6
          PAFE =MB 110
0005A    :
0005C    :BE

```

## Standard handler blocks for the CPU 21x

### ASCII, STX/ETX or 3964(R) communication

This type of communication procedure is always active, i.e. both partners must handle data transmission and reception in active mode. When data is transmitted neither the destination (transmission) nor the source (reception) are transferred.

Valid commands under ASCII, STX/ETX or 3964(R) are:

SEND_ACII_STX_3964	FC0	(data transmission)
RECEIVE_ACII_STX_3964	FC1	(data reception)

### RK512 communication

The linkage procedure RK512 employs a master-slave scheme. The master can request data from the slave by means of a FETCH and it can transfer data to the partner by means of a SEND. In this case the message header contains the destination or the source.

To enable the slave to react to jobs issued by the master, a S/R\_ALL (Send/Receive\_All) must be executed cyclically within the slave.

Valid RK512 commands are:

FETCH_RK512	FC2	(request data)
SEND_RK512	FC3	(transmit data)
S/R_ALL_RK512	FC4	(slave reaction to requests)

### Modbus communication

Depending on the chosen operating mode, the Modbus CP may be deployed as master or as slave.

In slave mode no handler blocks are required. In master mode the communication with the slave takes place via SEND and RECEIVE commands. In both cases the CP occupies 16Byte in the CPU for input and output.

Valid commands under Modbus are:

SEND_ASCII_STX_3964	FC0	(send data)
RECEIVE_ASCII_STX_3964	FC1	(receive data)

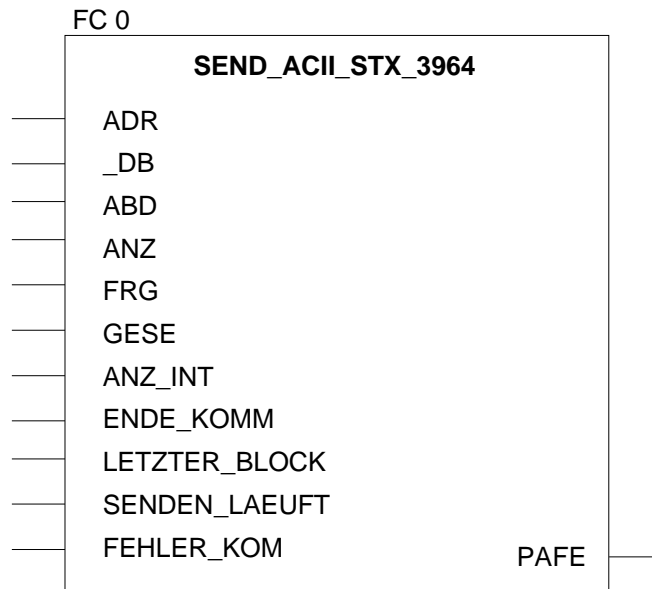
---

**SEND\_ASCII\_  
STX\_  
3964 (FC 0)**

This FC is used to transfer data to a peripheral unit in ASCII, STX/ETX, 3964(R) and Modbus.

The labels `_DB`, `ADB` and `ANZ` define the transmit slot.

The bit `FRG` initiates the transmission of the data. When all the data has been transmitted, the `FRG` bit is reset by the `HTB`.

**ADR**

Int: Peripheral address for accessing the CP 240 module. In your configuration tool you specify the peripheral address that will be used by the system to access the CP 240.

The address range is PY000 ... PY240.

**\_DB**

Block\_DB: the number of the data block that contains the transmit data.

**ABD**

Word: word variable that contains the number of the data word that contains the characters that will be transmitted.

**ANZ**

Word: word variable that contains the number of bytes that will be transmitted.

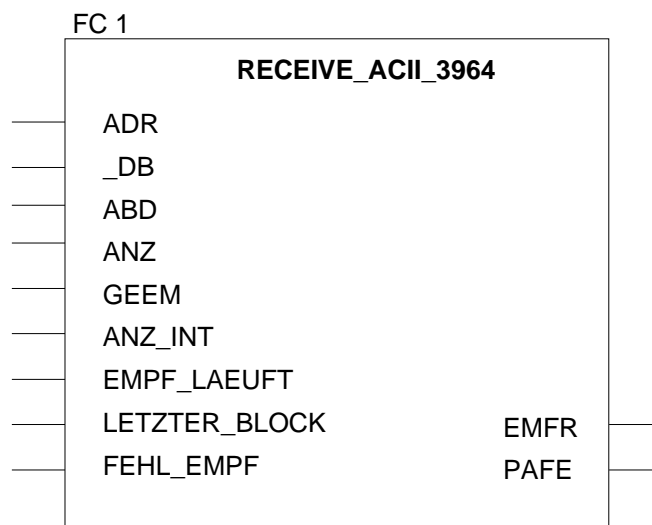
<b>FRG transmit enable</b>	Bool: when this flag bit is set to "1", the data quantity specified in ANZ is transmitted once. After transmission the bit is reset to "0". If this bit is already at "0" when the FC is accessed, the function is skipped immediately!
<b>GESE</b>	Word: quantity of data words that have already been transmitted.
<b>ANZ_INT</b>	Word: specifies the number of bytes to be transmitted.
<b>ENDE_KOM</b>	Bool: communication has been completed.
<b>LETZTER_BLOCK</b>	Bool: last block is transmitted.
<b>SENDEN_LAEUFT</b>	Bool: the data block is transmitted.
<b>FEHLER_KOM</b>	Bool: communication error.
<b>PAFE</b>	<p>Byte: all the bits in this flag byte are set to "0" when a function completes without errors. If an error occurs, the respective error code is stored in this location. This error code is cleared to "0" automatically when the error cause is removed.</p> <p>The following errors can occur:</p> <ul style="list-style-type: none"><li>1 = Data block does not exist</li><li>2 = Data block too short</li><li>3 = Data block number not within the valid range</li></ul>

**RECEIVE\_ASCII\_  
STX\_3964 (FC 1)**

This FC is provided for the purpose of receiving data from a peripheral device in ASCII, STX/ETX, 3964(R) and Modbus mode.

The labels `_DB` and `ADB` define the start of the receive slot.

When output `EMFR` is set, a new message has been retrieved completely. The length of the message is stored in `ANZ`. When the message has been analyzed, the user resets this bit. The PLC will not accept any new messages while the bit is "1". Depending on the number of buffers any received messages are saved by the module.

**ADR**

Int: peripheral address for accessing the CP 240 module. You specify the peripheral address that will be used by the system to access the CP 240 by means of your configuration tool.

The range of the address is PY000 ... PY240.

**\_DB**

Block\_DB: the number of the data module where the received data is stored.

**ABD**

Word: first data word of the receive slot.

**ANZ**

Word: word variable that contains the number of bytes that will be received.

<b>GEEM</b>	Word: the quantity of data that has already been received.
<b>ANZ_INT</b>	Word: length of received data in bytes.
<b>EMPF_LAEUFT</b>	Bool: reception is active.
<b>LETZTER_BLOCK</b>	Bool: the last block has been transmitted.
<b>FEHL_EMPF</b>	Bool: communication error
<b>PAFE</b>	<p>Byte: all the bits in this flag byte are set to "0" when a function completes without errors. If an error occurs, the respective error code is stored in this location. This error code is cleared to "0" automatically when the error cause is removed.</p> <p>The following errors can occur:</p> <ul style="list-style-type: none"><li>1 = Data block does not exist</li><li>2 = Data block too short</li><li>3 = Data block number not within the valid range</li></ul>

**FETCH\_RK512  
(FC 2)**

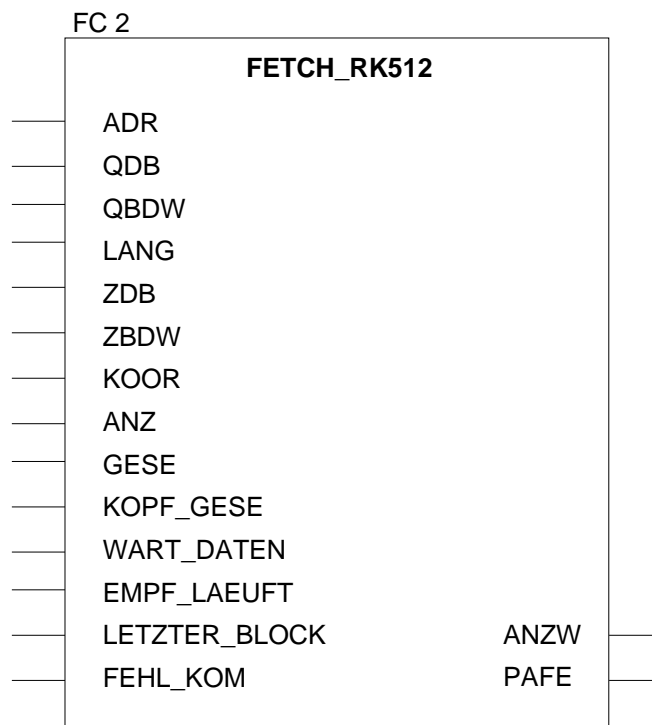
Request data by means of RK512

This FC is used to request data from a peripheral device using RK512. This FC can only be used in conjunction with RK512.

The purpose of the FETCH is to request the data from a communication partner. A message containing the source data is transmitted to the partner. The partner then compiles the data and returns them. Now the received data is saved to the specified destination. When the FC is issued, the labels QDB, QBDW, LANG define the source area and the labels ZDB and ZBDW the destination area.

When the FC is called, the control bits are used to check whether a job is present and active or not. When all the control bits are "0", a new FETCH command is initiated. For this purpose the message header is transferred to the CP and a delay is executed to receive the expected acknowledgment along with application data. As long as the partner has not sent the acknowledgment message, the indicator word contains "job active". Only when the CP has signaled the acknowledgment message to the PLC and when the application data has been transferred, the indicator word is set to "job completed" and the communication link to the CP is terminated. In case of communication errors, the CP returns an error number to the PLC. The respective error number is entered into the indicator word and the bit "job completed with errors" is set.

The cyclic portion of the program must process the function until "job completed - with/without - errors" is set in the indicator word.





**Note!**

In order to enable the slave to react to requests from the master, the slave must execute a S/R\_ALL (Send/Receive\_All) cyclically.

<b>ADR</b>	Int: Peripheral address for accessing the CP 240 module. You specify the peripheral address that will be used by the system to access to the CP 240 by means of your configuration tool. The range of the address is PY000 ... PY240.
<b>QDB</b>	Int: the number of the data block that contains the transmit data.
<b>QBDW</b>	Int: first data word in the source data block.
<b>LANG</b>	Int: length of data in words.
<b>ZDB</b>	Block_DB: destination data block The number of the data block where the transmitted data is stored.
<b>ZBDW</b>	Int: first data word in the destination data block.
<b>KOOR</b>	Word: coordination byte allocation: This provides the configuration of the coordination flag. The High Byte must contain the byte number and the Low Byte the bit number of the coordination flag. If the coordination flag should not be used, High Byte and Low Byte must be set to "255". The coordination flag controls access to the source area: When the flag is set, your transmit data in the partner PLC is protected from being overwritten. When the flag is reset, the data may be overwritten again.
<b>ANZ</b>	Word: number of bytes received (internal).
<b>GESE</b>	Word: number of bytes received (internal).

<b>KOPF_GESE</b>	Bool: header was transmitted to partner.
<b>WART_DATEN</b>	Bool: wait for data.
<b>EMPF_LAEUFT</b>	Bool: reception active.
<b>LETZTER_BLOCK</b>	Bool: last block was transmitted.
<b>FEHL_KOM</b>	Bool: a communication error has occurred.
<b>ANZW</b>	Word: Indicator word. The indicator word occupies a flag word. Status bits are stored in the right-hand byte. When the right-hand byte contains the flag "ready with error", the left-hand byte contains an error number.
<b>PAFE</b>	Byte: all the bits in this flag byte are set to "0" when a function completes without errors. If an error occurs, the respective error code is stored in this location. This error code is cleared to "0" automatically when the error cause is removed.  The following errors can occur: 1 = Data block does not exist 2 = Data block too short 3 = Data block number not within the valid range

**SEND\_RK512**  
**Transmit data**  
**via RK512 (FC 3)**

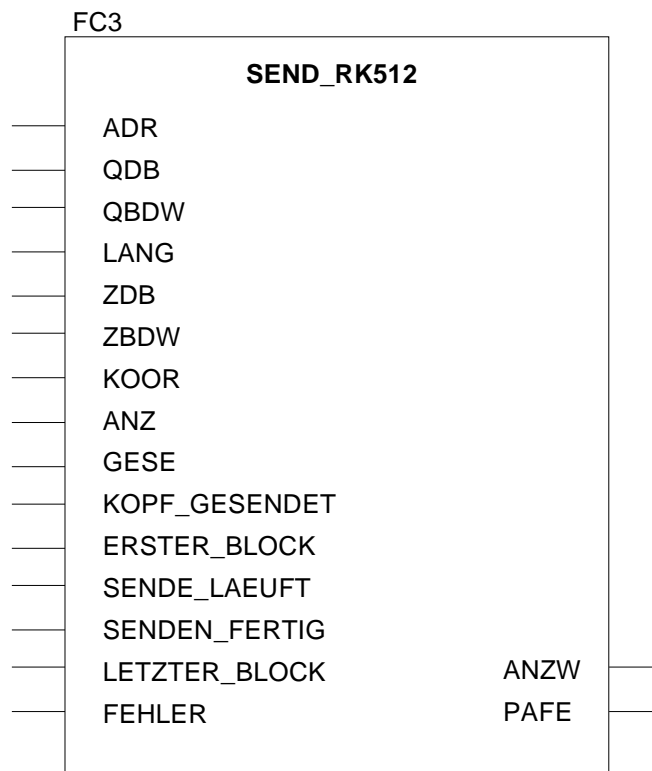
The purpose of this FC is to output data to a peripheral device via RK512. This FB is only valid with RK512.

The purpose of the SEND\_RK512 is to transmit data from a defined source area of the PLC to a partner and to instruct it where to deposit the received data. The source area is defined by means of the labels QDB, QBDW and LANG. The destination area of the partner is defined by means of the labels ZDB and ZBDW.

When the FC is accessed, the control bits are used to check whether a job is present and active or not. When all the control bits are "0", a new send job is initiated. For this purpose the message consisting of the header and the application data is transferred to the CP and the respective acknowledgment is expected.

While the partner has not sent the acknowledgment message, the indicator word contains "job active". Only when the CP has signaled the acknowledgment message to the PLC, the indicator word is changed to "job completed" and the communication link to the CP will be terminated. In case of communication errors, the CP returns an error number to the PLC. The respective error number is entered into the indicator word and the bit "job completed with errors" is set.

The communication session with the CP is terminated. The cyclic portion of the program must process the function as long as "job completed - with/without - errors" is set in the indicator word.



**Note!**

In order to enable the slave to react to requests from the master, the slave must execute a S/R\_ALL (Send/Receive\_All) cyclically.

<b>ADR</b>	Int: peripheral address for accessing the CP 240 module. You specify the peripheral address that will be used by the system to access to the CP 240 by means of your configuration tool. The range of the address is PY000 ... PY240.
<b>QDB</b>	Block_DB: the number of the data block that contains the transmit data.
<b>QBDW</b>	Int: first data word of the send slot.
<b>LANG</b>	Int: quantity of send data.
<b>ZDB</b>	Int: the number of the data block where the transmitted data is stored.
<b>ZBDW</b>	Int: first data word of the receive slot.
<b>KOOR</b>	Word: This provides the configuration of the coordination flag. The High Byte contains the byte number and the Low Byte the bit number of the coordination flag. If the coordination flag should not be used, both the High Byte and the Low Byte must be set to "255". The coordination flag controls access to the source area: When the flag is set, your transmit data in the partner PLC is protected from being overwritten. When the flag is reset, the data may be overwritten again.
<b>ANZ</b>	Word: number of bytes transmitted (internal).
<b>GESE</b>	Word: number of bytes transmitted (internal).
<b>KOPF_GESENET</b>	Bool: header was transmitted to partner.

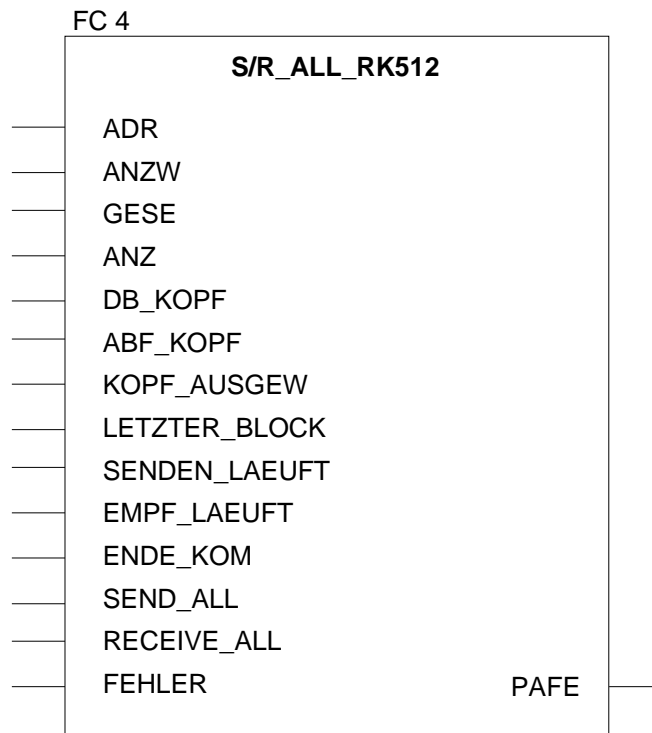
<b>ERSTER_BLOCK</b>	Bool: the first block of data was transmitted.
<b>SENDE_LAEUFT</b>	Bool: transmission is active.
<b>SENDEN_FERTIG</b>	Bool: data transmission completed.
<b>LETZTER_BLOCK</b>	Bool: last block was transmitted.
<b>FEHLER</b>	a communication error has occurred.
<b>ANZW</b>	Word: indicator word - the indicator word occupies a flag word. The right-hand byte is used to store status bits. The left-hand byte contains the error number if the right-hand byte contains the identifier "completed with error".
<b>PAFE</b>	Byte: All the bits in this flag byte are set to "0", when a function completes without errors. If an error occurs, the respective error code is stored in this location. This error code is cleared to "0" automatically when the error cause is removed.  The following errors can occur: 1 = Data block does not exist 2 = Data block too short 3 = Data block number not within the valid range

**S/R\_ALL\_RK512  
(FC 4)**

The purpose of this block is to process FETCH and SEND jobs from the partner.

When the system is used with the CP 240 as slave, the slave CPU has to access this FC cyclically. This is the only manner in which the CP 240 is able to react to the jobs issued by the master. After a FETCH, the data is collected and transmitted to the master. The data received from the master by a SEND is retrieved, saved and acknowledged.

This FB is only valid in conjunction with RK512.

**ADR**

Int: peripheral address for accessing the CP 240 module. You specify the peripheral address that will be used by the system to access to the CP 240 by means of your configuration tool.

The range of the address is PY000 ... PY240.

**ANZW**

Word: indicator word. The indicator word occupies a flag word. Status bits are stored in the right-hand byte. When the right-hand byte contains the flag "ready with error", the left-hand byte contains an error number.

**GESE**

Word: number of bytes received (internal).

<b>ANZ</b>	Word: number of bytes received (internal).
<b>ABF_KOPF</b>	Word: first data word in the receive/transmit slot.
<b>KOPF_AUSGEW</b>	Bool: the telegram header has been analyzed.
<b>LETZTER_BLOCK</b>	Bool: last block was transmitted.
<b>SENDEN_LAEUFT</b>	Bool: transmission is active.
<b>EMPF_LAEUFT</b>	Bool: data reception is active.
<b>ENDE_KOM</b>	Bool: the message was transmitted/received completely.
<b>SEND_ALL</b>	Bool: the block operates in SEND_ALL mode.
<b>RECEIV_ALL</b>	Bool: the block operates in RECEIVE_ALL mode.
<b>FEHLER</b>	Bool: a communication error has occurred.
<b>PAFE</b>	<p>Byte: all the bits in this flag byte are set to "0" when a function completes without errors. If an error occurs, the respective error code is stored in this location. This error code is cleared to "0" automatically when the error cause is removed.</p> <p>The following errors can occur:</p> <ul style="list-style-type: none"><li>1 = Data block does not exist</li><li>2 = Data block too short</li><li>3 = Data block number not within the valid range</li></ul>

---

**SYNCHRON\_**  
**RESET**  
**synchronization**  
**and reset (FC 9)**

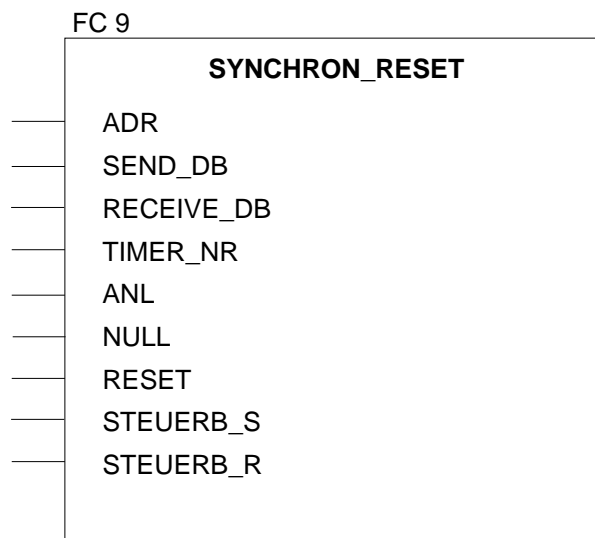
This block must be accessed from the cyclic portion of the program. This function acknowledges the start-up flag from the CP 240 to establish synchronism between the CPU and the CP. Furthermore it is possible to reset the CP when communication should be interrupted to ensure that the start-up procedure is synchronized properly.

**Start-up**

After a reboot or a re-start, the bit defined in label ANL has to be set to "1" for a single cycle to inhibit processing of the SEND-/RECEIVE blocks as long as this bit is not acknowledged by the function.

**Reset**

Setting the RESET bit, a flag is transferred to the CP that causes this to clear all buffers and pointers. When the CP has completed this action, it sets the SYNCHRON flag. When the CPU has acknowledged this flag, the system can continue communicating.

**ADR**

Int: peripheral address for accessing the CP 240 module. You specify the peripheral address that will be used by the system to access to the CP 240 by means of your configuration tool.

The range of the address is PY000 ... PY240.

**SEND\_DB**

Block\_DB: data block for the data handling over to the CP.

**RECEIVE\_DB**

Block\_DB: data block for the data fetch from the CP.

**TIMER\_NR**

Timer: number of the timer for the delay time.



<b>ANL</b>	<p>Bool: the start-up will be executed.</p> <p>This bit informs the HTB that the CPU has executed a STOP/START or POWER-OFF/POWER-ON and that synchronization is required.</p> <p>The bit is cleared by the HTB when the synchronization has been completed.</p>
<b>NULL</b>	<p>Bool: send zero to the CP (for internal purposes).</p>
<b>RESET</b>	<p>Bool: reset of the CP 240.</p> <p>The CP 240 is reset when the user sets this bit in the PLC program. When this bit is set, the handler block enters the RESET flag in the CP and waits until this is acknowledged. The process continues as with the start-up.</p>
<b>STUERB_S</b>	<p>Byte: control bit for SEND-FC and S/R_ALL-FC.</p> <p>Here you specify the flag byte where the control bits for the SEND-FC are saved.</p>
<b>STUERB_R</b>	<p>Byte: control bit for RECEIVE-FC and FETCH-FC.</p> <p>Here you specify the flag byte where the control bits for the RECEIVE-FC are saved.</p>

**Programming  
example**

Call the blocks in OB1

```

CALL "DPRD_DAT"           //Read data from modules
  LADDR :=W#16#100
  RET_VAL:=MW100
  RECORD :=P#DB11.DBX 0.0 BYTE 16

CALL FC 9                 //call Synchron
  ADR :=0                 //1. DW in SEND/EMPF_DB
  SEND_DB :=DB10         //Send_DB module
  RECEIVE_DB:=DB11       //Empfang_DB module
  TIMER_NR :=T2          //Delay time Synchron
  ANL :=M3.0             //Start-up completed
  NULL :=M3.1           //Intermediate flag
  RESET :=M3.2          //Execute module reset
  STEUERB_S :=MB2       //Control bits Sende_FC
  STEUERB_R :=MB1       //Control bits Receive_FC

U M 3.0                   //No SEND/RECEIVE processing
                           during start-up
SPB schr

CALL FC 1                 //Receive data
  ADR :=0                 //1. DW in SEND/EMPF_DB
  SEND_DB :=DB10         //Send_DB module
  EMPF_DB :=DB11        //Empfang_DB module
  _DB :=DB11            //Empfang_DB message
  ABD :=W#16#14         //1. DW receive buffer
                           (DW20)
  ANZ :=MW10            //Received data quantity
  EMFR :=M1.0          //Receive completed
  PAFE :=MB12          //Error byte
  GEEM :=MW100         //Data used internally
  ANZ_INT :=MW102      //Data used internally
  empf_laeuft :=M1.1   //Data used internally
  letzter_block:=M1.2  //Data used internally
  fehl_empf :=M1.3     //Data used internally

U M 1.0                   //Receive complete
R M 1.0                   //clear receive complete

```

```
CALL FC      0           //Send Data
ADR          :=0         //1. DW in SEND/EMPF_DB
SEND_DB     :=DB10      //Send_DB module
EMPF_DB     :=DB11      //Empfang_DB module
_DB         :=DB10      //Sende_DB message
ABD         :=W#16#14    //1. DW send buffer
                          (DW20)
ANZ         :=MW14      //Send data quantity
FRG         :=M2.0      //Specify send complete
PAFE        :=MB16      //Error byte
GESE        :=MW104     //Data used internally
ANZ_INT     :=MW106     //Data used internally
ende_kom    :=M2.1      //Data used internally
letzter_block:=M2.2     //Data used internally
senden_laeuft:=M2.3     //Data used internally
fehler_kom  :=M2.4      //Data used internally

schr: CALL "DPWR_DAT"    //Write data to module
      LADDR :=W#16#100
      RECORD :=P#DB10.DBX 0.0 BYTE 16
      RET_VAL:=MW102
```

Program in start-up OB100

```
UN    M    3.0
S     M    3.0           //CPU will start-up
```



**CP 240 with  
RS422/485  
interface**

Electrical data	VIPA 240-1CA00, VIPA 240-1CA10
Number of channels	1
Power supply	5V via backplane bus
Current consumption	max. 200mA
ext. power supply	-
Isolation	>= AC 500V, according to DIN 19258
Status indicator (LEDs)	via LED on the front
Connectors / interfaces	25pin D-type socket for RS422/RS485 ASCII transfer, 3964(R), 3964(R) with RK512, Modbus (VIPA 240-1CA10)
Data transfer rate -1CA00	150Baud up to 115kBaud
-1CA10	150Baud up to 38.4kBaud
Stop bits	1, 1.5, 2 (configurable)
Parity	none, even, odd (configurable)
Flow control	none, hardware, XON/XOFF
ZVZ	values from 0 to 5s
Programming data	
Input data	16Byte
Output data	16Byte
Parameter data	16Byte
Diagnostic data	4Byte (none for Modbus)
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	80 g



## Chapter 10 Counter modules

### Overview

This chapter contains information on the interfacing and configuration of the SSI-module FM 250 S.

The different operating modes and counting options are described for the counter module FM 250, i.e. the behavior of the counter when the different input signals are connected.

Below follows a description of:

- SSI module FM 250S
- Counter module FM 250
- Technical data

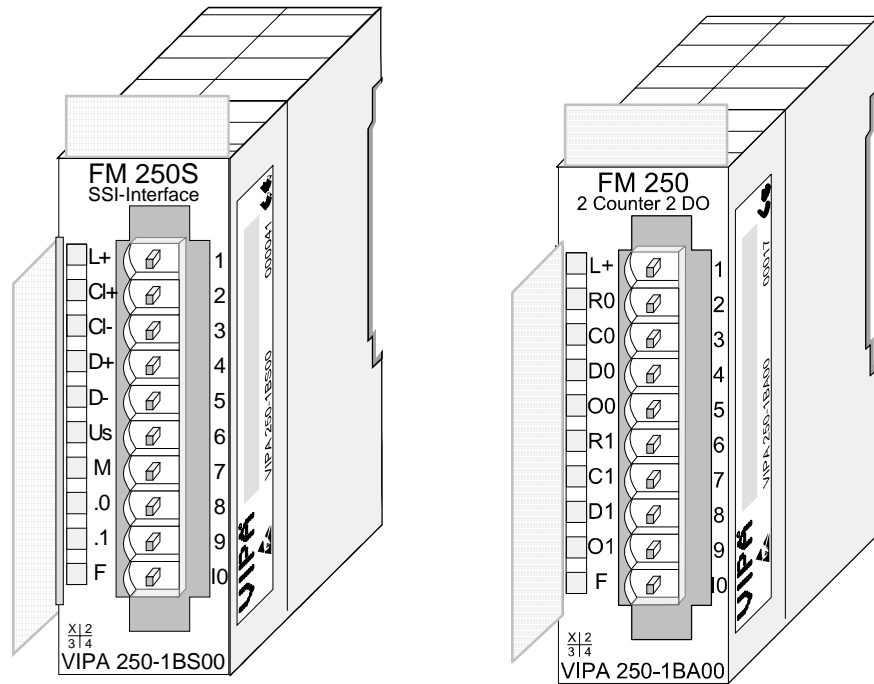
### Contents

Topic	Page
<b>Chapter 10 Counter modules .....</b>	<b>10-1</b>
System overview .....	10-2
FM 250S - SSI-Interface - Construction .....	10-3
FM 250 - Counter module - Construction .....	10-9
Summary of counter modes and interfacing .....	10-12
Counter modes .....	10-14
Technical data .....	10-58

## System overview

Here follows a summary of the measurement modules that are currently available from VIPA:

SSI-Interface FM 250S, counter module FM 250



### Order data

Type	Order number	Description
FM 250S	VIPA 250-1BS00	SSI-Interface
FM 250	VIPA 250-1BA00	Counter module (2 counter 2 DO)



## FM 250S - SSI-Interface - Construction

### Principles

The SSI interface is a synchronous serial interface. SSI is the abbreviation for **S**ynchronous **S**erial **I**nterface. The SSI module provides the connection for transducers with absolute coding and a SSI interface.

The module converts the serial information of the transducer into parallel information for the controller. Data can be transferred in gray or in binary code.

### Configurable outputs

The interface has connections for the SSI signals clock, data and the transducer supply voltage as well as two additional outputs that may be set or reset when a limit value is exceeded.

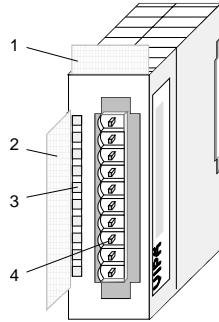
Output 0 can also be programmed as hold input. This causes the SSI transducer value to be frozen when a 24V high level is applied to output 0. A low level will cause the transducer to transmit the actual SSI values.

You can also configure the outputs that they will remain set if the BASP signal is active.

### Properties

- Wiring does not depend on the length of the data word. The interface always uses 4 wires.
- Maximum security due to the use of symmetrical clock and data signals.
- Secure data acquisition due to the use of single-step gray code (configurable).
- Galvanic isolation of receiver and encoder by means of opto couplers.
- 1 SSI channel
- Direct power supply to the SSI transducer via front-facing connector
- DC 24V power supply
- Baudrate selection between of 100kBaud and 600kBaud
- 2 configurable digital outputs, one may be used as hold input to freeze the current SSI transducer value
- Measured value available in gray or in binary code
- 4Byte of parameter data
- 4Byte of input data
- 4Byte of output data
- Configuration by means of control byte

**Construction**

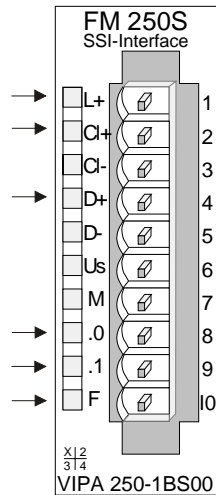


- [1] Label for module name
- [2] Label for bit address with description
- [3] LED status indicator
- [4] Edge connector

**Status indicator pin assignment**

**LED Description Pin Assignment**

- L+ LED (yellow)  
Supply voltage available
- Ci+ LED (green)  
Clock output
- D+ LED (green)  
Transducer data input
- .0 LED (green)  
Input/output 0
- .1 LED (green)  
Input/output 1
- F LED (red)  
Error /overload



- 1 Supply voltage DC +24V
- 2 CLK+ (Output)
- 3 CLK- (Output)
- 4 Data+ (Input)
- 5 Data- (Input)
- 6 DC 24V SSI transducer supply voltage
- 7 Common SSI transducer supply
- 8 Input/output .0 and hold input
- 9 Input/output .1
- 10 Common of supply voltage

**LEDs**

The SSI-Interface has a number of LEDs. The following table explains the significance of these LEDs:

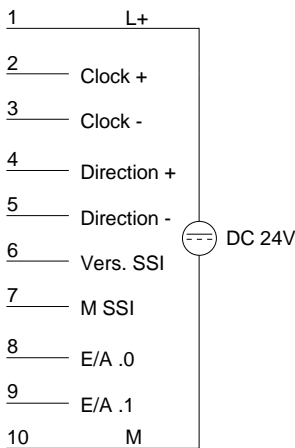
Name	Color	Description
L+	yellow	Indicates that 24V power is available
C+	green	ON when clock pulses are transmitted OFF when hold function has been activated and 24V at I/O .0
D+	green	ON when data is received from the transducer (wiring test)
.0	green	ON when 24V power is available at I/O .0
.1	green	ON when 24V power is available at I/O .1
F	red	ON when short circuit or overload is detected on one of the two I/O .0/.1

**Line distances**

The baudrate depends on the length of the communication line and on the SSI transducer. Wiring has to consist of screened twisted pair cables. The specifications below are only intended as a guideline.

- < 400m: → 100kBaud
- < 100m: → 300kBaud
- < 50m: → 600kBaud

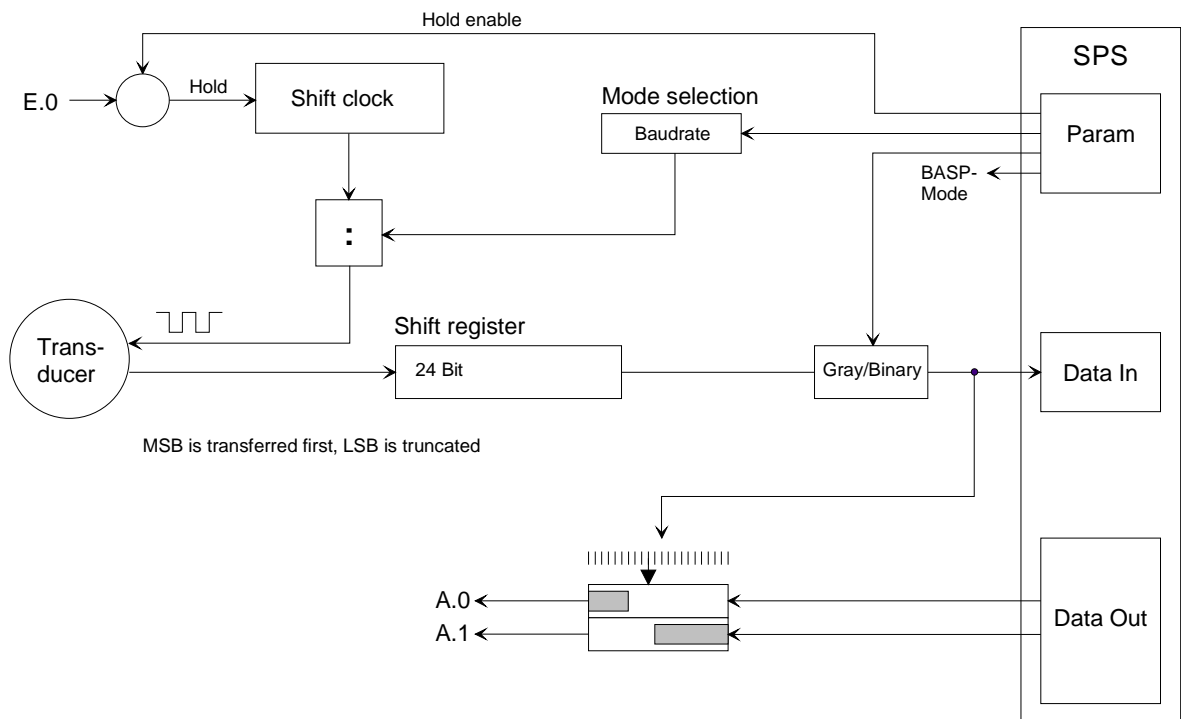
**Wiring diagram**



The SSI interface has an internal power supply. This power supply requires a voltage of DC 24V via L+ and M.

The supply voltage provides power to the interface electronics as well as the SSI transducer connected with DC 24V to pin 6 and 7.

**Block diagram**



**Configuration data** 4Byte of configuration data are transferred. In these bytes you define the baudrate, the coding and the analysis of the combined I/O .0 as well as the BASP signal.

The structure of the configuration data is as follows:

Byte	Bit 0 ... Bit 7
0	Bit 0 ... Bit 7: reserved
1	Bit 0 ... Bit 7: reserved
2	Baudrate 0: 300kBaud (default) 1: 100kBaud 2: 300kBaud 3: 600kBaud 4...255: 300kBaud
3	Bit 0: Coding 0: Binary code (default) 1: Gray code Bit 2: SSI format 0: Multiturn (24 bit) 1: Singleturn (12 bit) Bit 4: Hold function 0: deactivate 1: activate Bit 7: BASP signal 0: ignore 1: analyze

### Parameter

#### Baudrate

The transducer connected to the SSI channel transmits serial data. It requires a clock pulse from the SSI interface. The baudrate defines this clock. You may choose a value of 100, 300 or 600kBaud. The default setting is 300kBaud.

**Coding**

The gray code is a different form of binary code. The principle of the gray code is that two neighboring gray numbers will differ in exactly one single bit.

When the gray code is used, transmission errors can be detected easily as neighboring characters may only be different in a single location.

Table of rules for the gray code:

Decimal	Gray Code
0	0 0 0 0
1	0 0 0 1
2	0 0 1 1
3	0 0 1 0
4	0 1 1 0
5	0 1 1 1
6	0 1 0 1
7	0 1 0 0
8	1 1 0 0
9	1 1 0 1
10	1 1 1 1
11	1 1 1 0
12	1 0 1 0
13	1 0 1 1
14	1 0 0 1
15	1 0 0 0

i.e. the last digit of the number results from the vertical repetition of the sequence "0 11 0", the penultimate digit results from the repetition "00 1111 00", the third-last number from the repetition of 4x"0", 8x"1" and again 4x"0", etc. (see columns in the table!).

**Hold function**

Here you define that I/O .0 should be used as hold input. When you have activated this function, the current transducer value will be stored when I/O .0 is connected to 24V. The transducer value is only updated when the 24V level is removed from I/O .0.

In this case you have to be aware that I/O .0 operates only in input mode.

**BASP signal**

BASP is a German abbreviation for command output inhibited, i.e. all outputs are reset and inhibited as long as the BASP signal is applied via the backplane bus. You may disable the evaluation of the BASP signals by setting this bit. This means that the outputs will remain set.

**Access to the SSI Interface****Input data (Data In)**

The input data from the SSI transducer has a length of 4Byte. Byte 0 can be used as an I/O status indicator for the. Data is supplied in binary or in gray code, depending on the selected mode.

Byte	Data In
0	Bit 0: Status I/O .0. The status feedback only occurs when the output double-word for the respective I/O was preset! Bit 1: Status I/O .0. The status feedback only occurs when the output double-word for the respective I/O was preset! Bit 2-7: reserved
1	SSI transducer value: HB
2	SSI transducer value: MB
3	SSI transducer value: LB

**Output data (Data Out)**

Data Out provides the option of controlling the 2 I/O ports on the SSI interface depending on the value of a transducer input. Output data consists of 4Byte.

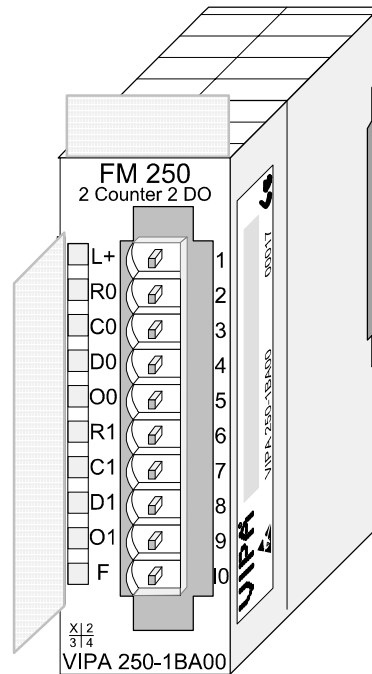
The SSI transducer stores 8Byte of output data, i.e. you may define two comparative values along with the respective control byte.

In the control byte you are able to specify how the reference value should affect which output and whether the status of the I/Os should be signaled via the input bytes.

The following table shows the assignment of these output bytes.

Byte	Data Out
0	Bit 0-1: set point value 00: no set point value 01: for output 0 10: for output 1 11: for both outputs Bit 2: status transfer into input data area 0: no status transfer 1: status transfer to input area Bit 3: set conditions for output 0: when actual value exceeds comparison value 1: when actual value is less than comparison value Bit 4-7: reserved
1	Comparison value: HB
2	Comparison value: MB
3	Comparison value: LB

## FM 250 - Counter module - Construction



### Note!

The following information is only applicable to counter modules with *order no.:* VIPA 250-1BA00 and a revision level 5 and higher.

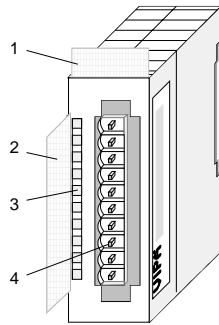
The counter module accepts the signals from transducers connected to the module and processes these pulses in accordance with the selected mode of operation. The module has 2 channels with a data resolution of 32Bit each.

These modules provide 24 counter modes and one 24V output per channel that is controlled in accordance with the selected mode.

### Properties

- two 32Bit channels
- DC 24V supply voltage or via backplane bus
- freely configurable DC 24V outputs (0.5A max.)
- Counters and compare registers are loaded by means of a control byte
- Standard up-down counter with a resolution of 32Bit or 16Bit
- Compare and auto-reload functions
- Different modes for encoder pulses
- Pulse-width measurements and frequency measurements

Construction

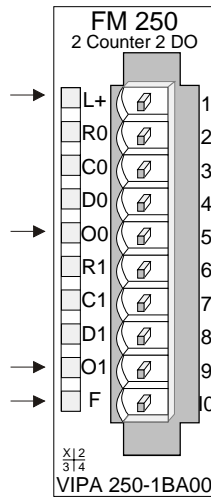


- [1] Label for module name
- [2] Label for bit address with description
- [3] LED status indicator
- [4] Edge connector

Status indicator pin assignment

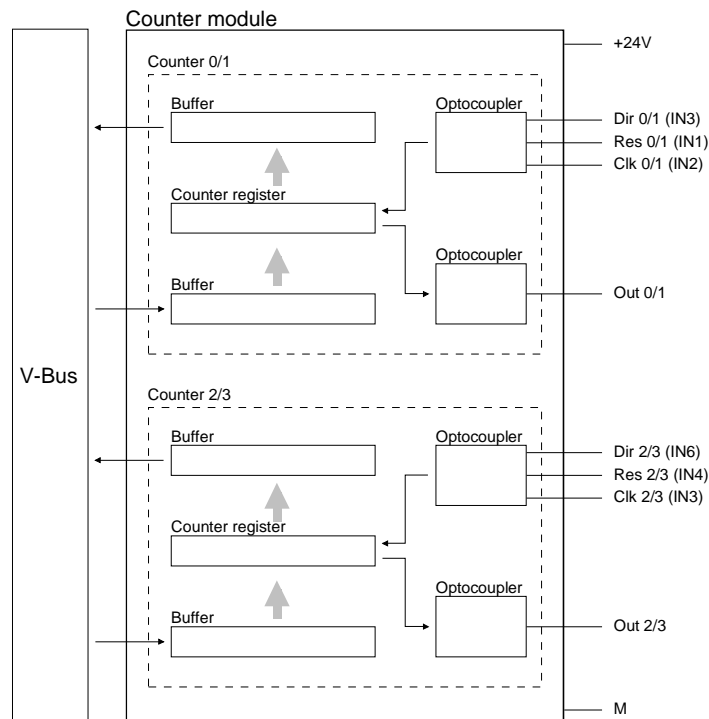
LED	Description	Pin	Assignment
-----	-------------	-----	------------

L+	LED (yellow) Supply voltage available	1	Supply voltage +24V DC
O0	LED (green) Output counter 0	5	OUT0 output counter 0/1
O1	LED (green) Output counter 1	9	OUT1 output counter 2/3
F	LED (red) Error /overload	10	Common of supply voltage



2	IN1 input 1 counter 0/1
3	IN2 input 2 counter 0/1
4	IN3 input 3 counter 0/1
6	IN4 input 4 counter 2/3
7	IN5 input 5 counter 2/3
8	IN6 input 6 counter 2/3

Block diagram





**Access to the counter module**

The module has 2 channels with a resolution of 32Bit each. You may use parameters to specify the mode for each channel. The pin assignment for the channel depends upon the selected mode (see description of modes).

10 data bytes are required for the data input and output. Data output to a channel of a counter requires 10Byte, for example for defaults or for comparison values. In the latter case Byte 9 (control) is used to initiate a write operation into the required registers of the counter as every counter word is associated with a bit in the 9th byte. The respective values are transferred into the counter registers when they are toggled (0→1).

The 10<sup>th</sup> byte (status byte) controls the behavior of the counter during a restart of the next higher master module. You may set the counter level to remanent by means of a combination of Bits 0 and 1; i.e. the original counter level will not be reset when the next higher master module restarts.

The following combinations are possible:

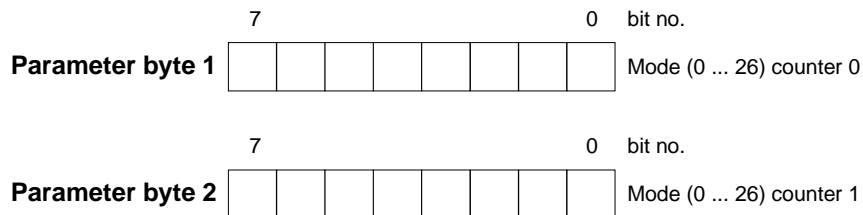
- Bit 0=1, Bit 1=0            counter value is remanent during restart
- Bit 0=x, Bit 1=1           counter value is reset during restart (default)

You may check your settings at any time by reading Byte 10 of the output data.

<p>Data sent to module</p> <table border="1" style="border-collapse: collapse; width: 100%;"> <tr><td>00h</td><td>DE0</td><td rowspan="4" style="padding-left: 10px;">Counter 0/1</td></tr> <tr><td>01h</td><td>DE1</td></tr> <tr><td>02h</td><td>DE2</td></tr> <tr><td>03h</td><td>DE3</td></tr> <tr><td>04h</td><td>DE4</td><td rowspan="4" style="padding-left: 10px;">Counter 2/3</td></tr> <tr><td>05h</td><td>DE5</td></tr> <tr><td>06h</td><td>DE6</td></tr> <tr><td>07h</td><td>DE7</td></tr> <tr><td>08h</td><td>Control</td><td></td></tr> <tr><td>09h</td><td>Status</td><td></td></tr> </table>	00h	DE0	Counter 0/1	01h	DE1	02h	DE2	03h	DE3	04h	DE4	Counter 2/3	05h	DE5	06h	DE6	07h	DE7	08h	Control		09h	Status		<p>Data received from module</p> <table border="1" style="border-collapse: collapse; width: 100%;"> <tr><td>00h</td><td>DA0</td><td rowspan="4" style="padding-left: 10px;">Counter 0/1</td></tr> <tr><td>01h</td><td>DA1</td></tr> <tr><td>02h</td><td>DA2</td></tr> <tr><td>03h</td><td>DA3</td></tr> <tr><td>04h</td><td>DA4</td><td rowspan="4" style="padding-left: 10px;">Counter 2/3</td></tr> <tr><td>05h</td><td>DA5</td></tr> <tr><td>06h</td><td>DA6</td></tr> <tr><td>07h</td><td>DA7</td></tr> <tr><td>08h</td><td></td><td></td></tr> <tr><td>09h</td><td>Status</td><td></td></tr> </table>	00h	DA0	Counter 0/1	01h	DA1	02h	DA2	03h	DA3	04h	DA4	Counter 2/3	05h	DA5	06h	DA6	07h	DA7	08h			09h	Status	
00h	DE0	Counter 0/1																																															
01h	DE1																																																
02h	DE2																																																
03h	DE3																																																
04h	DE4	Counter 2/3																																															
05h	DE5																																																
06h	DE6																																																
07h	DE7																																																
08h	Control																																																
09h	Status																																																
00h	DA0	Counter 0/1																																															
01h	DA1																																																
02h	DA2																																																
03h	DA3																																																
04h	DA4	Counter 2/3																																															
05h	DA5																																																
06h	DA6																																																
07h	DA7																																																
08h																																																	
09h	Status																																																

**Configuration parameters**

The configuration parameters consist of 2Byte. You use these bytes to define the operating mode of each channel by means of a mode number. This chapter contains a detailed description of the different modes further below. The different combinations of the various modes are available from the table on the next page. The procedure for the transfer of parameter bytes is available from the description for the System 200V bus coupler or the master system.



## Summary of counter modes and interfacing

Mode	may be combined	Function	IN1	IN2	IN3	IN4	IN5	IN6	OUT0	OUT1	Auto Re-load	Compare Load
			<b>Counter 0/1</b>			<b>Counter 2/3</b>						
0	yes	32 bit counter	RES	CLK	DIR	RST	CLK	DIR	=0	=0	no	=0
1	yes	Encoder 1 edge	RES	A	B	RST	A	B	=0	=0	no	=0
3	yes	Encoder 2 edges	RES	A	B	RST	A	B	=0	=0	no	=0
5	yes	Encoder 4 edges	RES	A	B	RST	A	B	=0	=0	no	=0
			<b>Counter 1 counter 0</b>			<b>Counter 3 counter 2</b>						
8	yes	2x16 bit counter up/up	-	CLK	CLK	-	CLK	CLK	-	-	no	no
9	yes	2x16 bit counter down/up	-	CLK	CLK	-	CLK	CLK	-	-	no	no
10	yes	2x16 bit counter up/down	-	CLK	CLK	-	CLK	CLK	-	-	no	no
11	yes	2x16 bit counter down/down	-	CLK	CLK	-	CLK	CLK	-	-	no	no
			<b>Counter 0/1</b>			<b>Counter 2/3</b>						
12	yes	32 bit counter up + gate	RES	CLK	Gate	RST	CLK	Gate	=comp	=comp	no	yes
13	yes	32 bit counter down + gate	RES	CLK	Gate	RST	CLK	Gate	=comp	=comp	no	yes
14	yes	32 bit counter up + gate	RES	CLK	Gate	RST	CLK	Gate	=comp	=comp	yes	yes
15	yes	32 bit counter down + gate	RES	CLK	Gate	RST	CLK	Gate	=comp	=comp	yes	yes
			<b>Combination of counter 0 ... 3</b>									
16	no	Frequency measurement	RES	CLK	Start	Stop	-	-	Meas. active	Meas. compl.	no	yes
17	no	Period measurement	RES	CLK	Start	Stop	-	-	Meas. active	Meas. compl.	no	yes
18	no	Frequency measurement with gate output	RES	CLK	Start	Stop	-	-	Meas. gate	Gate	no	yes
19	no	Period measurement with gate output	RES	CLK	Start	Stop	-	-	Meas. gate	Gate	no	yes
			<b>Counter 0/1</b>			<b>Counter 2/3</b>						
6	yes	Pulse low, 50kHz with Direction Input	RES	Pulse	DIR	RES	Pulse	DIR	-	-		
20	yes	Pulse low, prog. time base with Direction Input	RES	Pulse	DIR	RES	Pulse	DIR	-	-		
21	yes	Pulse low, up, prog. time base with Gate	RES	Pulse	Gate	RES	Pulse	Gate	-	-		
22	yes	Pulse high, up, prog. time base with Gate	RES	Pulse	Gate	RES	Pulse	Gate	-	-		
			<b>Counter 0/1</b>			<b>Counter 2/3</b>						
23	yes	One Shot, up, Set	RES	CLK	Gate	RES	CLK	Gate			no	yes
24	yes	One Shot, down, Set	RES	CLK	Gate	RES	CLK	Gate			no	yes
25	yes	One Shot, up, Reset	RES	CLK	Gate	RES	CLK	Gate			no	yes
26	yes	One Shot, down, Reset	RES	CLK	Gate	RES	CLK	Gate			no	yes
			<b>Counter 0/1</b>			<b>Counter 2/3</b>						
27	yes	32 bit counter	Gate/R	CLK	DIR	Gate/R	CLK	DIR	=0	=0	no	=0
28	yes	Encoder 1 edge	Gate/R	A	B	Gate/R	A	B	=0	=0	no	=0
29	yes	Encoder 2 edges	Gate/R	A	B	Gate/R	A	B	=0	=0	no	=0
30	yes	Encoder 4 edges	Gate/R	A	B	Gate/R	A	B	=0	=0	no	=0

Due to technical advances the revision level and the functionality of the counter module was continuously expanded. Below follows a list that allocates the different modes to the revision level:

- |           |                  |               |                    |
|-----------|------------------|---------------|--------------------|
| Mode 0-5  | revision level 3 | Mode 6, 20-26 | revision level 6/7 |
| Mode 0-17 | revision level 4 | Mode 27-30    | revision level 8   |
| Mode 0-19 | revision level 5 |               |                    |

**Terminology****RES**

RESET signal that has to be LOW during the measuring process. A HIGH level erases one or both counters, depending on the selected mode.

**CLK**

The clock signal from the connected transducer.

Start or Stop

A HIGH level starts or stops the counter. When the start level is active, the counter will start with the next CLK pulse that corresponds to the selected mode.

**DIR**

In mode 0 the level of the DIR signal determines the direction of the counting process.

LOW level: count up

HIGH level: count down

**Auto Reload**

The Auto Reload function transfers a user-defined value into the counter when the counter reaches the number contained in the compare register.

**Compare Load**

You may use the compare function to specify an stop value for the counter. Depending on the selected mode an output is activated or the counter is re-started when it reaches this value.

**Gate**

Gate signal enabling the counter.

**Gate/R**

The rising edge of this signal sets the counter back. As long as the signal is at "1", the counter is released.

**Measurement gate**

Status indicator of the counter activity - is set to a HIGH level after the 1<sup>st</sup> CLK signal and LOW level after the last CLK signal (mode 18 ... 19).

**Pulse**

The pulse width of the introduced signal is determined by means of the internal time base.

**Fref**

Reference or clock frequency that is set permanently to 50kHz in mode 6.

The clock frequency "Fref" for counter modes 20, 21, 22 is programmable:

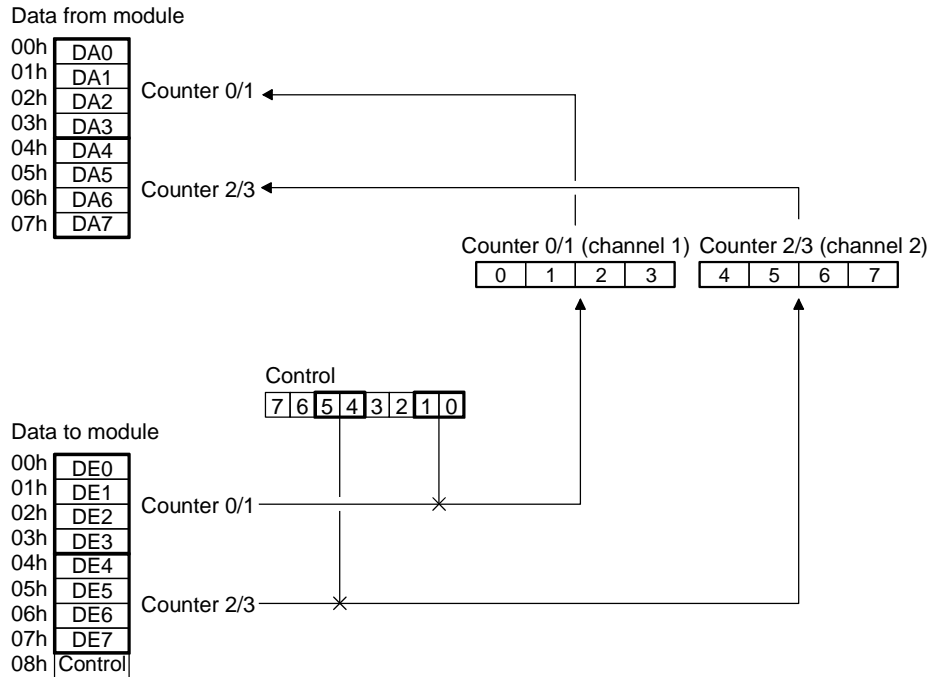
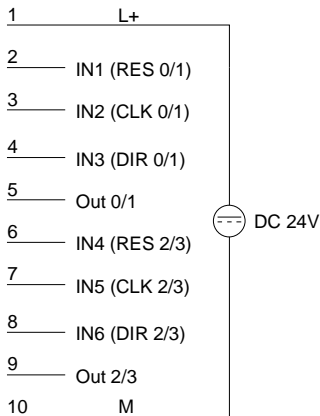
Parameter	Fref
0	10MHz
1	1MHz
2	100kHz
3	10kHz

# Counter modes

## Mode 0 32Bit counter

In mode 0 two counters (16Bit) are combined to produce a 32Bit counter. You determine the direction by means of the DIR input (IN3 or IN6). Every rising or falling edge of the input clock signal increments or decrements the counter. During the counting process the RES signal must be at a LOW level. If the RES signal is at a HIGH level, the counter is cleared. When the counter reaches zero, output OUT of the respective counter is active for a minimum period of 100ms, even if the counter should continue counting. If the counter stops at zero, the output remains active.

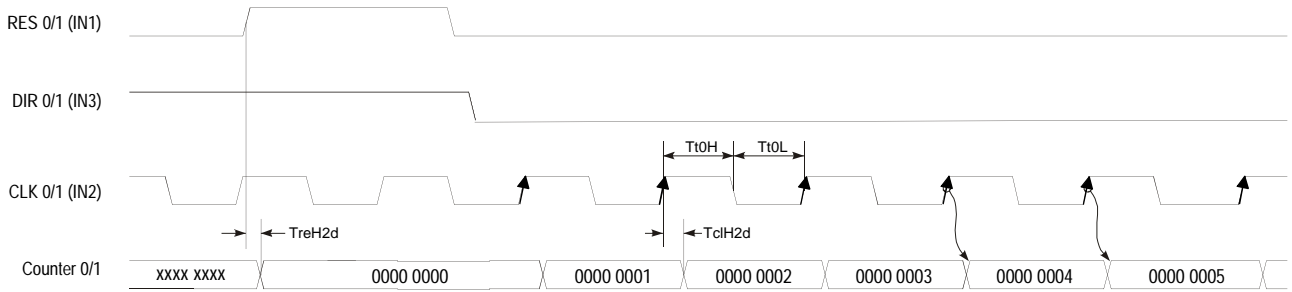
### Pin assignment access to counter



**Up counter**

In mode 0, a LOW level at the DIR input configures the counter for counting up.

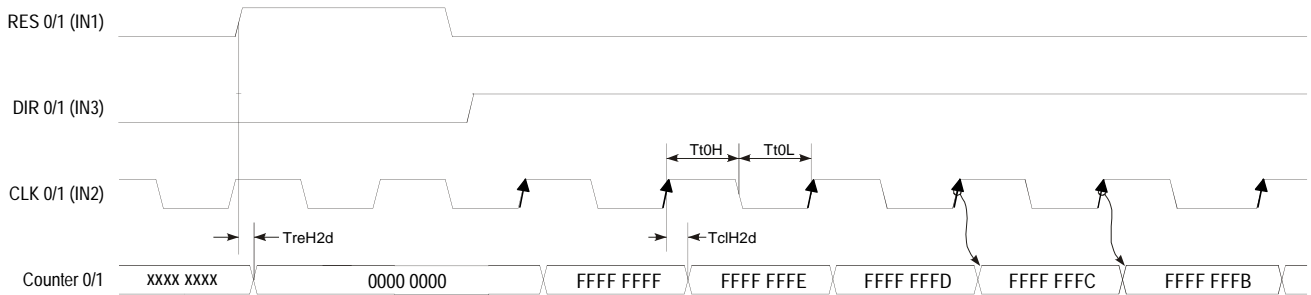
Timing diagram of the counter 0/1 example:



**Down counter**

In mode 0, a HIGH level at the DIR input configures the counter for counting down.

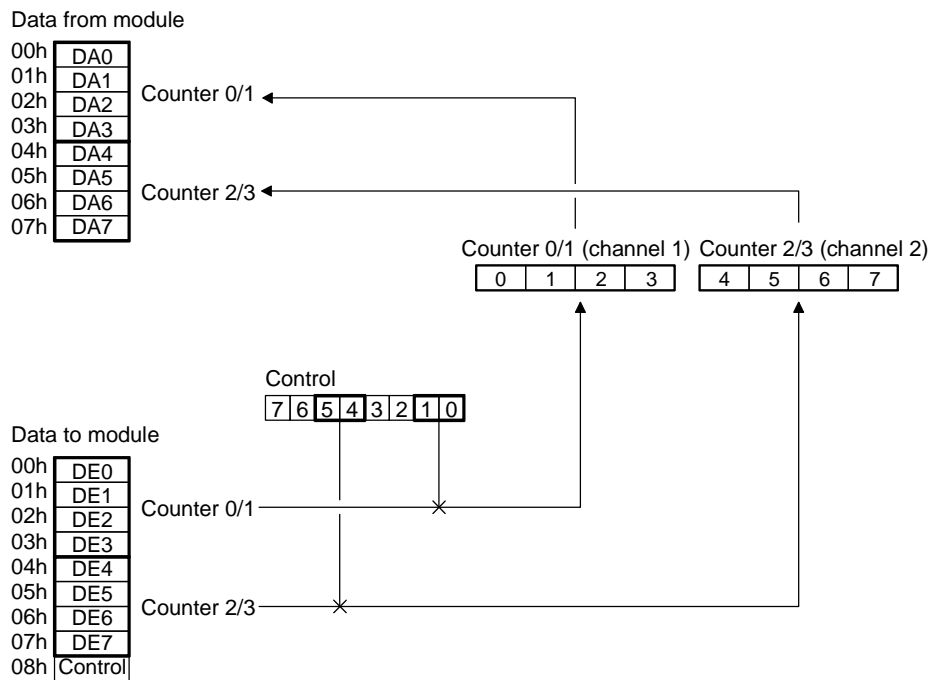
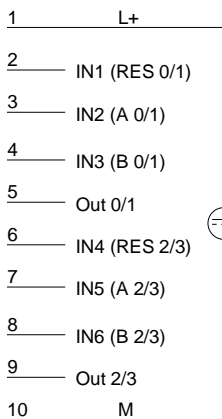
Timing diagram of the counter 0/1 example:



**Mode 1  
Encoder 1 edge**

In mode 1 you may configure an encoder for one of the channels. Depending on the direction of rotation this encoder will increment or decrement the internal counter with every falling edge. The RES input has to be at a LOW level during the counting process. A HIGH level clears the counter. When the counter reaches zero, output OUT of the respective counter is active for a minimum period of 100ms, even if the counter continues counting. If the counter stops at zero the output remains active.

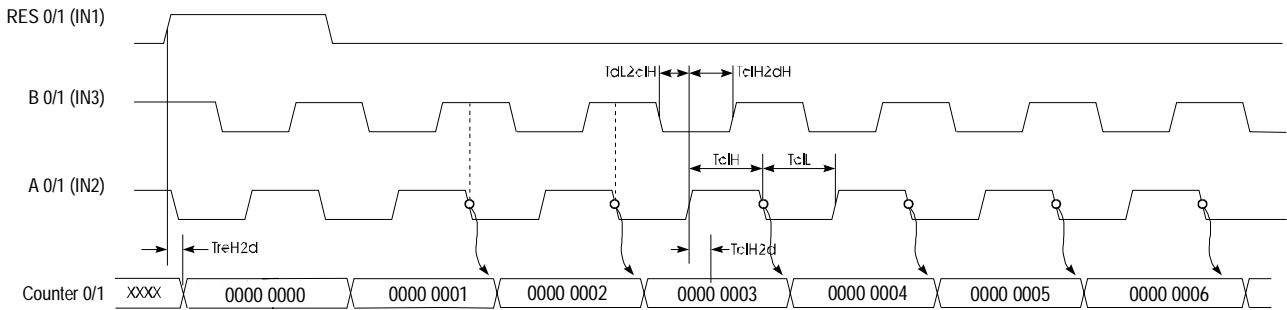
**Pin assignment  
access to counter**



**Up counter**

Every falling edge of the signal at input A increments the counter if input B is at HIGH level at this moment.

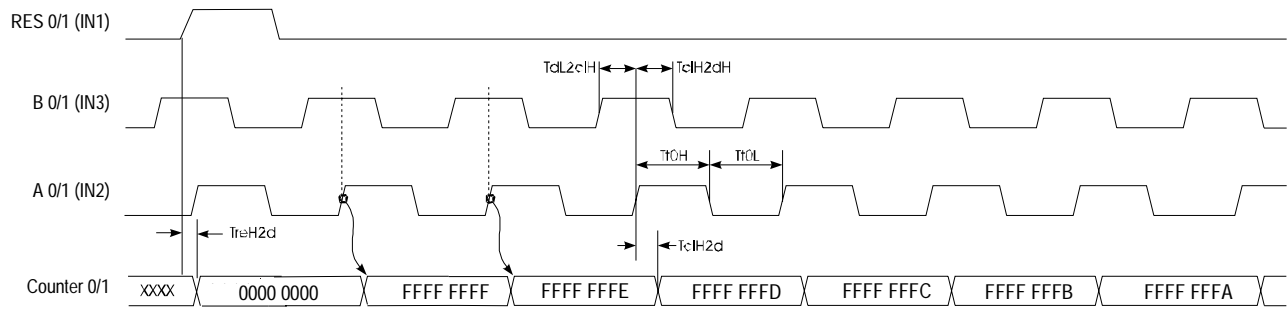
Timing diagram for the counter 0/1 example:



**Down counter**

Every rising edge of the signal at input A decrements the internal counter if input B is at HIGH level at this moment.

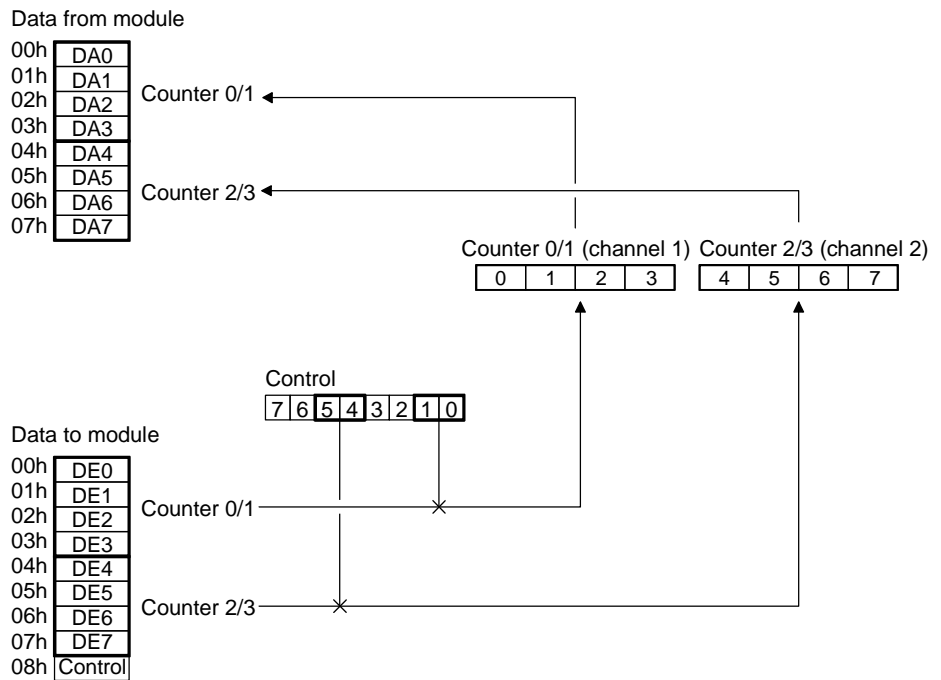
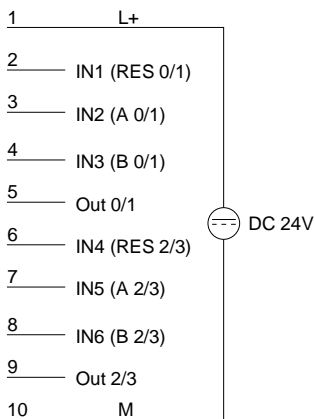
Timing diagram for the counter 0/1 example:



**Mode 3  
Encoder 2 edges**

Every rising or falling edge of the signal at input A changes the counter by 1. The direction of the count depends on the level of the signal applied to input B. RES has to be at a LOW level during the counting process. A HIGH level clears the counter. When the counter reaches zero, output OUT of the respective counter is active for a minimum period of 100ms, even if the counter continues counting. If the counter stops at zero the output remains active.

**Pin assignment  
access to counter**

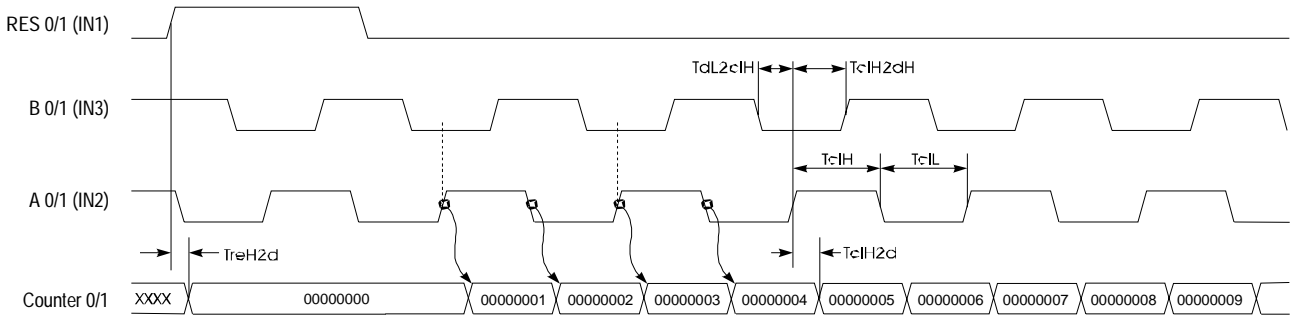




**Up counter**

The counter is incremented by the rising edge of signal A if input B is at a LOW level or by the falling edge of input A when input B is at a HIGH level.

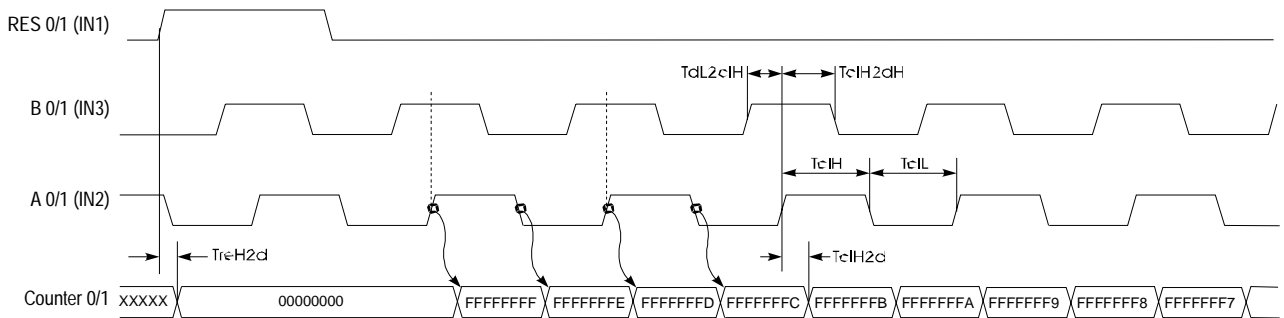
Timing diagram for the counter 0/1 example:



**Down-counter**

The counter is decremented by the rising edge of signal A if input B is at a HIGH level or by the falling edge of input A when input B is at a LOW level.

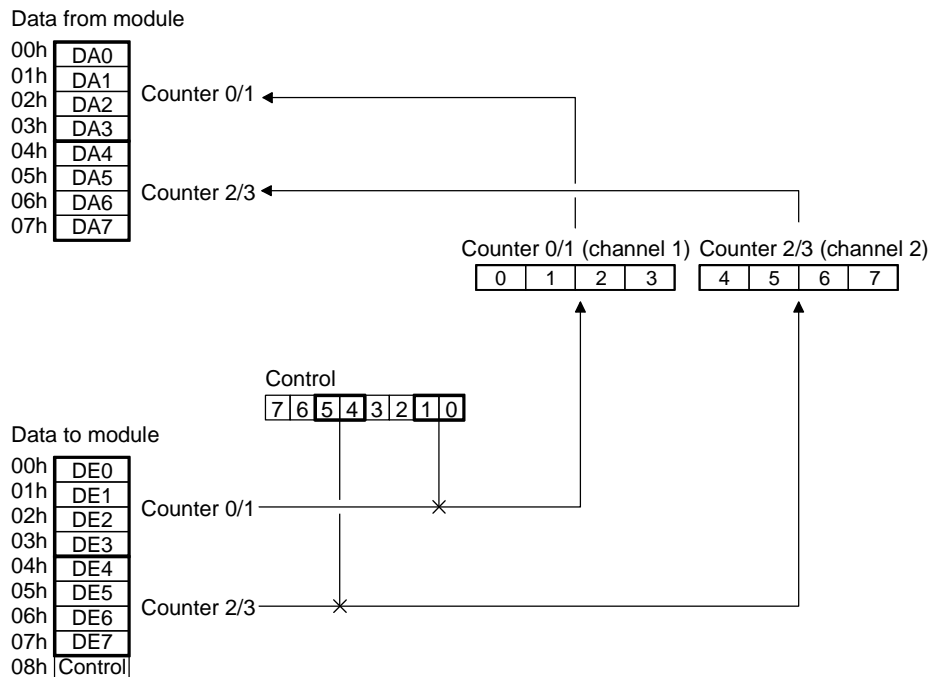
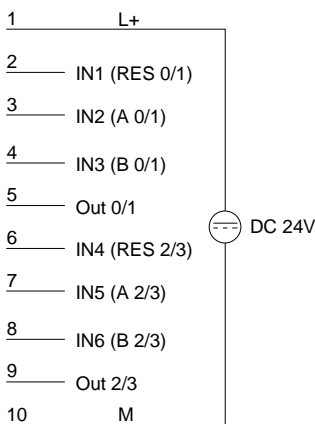
Timing diagram for the counter 0/1 example:



**Mode 5  
Encoder 4 edges**

Every rising or falling edge at inputs A or B increments or decrements the counter. The direction depends on the level applied to the other input (B or A). RES has to be at a LOW level during the counting process. A HIGH level clears the counter. When the counter reaches zero, output OUT of the respective counter is active for a minimum period of 100ms, even if the counter continues counting. If the counter stops at zero, the output remains active.

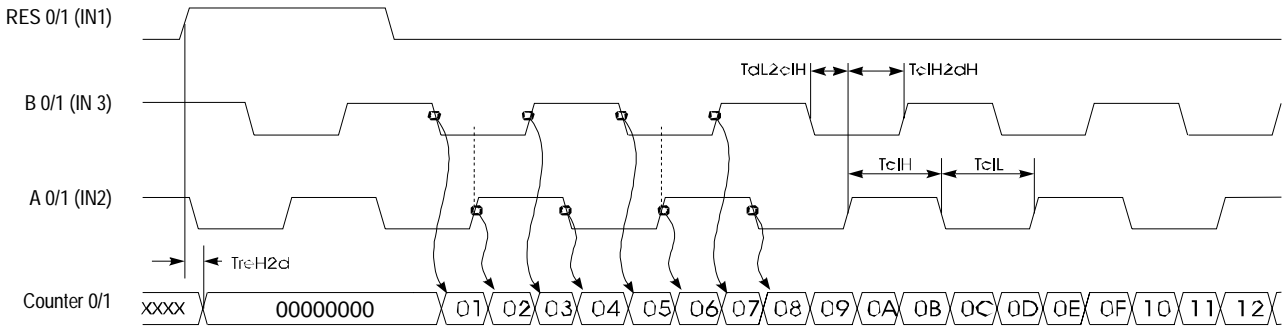
**Pin assignment  
access to counter**



**Up counter**

The counter is incremented when a rising edge is applied to B while input A is at a HIGH level or if a falling edge is applied to B when input A is at a LOW level. Alternatively it is also incremented when a rising edge is applied to A when input B is at a LOW level or by a falling edge at A when input B is at a HIGH level.

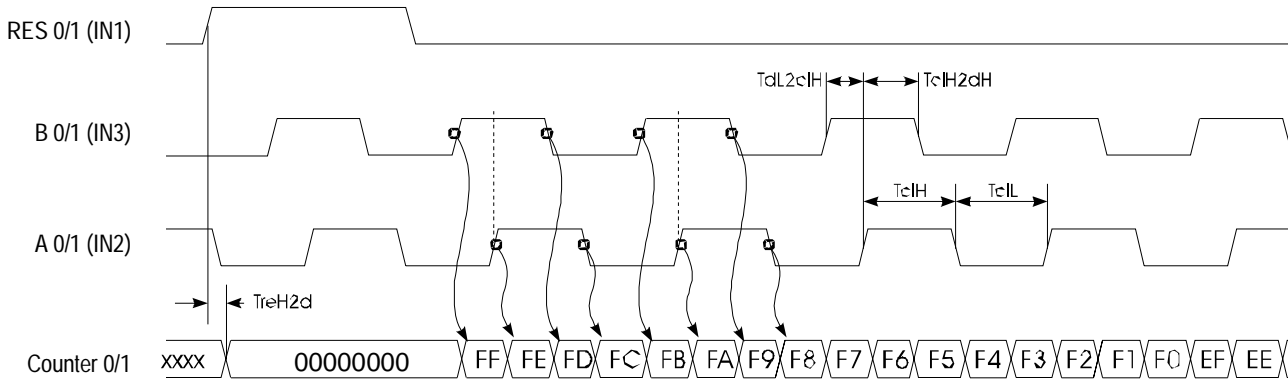
Timing diagram for the counter 0/1 example:



**Down counter**

The counter is decremented when a rising edge is applied to B while input A is at a LOW level or if a falling edge is applied to B when input A is at a HIGH level. Alternatively it is also decremented when a rising edge is applied to A when input B is at a HIGH level or by a falling edge at input A when input B is at a LOW level.

Timing diagram for counter 0/1 example:



**Mode 6  
pulse measuring,  
Pulse LOW, 50kHz  
with direction  
control**

The pulse width of a signal connected to the CLK input is determined by means of an internal time base and saved. The measurement is started with the falling edge of the input signal and it is stopped by the rising edge of the input. This saves the value in 20µs units in a buffer from where it may be retrieved (corresponds to f ref = 50kHz).

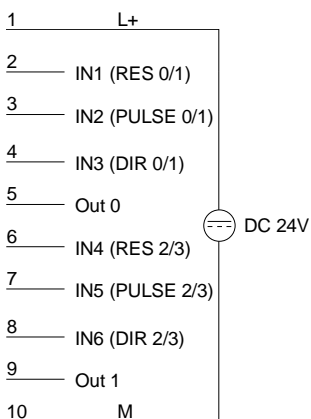
Input DIR determines the counting direction of the counter. If DIR is at a LOW level the counter counts up. A HIGH level lets the counter count down.

The input RES has to be at a LOW level. A HIGH at this input would clear the counter.

With the rising edge of the signal pulse, a result is transferred into the DA area; the result remains available until it is overwritten by the next new result.

Signals Out 0 or Out 1 are not modified.

**Pin assignment  
access to counter**

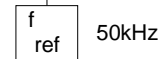
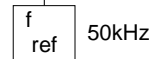
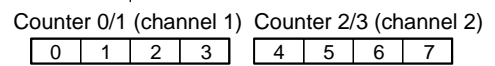


Data from module

00h	DA0
01h	DA1
02h	DA2
03h	DA3
04h	DA4
05h	DA5
06h	DA6
07h	DA7

Counter 0/1

Counter 2/3



Data to module

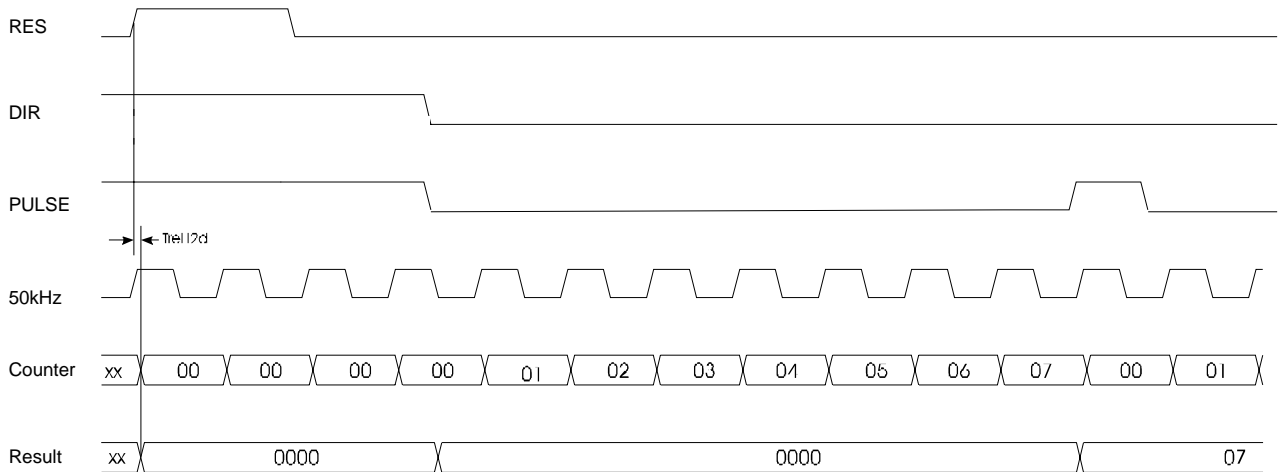
00h	
01h	
02h	
03h	DE3
04h	
05h	
06h	
07h	DE7
08h	Control

Counter 0/1

Counter 2/3

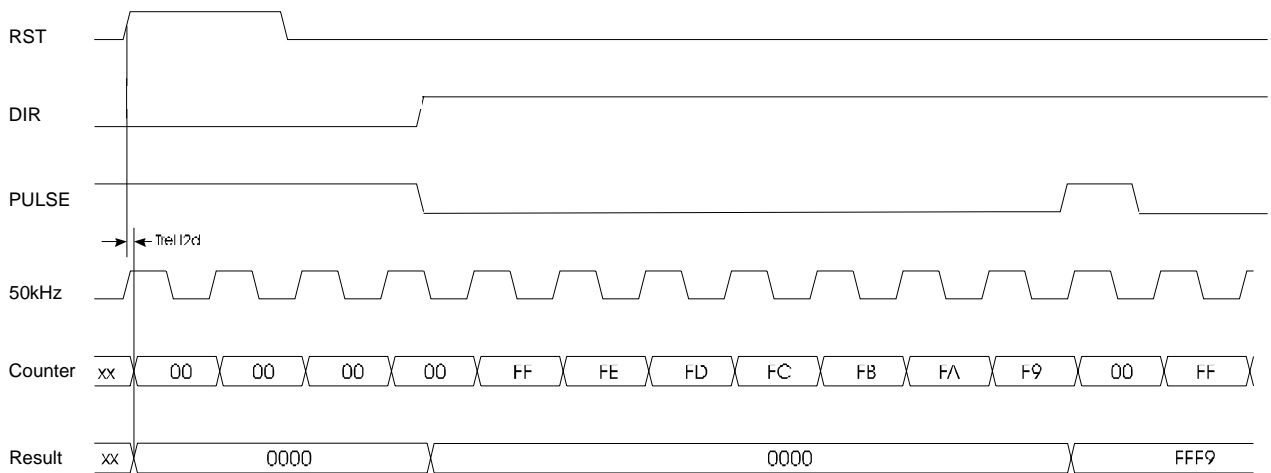
**Up counter**

The RES signal (R0) and the DIR signal (D0) are reset. The measurement is started by the falling edge at input PULSE (C0) and the counter is clocked up by the 50kHz clock. The rising edge of the signal at input PULSE (C0) terminates the count operation and the result is transferred into the result register. The result is available to the PLC. The value remains in the result register until a new measurement has been completed which overwrites the register.



**Down counter**

The RES signal (R0) is reset and the DIR signal (D0) is placed at a HIGH level. The measurement is started by the falling edge at input PULSE (C0) and the counter is clocked down by the 50kHz clock. The rising edge of the signal at input PULSE (C0) terminates the count operation and the result is transferred into the result register. The result is available to the PLC. The value remains in the result register until a new measurement has been completed which overwrites the register.

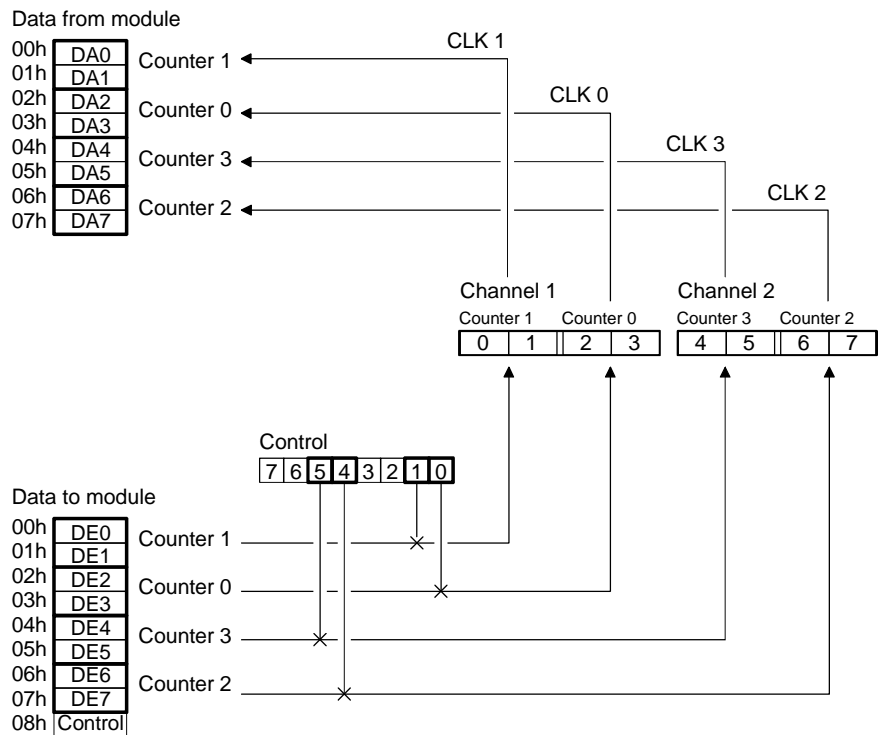
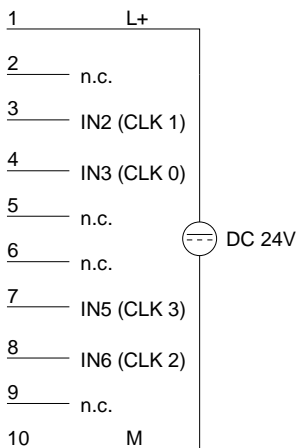


**Mode 8 ... 11  
two input counter  
function**

In this mode each channel provides 2 counters of 16Bit each. The rising edge of the input clock CLK x increments or decrements the respective counter. In this mode each counter can also be preset to a certain value by means of a control bit. Outputs are not available. A RESET is also not available. The following combinations are possible for every channel:

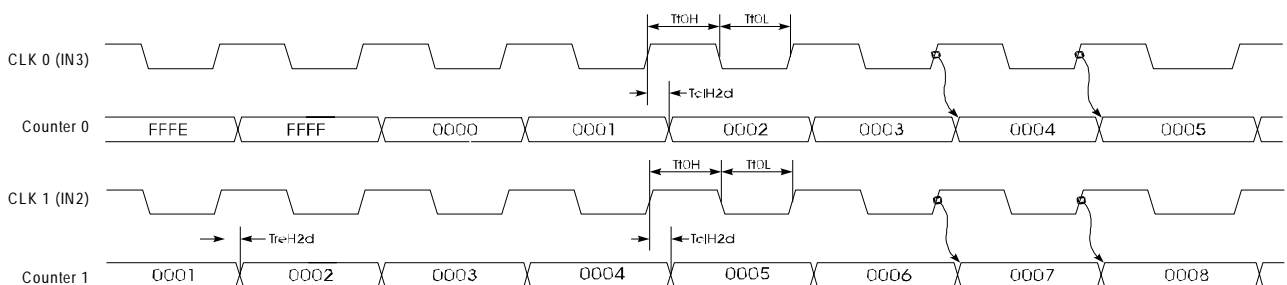
- Mode 8 - counter 0/2 up, counter 1/3 up**
- Mode 9 - counter 0/2 down, counter 1/3 up**
- Mode 10 - counter 0/2 up, counter 1/3 down**
- Mode 11 - counter 0/2 down, counter 1/3 down**

**Pin assignment  
access to counter**



**Timing diagram**

Below follows a timing diagram depicting an example of counter 0 and counter 1 in mode 8:



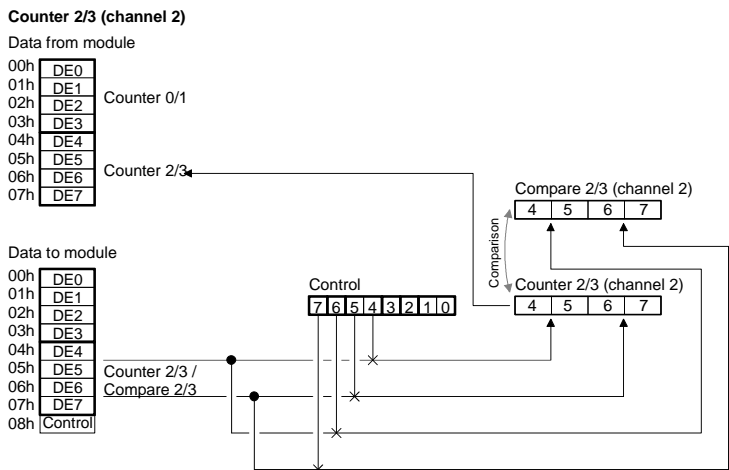
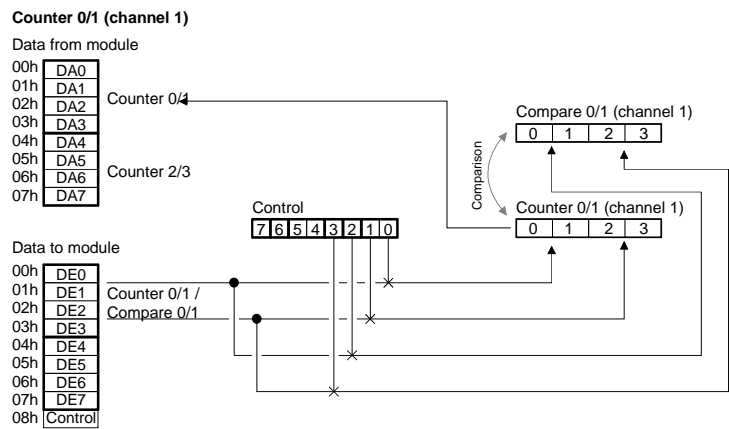
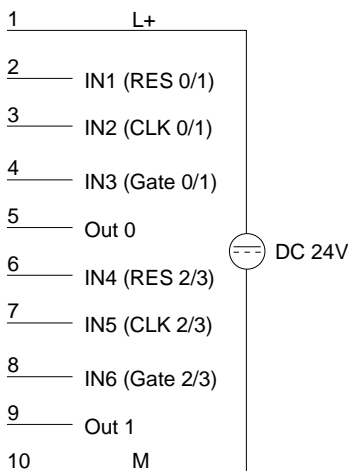
**Mode 12 and 13  
32 bit counter with gate**

In mode 12 and mode 13 you can implement a 32Bit counter that is controlled by a gating signal (Gate). The direction of counting depends on the selected mode. Every rising edge of the input signal increments or decrements the counter provided that the GATE signal is at HIGH level. RES has to be LOW during the counting process. A HIGH level clears the counter. When the counter reaches the value that was previously loaded into the compare register, output OUT is set active for a minimum period of 100ms while the counter continues counting.

**Mode 12 - 32Bit counter up + gate with compare**

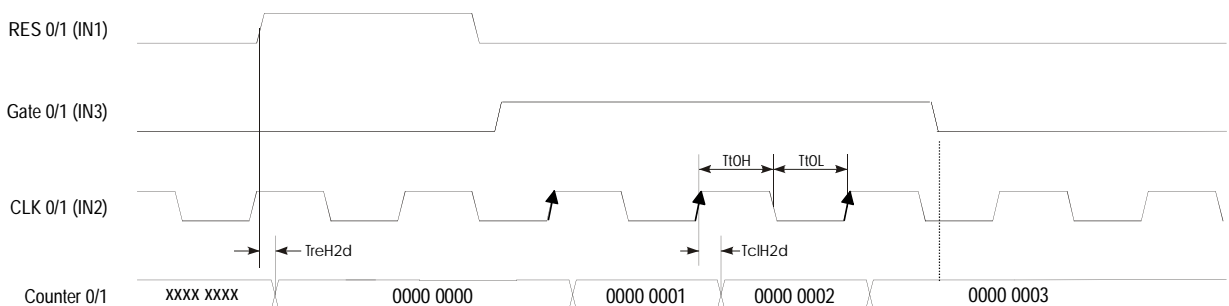
**Mode 13 - 32Bit counter down + gate with compare**

**Pin assignment  
access to counter**



**Timing diagram**

Below follows an example of a timing diagram of counter 0/1 in mode 12:



**Mode 14 and 15  
32Bit counter with  
gate and auto  
reload**

Modes 14 and 15 operate in the same manner as mode 12 and 13 with the addition of an Auto Reload function. The "Auto Reload" is used to define a value in the load register that is used to preset the counter automatically when it reaches the compare value.

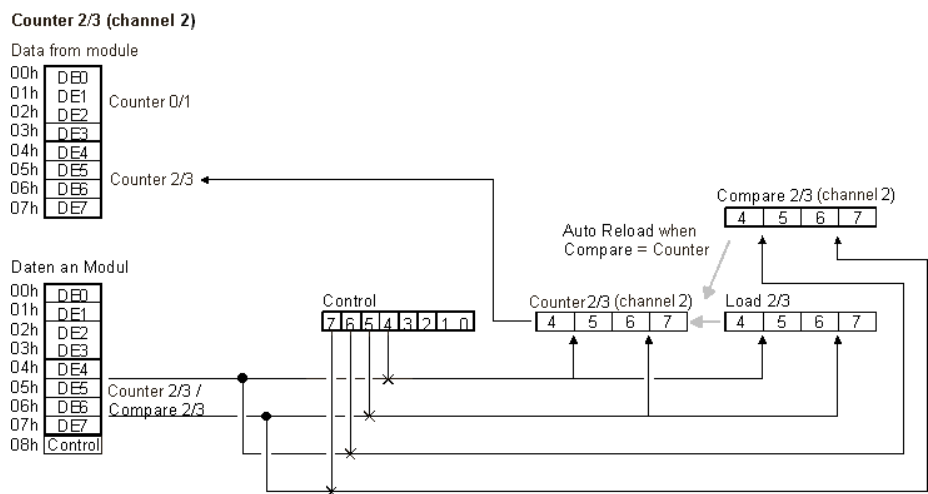
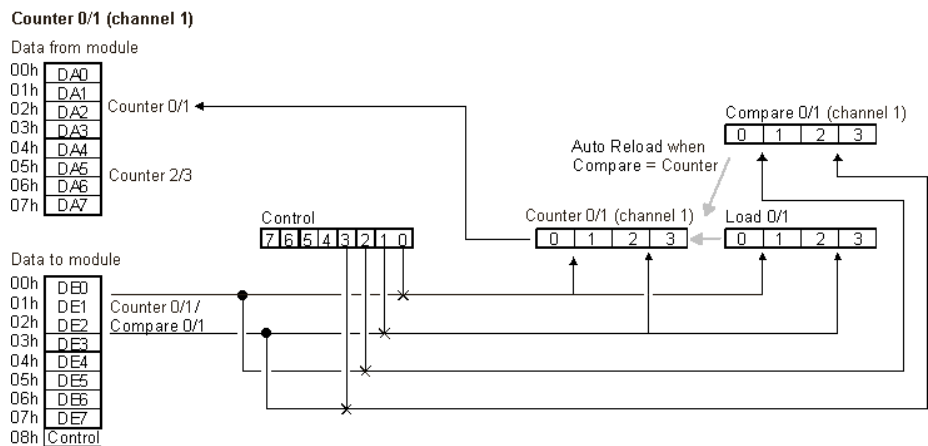
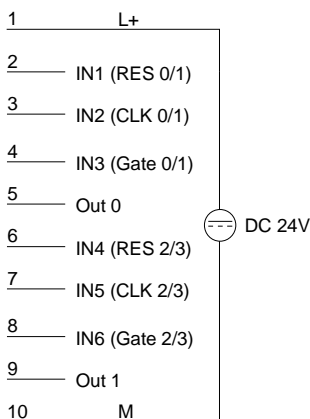
A HIGH pulse applied to RES clears the counter to 0000 0000. A HIGH level applied to GATE enables the counter so that is incremented/decremented by every rising edge of the CLK signal. As long as GATE is HIGH, the counter will count every rising edge of the signal applied to CLK until the count is one less than the value entered into COMPARE. The next pulse overwrites the counter with the value contained in the load register. This process continues until GATE is set to a LOW level. When an auto reload occurs, the status of the respective output changes.

The RES signal only resets the counter but not the outputs.

**Mode 14 - 32Bit counter up + gate with compare and auto reload**

**Mode 15 - 32Bit counter down + gate with compare and auto reload**

**Pin assignment  
access to counter**





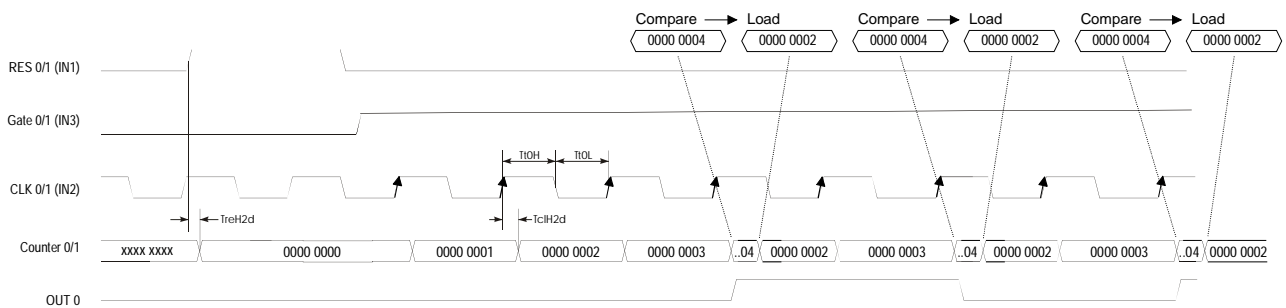
**Example**

This example is intended to explain the operation of the counters in mode 14 and 15.

A HIGH pulse applied to RES clears the counter to 0000 0000. A HIGH level applied to GATE enables the counter. As long as GATE is HIGH the counter will count every rising edge of the signal applied to CLK until the count is one less than the value entered into COMPARE. In this example the counter counts to 0000 0004 followed immediately by an auto reload, i.e. the counter is preset to the contents of the load register (in this case 0000 0002). The state of output OUT 0 changes every time an auto reload is executed.

In this example the counter counts from 0000 0002 to 0000 0004 as long as the GATE input is at a HIGH level.

Every load operation changes the status of output OUT 0.



**Mode 16  
frequency  
measurement**

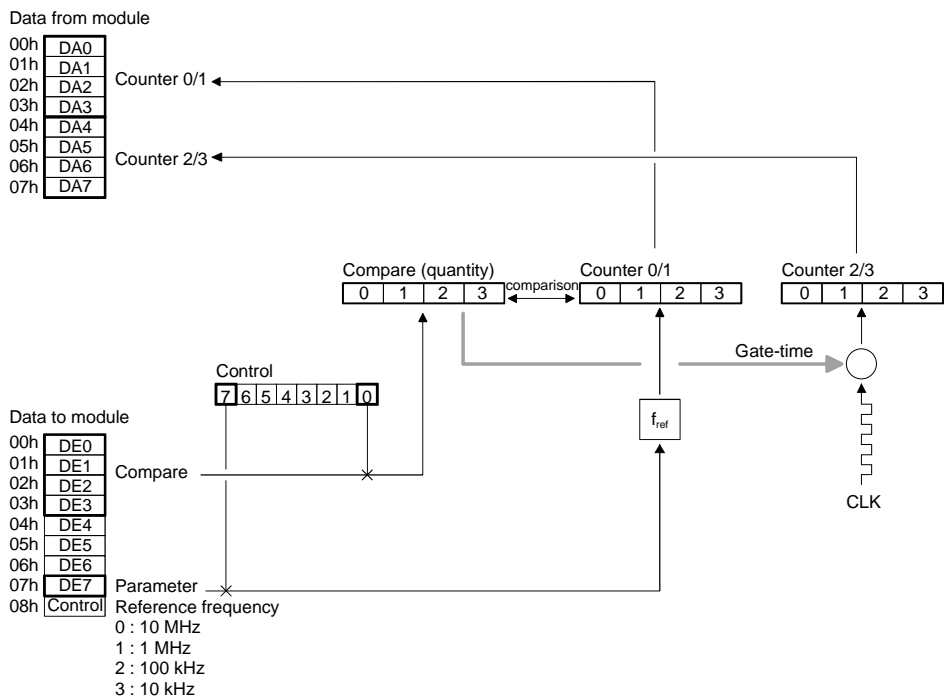
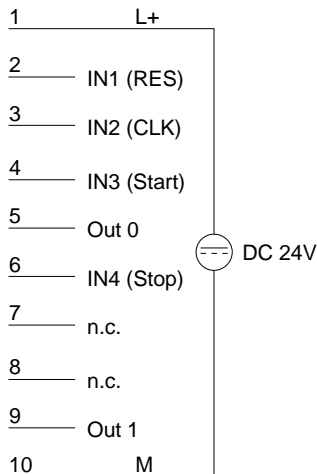
In this mode it is possible to determine the frequency of the signal that is applied to the CLK input. Counter 0/1 is provided with a reference signal by means of DE7 and a gate time that is controlled indirectly by the value n to determine the duration for which counter 2/3 is enabled. The value of n has a range from 1 to  $2^{32}-1$  and it is loaded into the COMPARE register.

When enabled by the rising edge of the signal applied to Start, counter 0/1 counts reference pulses of the reference clock generator from the first rising edge of the CLK signal.

During this time counter 2/3 counts every rising edge of the CLK signal. Both counters are stopped when counter 0/1 reaches the COMPARE value or when a HIGH level is applied to Stop. You may calculate the frequency by means of the formula shown below.

*This mode can not be combined with other modes!*

**Pin assignment  
access to counter**



**Frequency calculation**

When the measurement has been completed you may calculate the frequency as follows:

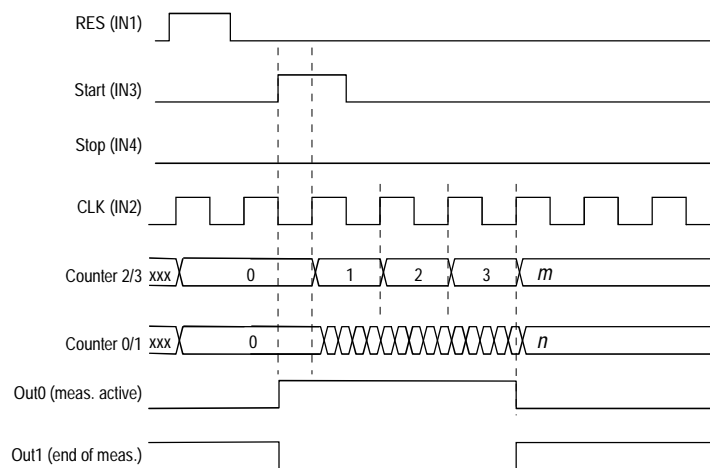
$$Frequency = \frac{fr \cdot m}{n}$$

where *fr* : reference frequency (is supplied via DE7 by means of control bit 7)

*m* : counter 2/3 contents (number of CLK pulses)

*n* : number of reference frequency pulses in counter 0/1 (equal to COMPARE, if the operation was not terminated prematurely by means of Stop)

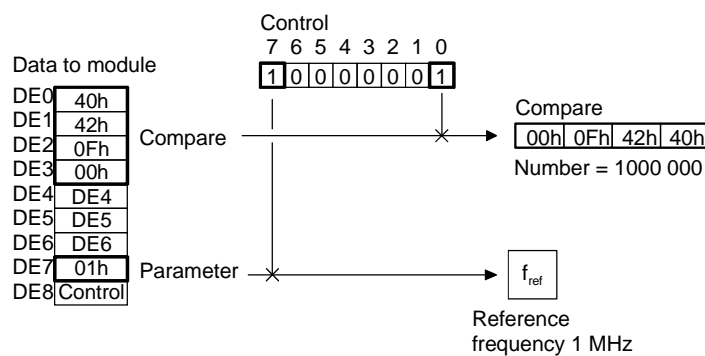
**Timing diagram**



**Example**

Quantity = 1 000 000 pulses

Reference frequency = 1MHz



Using a frequency of 1MHz and 1 000 000 pulses will return 1Hz, i.e. when the measurement is completed, counter 2/3 contains the frequency directly - no conversion is required.



**Note!**

Counter 2/3 will indicate the exact frequency if you choose *fr* and *n* so that the formula returns 1Hz precisely.

**Mode 17  
period  
measurement**

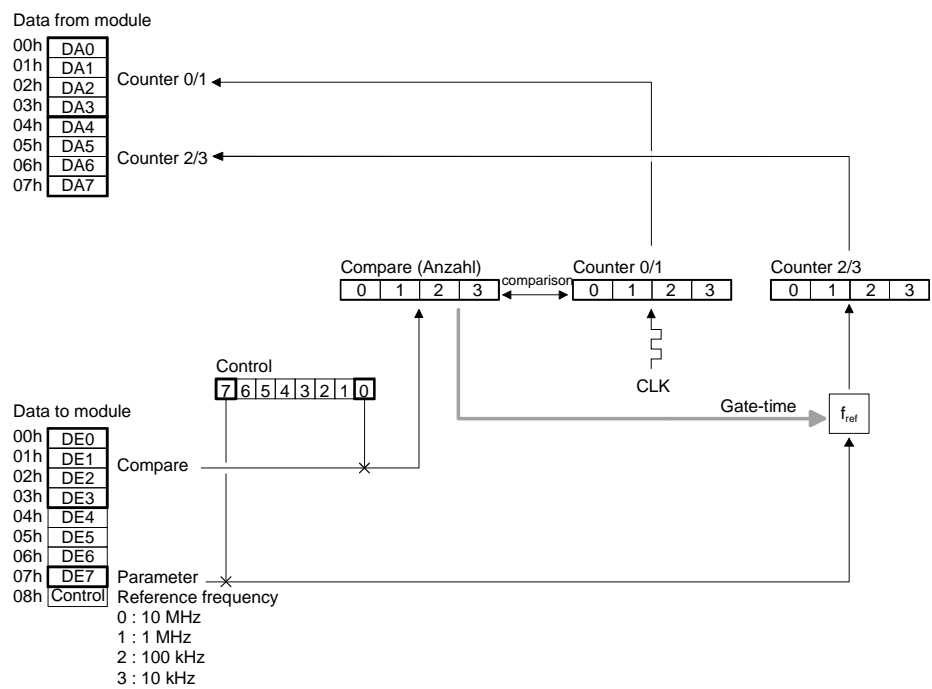
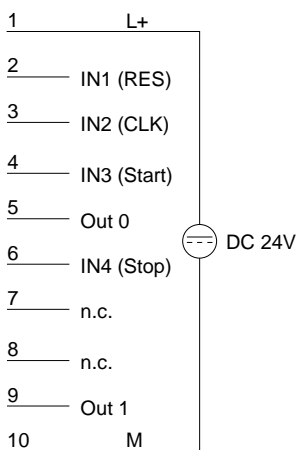
This mode is used to determine the average period of n measuring intervals of a signal that is connected to the CLK input. For this purpose you supply a reference clock to counter 2/3 by means of DE7 and indirectly a gate time defined by the value of n for which counter 2/3 is enabled. The value of n has a range from 1 to  $2^{32}-1$  and it is loaded into the COMPARE register.

The measurement period begins when a rising edge is applied to Start. During this period counter 2/3 counts reference pulses from the reference clock generator starting with the first rising edge of the CLK signal.

In the mean time counter 0/1 counts every rising edge of the CLK signal. Both counters are stopped when the count in counter 0/1 reaches the COMPARE value or when Stop is set to a HIGH level. You may then calculate the average period by means of the formula shown below.

*This mode can not be combined with other modes!*

**Pin assignment  
access to counter**

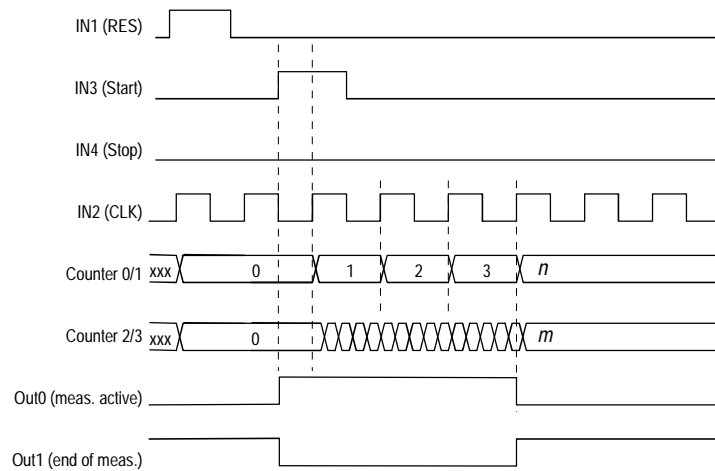


**Period calculation** When the measurement has been completed, you may calculate the period as follows:

$$Period = \frac{n}{fr \cdot m}$$

- where *fr*: reference frequency (supplied in DE7 with control bit 7)
- m*: contents of counter 2/3 (counts reference clock pulses)
- n*: number of CLK pulses in counter 0/1 (corresponds to COMPARE, provided it was not terminated prematurely by Stop)

**Timing diagram:**

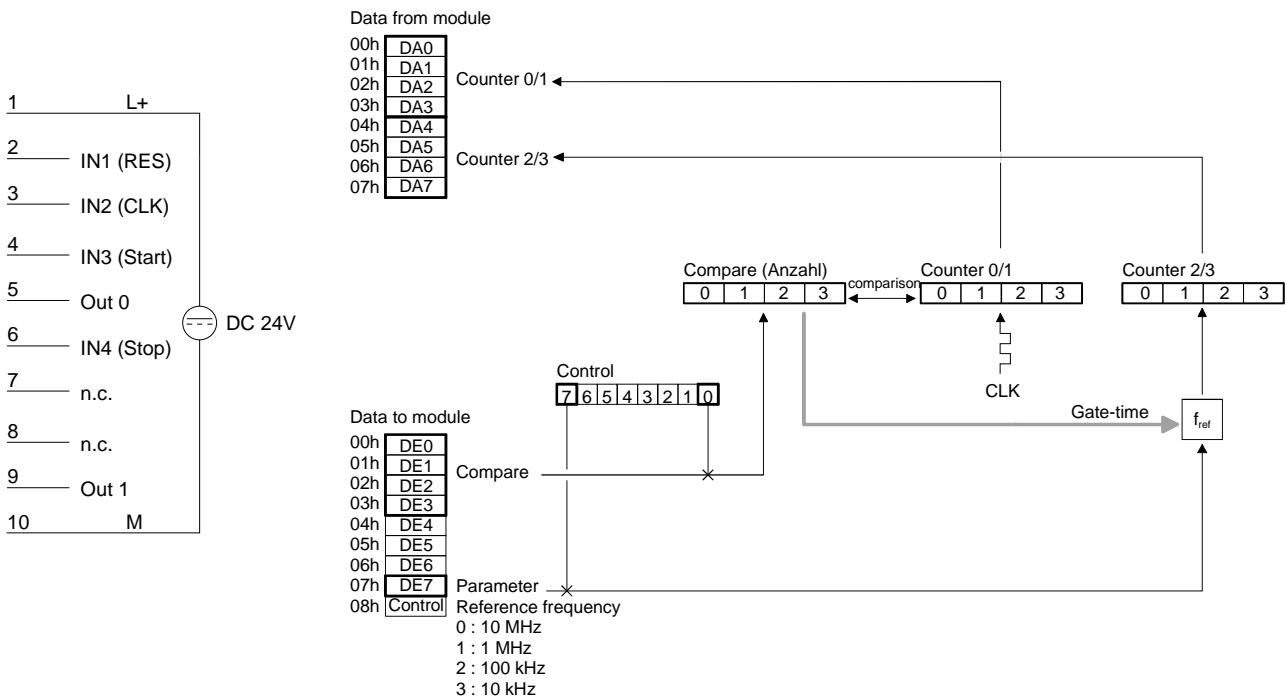


**Mode 18  
frequency  
measurement with  
gate output**

The operation of mode 18 is similar to mode 16. The only difference is the manner in which OUT 0 and OUT 1 are controlled. In this case OUT 0 is only activated when the counting operation starts and it is deactivated when counting ends, i.e. OUT 0 provides an indication of the internal gate. OUT 1 provides the inverted status of the gate.

*This mode can not be combined with other modes!*

**Pin assignment  
access to counter**



**Frequency  
calculation**

When the measurement has been completed, you may calculate the frequency as follows:

$$Frequency = \frac{fr \cdot m}{n}$$

where *fr*: reference frequency (supplied in DE7 with control bit 7)

*m*: contents of counter 2/3 (CLK pulse count)

*n*: number of pulses of the reference frequency in counter 0/1 (corresponds to COMPARE provided it was not terminated prematurely by Stop)

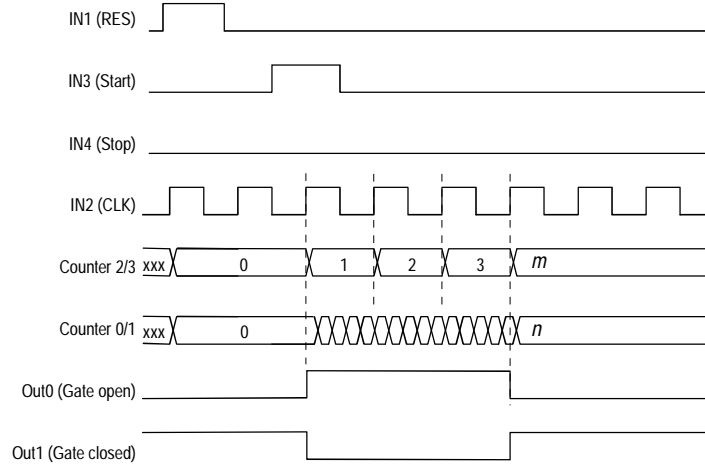


**Note!**

Counter 2/3 will indicate the exact frequency if you choose *fr* and *n* so that the formula returns 1Hz precisely.

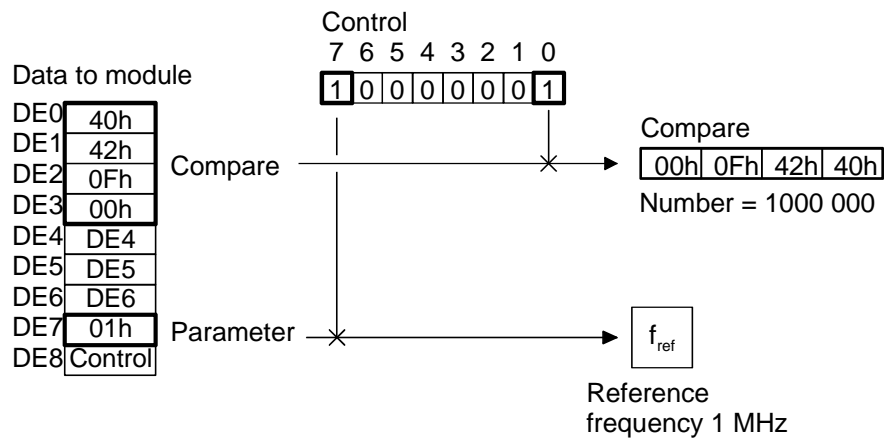
For example when the applied frequency is 1MHz and the number of pulses is 1 000 000 the result will be 1Hz, i.e. counter 2/3 contains the precise frequency after the measurement - this does not require further conversion.

**Timing diagram:**



**Example**

Pulse count = 1 000 000  
 Reference frequency = 1MHz

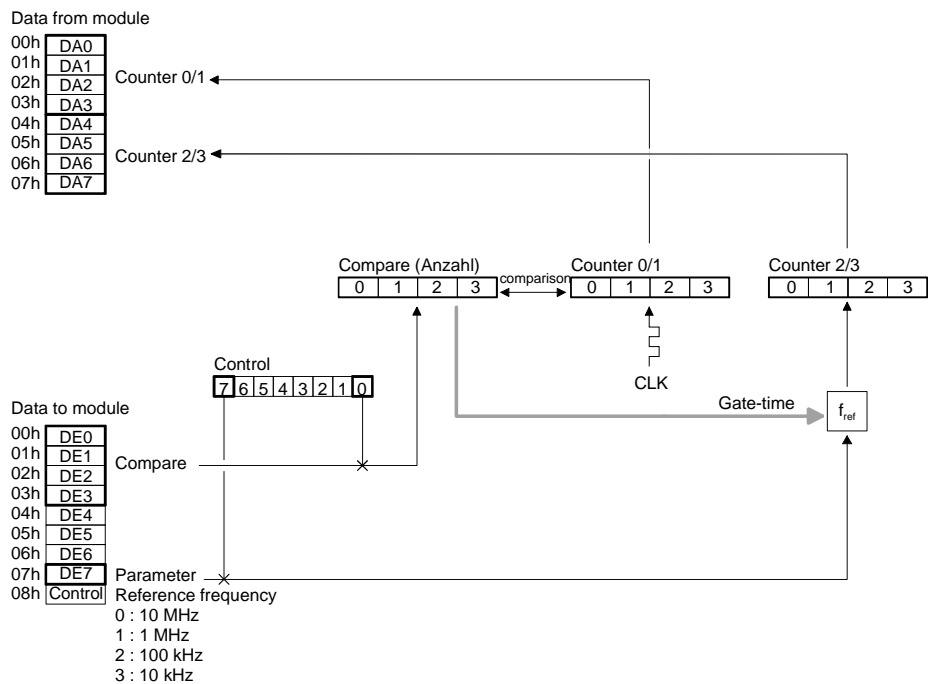
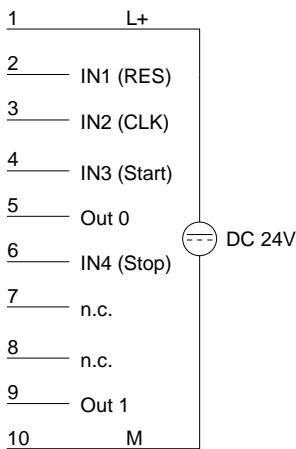


**Mode 19  
period  
measurement with  
gate output**

The operation of mode 19 is identical to mode 17. The only difference is the manner in which OUT 0 and OUT 1 are controlled. Other than for mode 17, OUT 0 is only activated when the counting operation starts and it is deactivated when counting ends, i.e. OUT 0 provides an indication of the internal gate. OUT 1 provides the inverted status of the gate.

*This mode can not be combined with other modes!*

**Pin assignment  
access to counter**



**Period calculation**

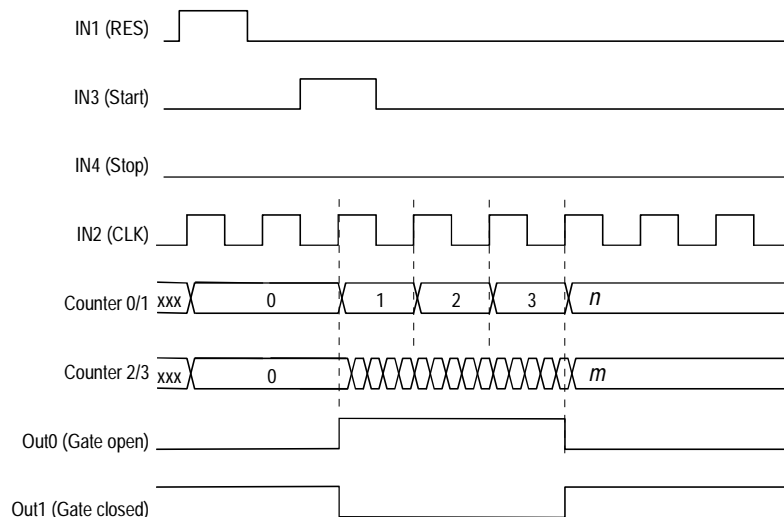
When the measurement has been completed you may calculate the mean period as follows:

$$Period = \frac{n}{fr \cdot m}$$

- where *fr*: reference frequency (supplied in DE7 with control bit 7)
- m*: contents of counter 2/3 (reference clock pulse count)
- n*: number of CLK pulses in counter 0/1 (corresponds to COMPARE, provided it was not terminated prematurely by Stop)



**Timing diagram:**



**Mode 20  
pulse  
measurements,  
pulse down**

**prog. time base with direction control**

The pulse width of a signal that is applied to the PULSE input is determined by means of an internal time base. The measurement is started by the falling edge of the input signal and ends with the rising edge. The rising edge of the measured signal stores the resulting pulse width in units of 1/Fref, that may be retrieved again.

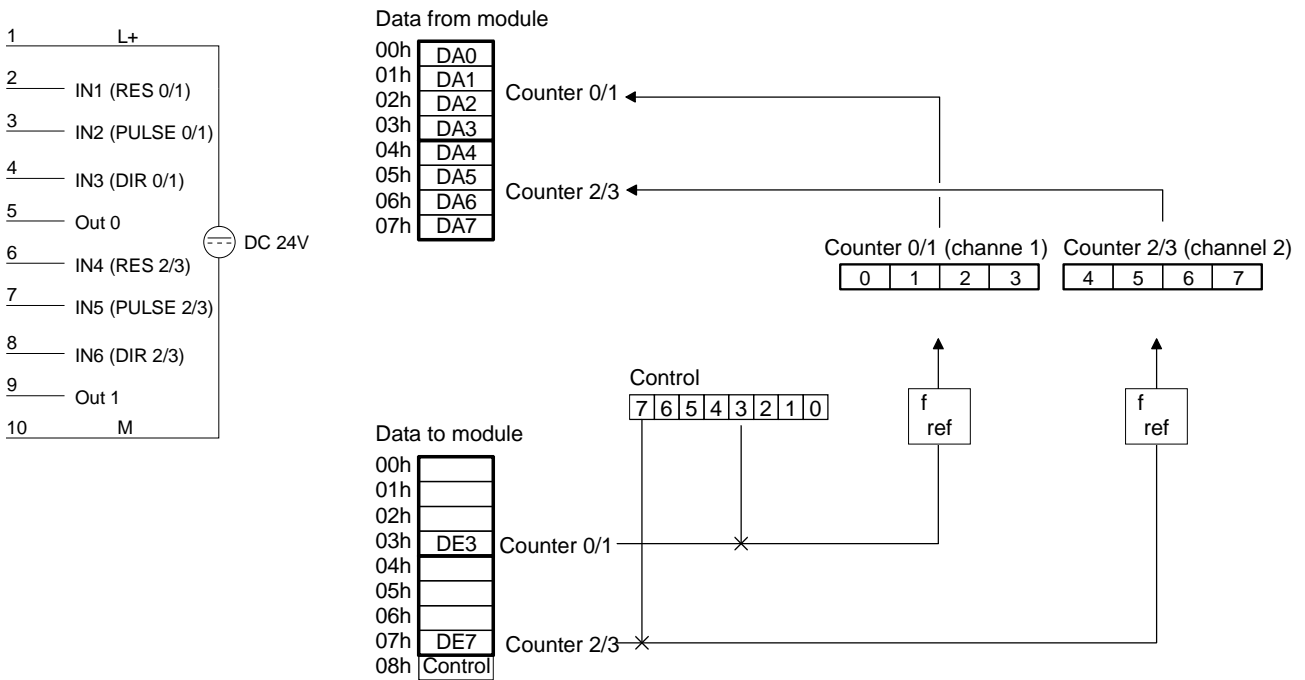
Input DIR controls the direction of the count. When DIR is held at a LOW level the counter counts up. When DIR is at a HIGH level the counter counts down.

RES has to be held at LOW during the counting operation. A HIGH level clears the counter.

Fref is programmable.

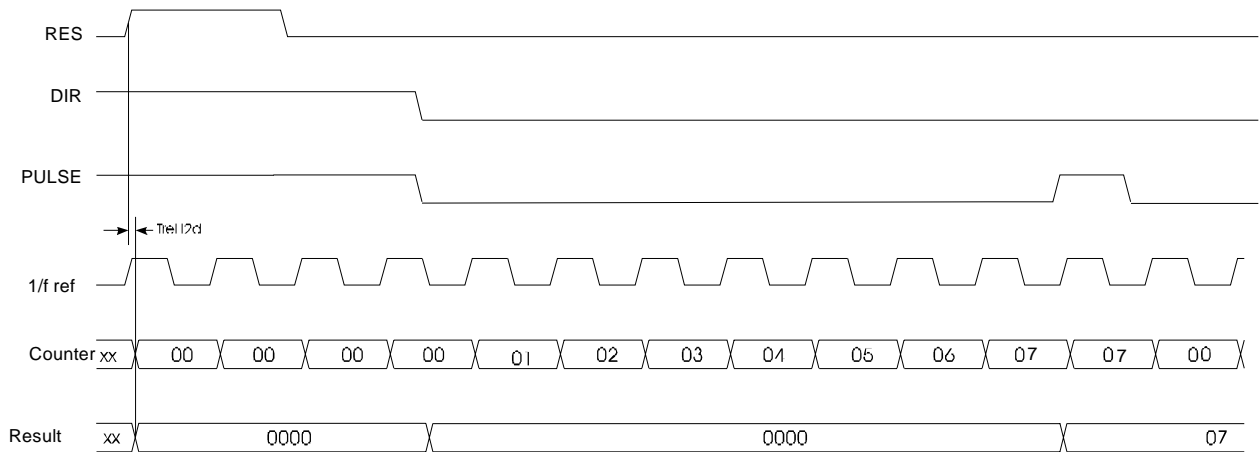
The OUT signal is not changed.

**Pin assignment  
access to counter**



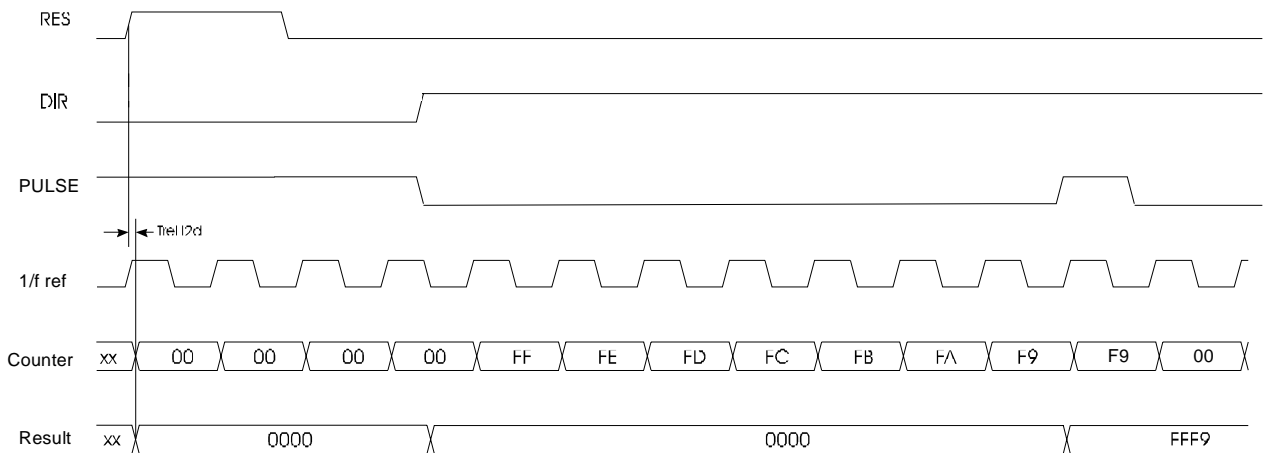
**Up counter**

The RES signal (R0) and the DIR signal (D0) are set to LOW. Subsequently the measurement is started with the falling edge of PULSE (C0) and the counter counts up in accordance with the selected time base. A rising edge at PULSE (C0) terminates the counting operation and the accumulated count is transferred into the result register. The result register is available to the PLC. The value remains in the result register until a new measurement has been completed and the register is changed by the new result.



**Down counter**

The RES signal (R0) is set to LOW and the DIR signal (D0) to HIGH. Subsequently the measurement is started with the falling edge of PULSE (C0) and the counter counts down in accordance with the selected time base. A rising edge at PULSE (C0) terminates the counting operation and the accumulated count is transferred into the result register. The result register is available to the PLC. The value remains in the result register until a new measurement has been completed and the register is changed by the new result.



**Mode 21  
pulse width  
measurement,  
pulse LOW**

**Direction up, prog. time base, with release**

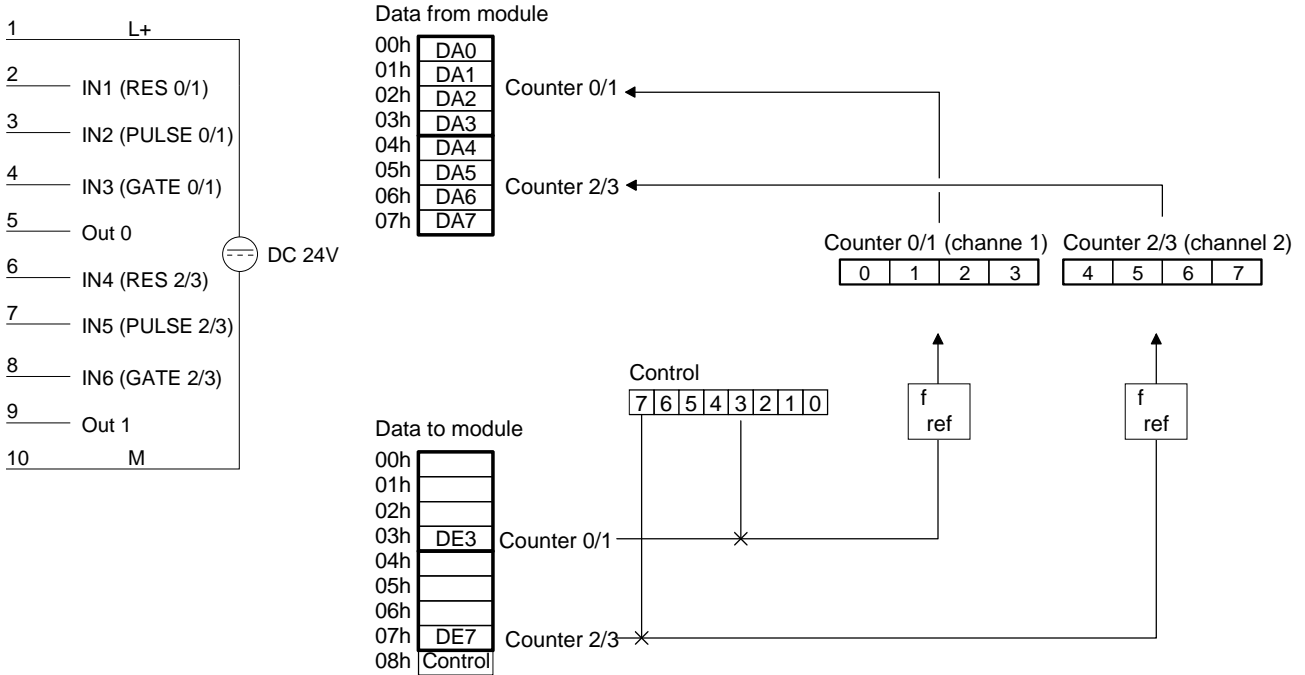
The pulse width of a signal applied to the PULSE input is determined by means of a programmable time base (fref). The measurement starts with the falling edge of the input signal and it is stopped by the rising edge of the input signal. The rising edge of the input signal saves the resulting pulse width in units of 1/fref. This is available to other devices.

A condition for the function is that a HIGH level is applied to the GATE input.

Input RES must be at a LOW level. A HIGH level at this input would clear the counter.

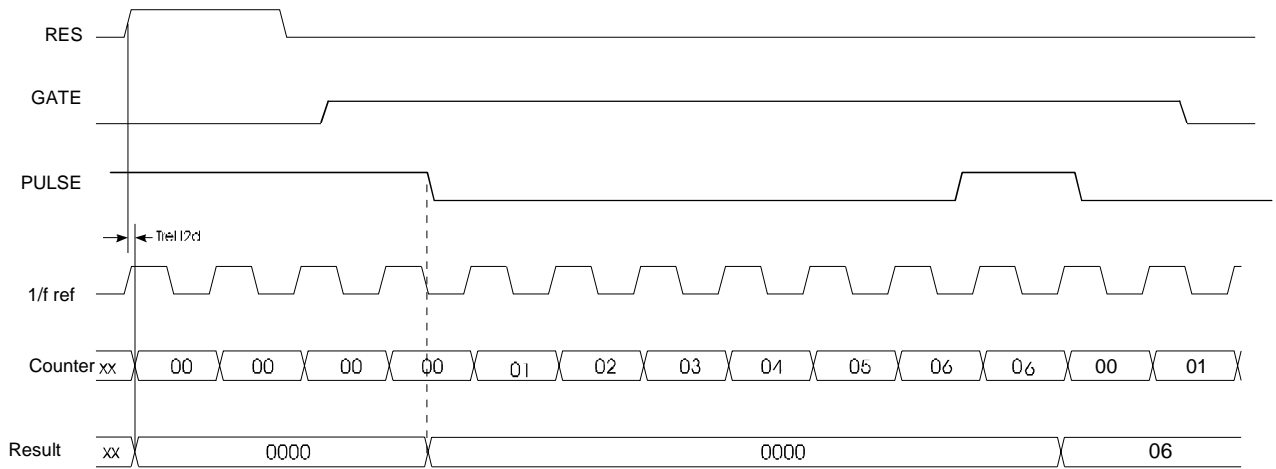
The OUT signal is not modified.

**Pin assignment  
access to counter**



**Up counter**

The RES signal (R0) is set to zero. The measurement can only be started when the GATE signal is at a HIGH level. The measurement is started with the falling edge of PULSE (C0) and the counter counts up in accordance with the selected time base. A rising edge at PULSE (C0) terminates the counting operation and the accumulated count is transferred into the result register. The result register is available to the PLC. The value remains in the result register until a new measurement has been completed and the register is changed by the new result. The GATE signal must be held at a HIGH level for the entire cycle, since the measurement could otherwise not be completed.



**Mode 22**  
**pulse width**  
**measurement,**  
**pulse HIGH**

**Direction down, prog. time base, with release**

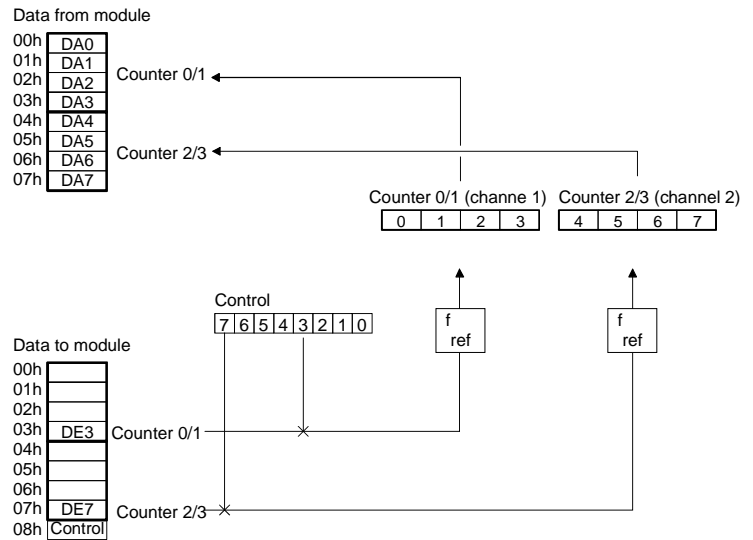
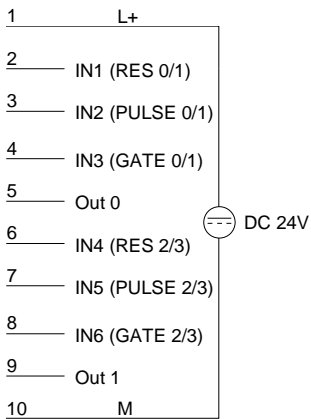
The pulse width of a signal applied to the PULSE input is determined by means of a programmable time base (fref). The rising edge of the input signal saves the resulting pulse width in units of 1/fref. This is available to other devices.

A condition for the function is that a HIGH level is applied to the GATE.

Input RES must be at a LOW level. A HIGH level at this input would clear the counter.

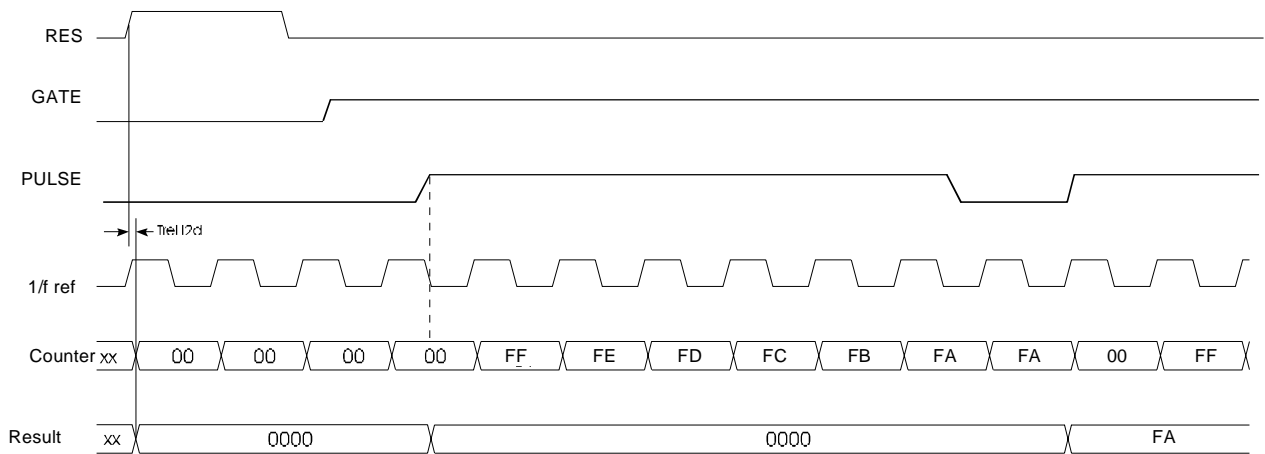
The OUT signal is not modified.

**Pin assignment**  
**access to counter**



**Down counter**

The RES signal (R0) is set to zero. The measurement can only be started when the GATE signal is at a HIGH level. The measurement is started with the rising edge of PULSE (C0) and the counter counts down in accordance with the selected time-base. A falling edge at PULSE (C0) terminates the counting operation and the accumulated count is transferred into the result register. The result register is available to the PLC. The value remains in the result register until a new measurement has been completed and the register is changed by the new result. A condition for the function is that a HIGH level is applied to the GATE input.

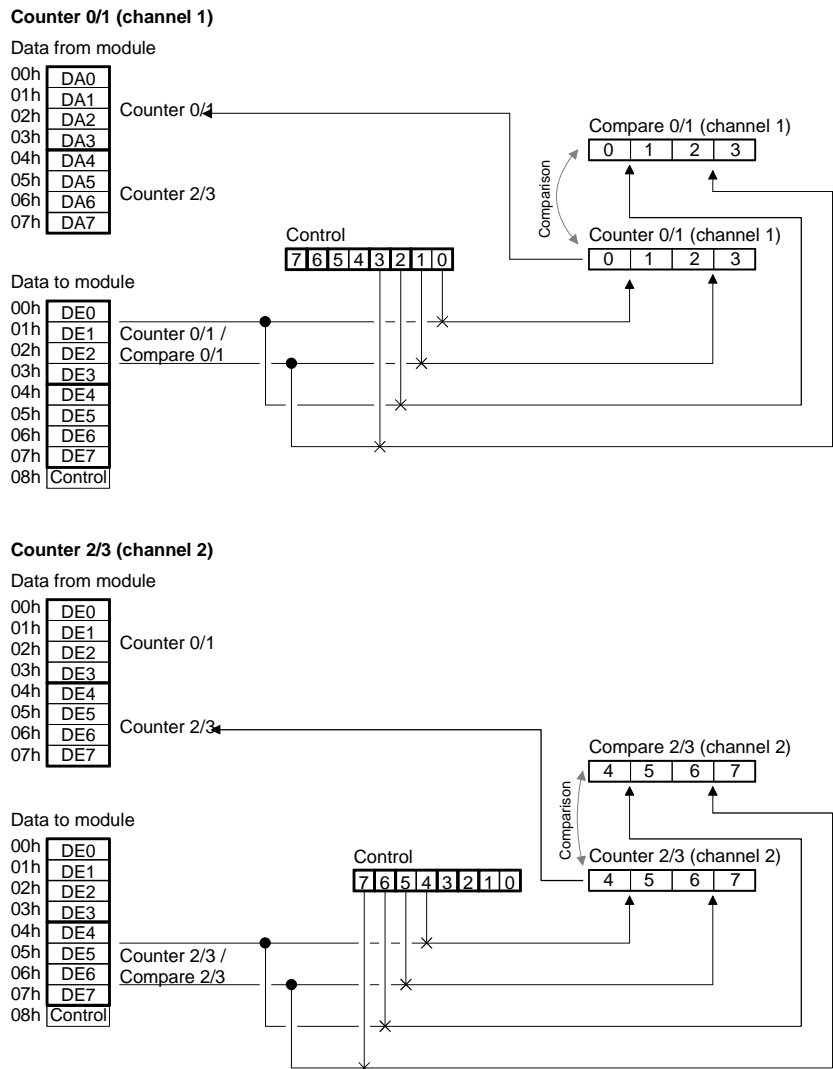
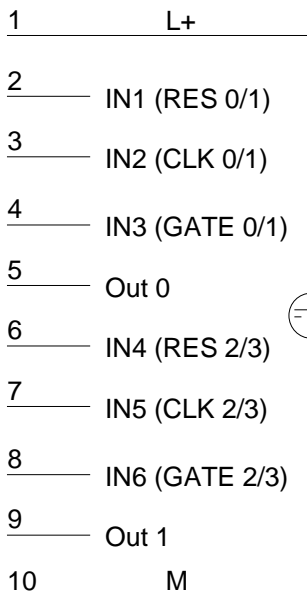


**Mode 23**  
**One Shot, count up, with release, output signal**

In mode 23 you may implement one 32Bit counter per channel, each one controlled by a GATE signal. Every rising edge of the input clock increments the counter by 1 as long as the signal applied to GATE is HIGH. RES must be at a LOW level. A HIGH level at this input would clear the counter. OUT changes to HIGH when the counter is loaded. OUT is cleared when the value entered into COMPARE is reached. The counter will continue the count operation after the value in COMPARE was reached.

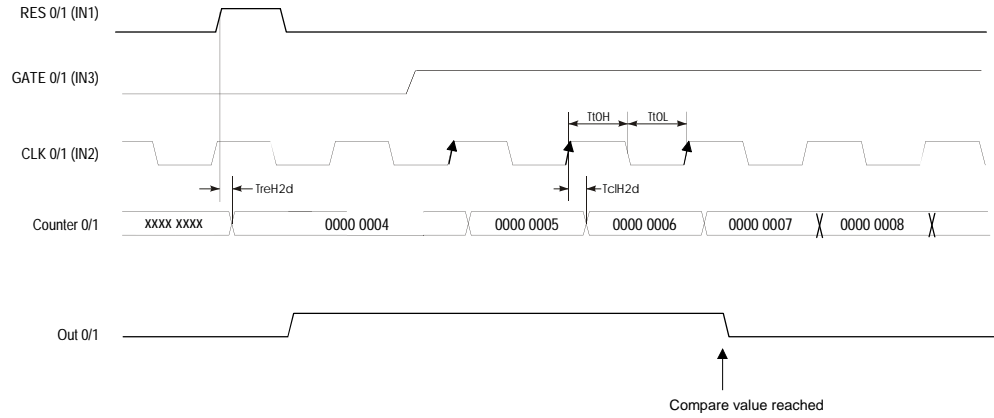
**Mode 23 - One Shot, up with Gate input, Output set**

**Pin assignment**  
**access to counter**





**Timing diagram** Example of counter 0/1 in mode 23:



1. The RES signal changes to LOW.
2. COMPARE is loaded once.
3. Counter (subject to Control) is loaded with, e.g. 0004.
4. The GATE signal is active.

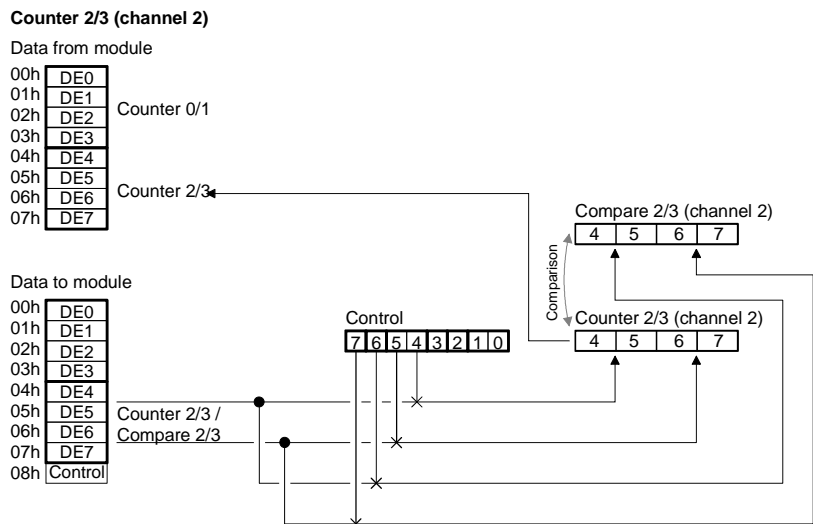
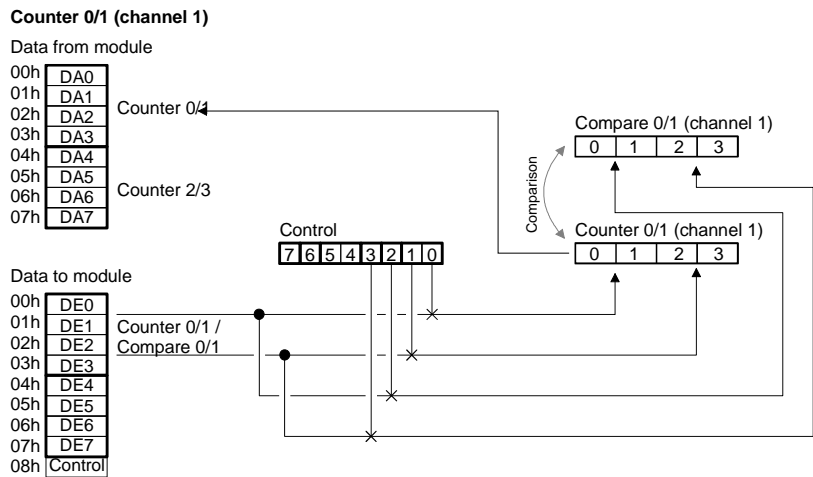
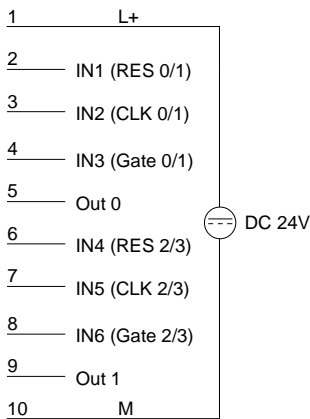
Stop by means of CONTROL = termination

**Mode 24**  
**One Shot, count down, with gate, output signal**

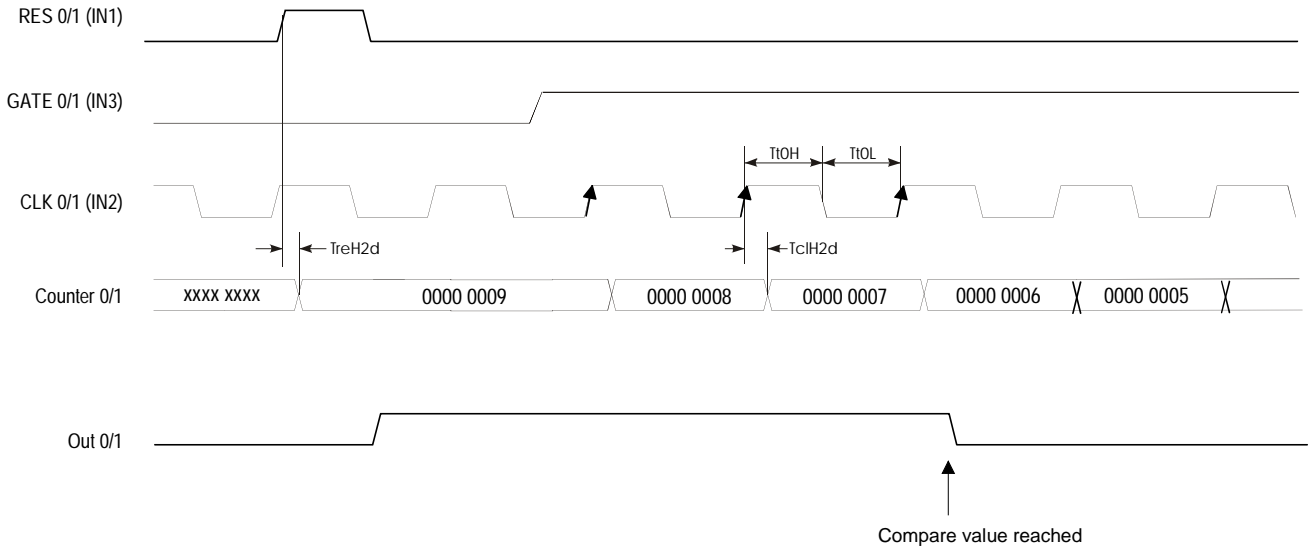
In mode 24 you may implement one 32Bit counter per channel, each one controlled by the signal applied to the GATE input. Every rising edge of the input clock decrements the counter by 1 as long as the signal applied to GATE is HIGH. RES must be at a LOW level. A HIGH level at this input would clear the counter. OUT changes to HIGH when the counter is loaded. OUT is cleared when the value entered into COMPARE is reached. The counter will continue the count operation after the value in COMPARE was reached.

**Mode 24 - One Shot, down with Gate-Input, Output set**

**Pin assignment**  
**access to counter**



**Timing diagram** Example of counter 0/1 in mode 24:



1. The RES signal changes to LOW.
2. COMPARE is loaded once.
3. Counter (subject to Control) is loaded with, e.g. 0009.
4. The GATE signal is active.

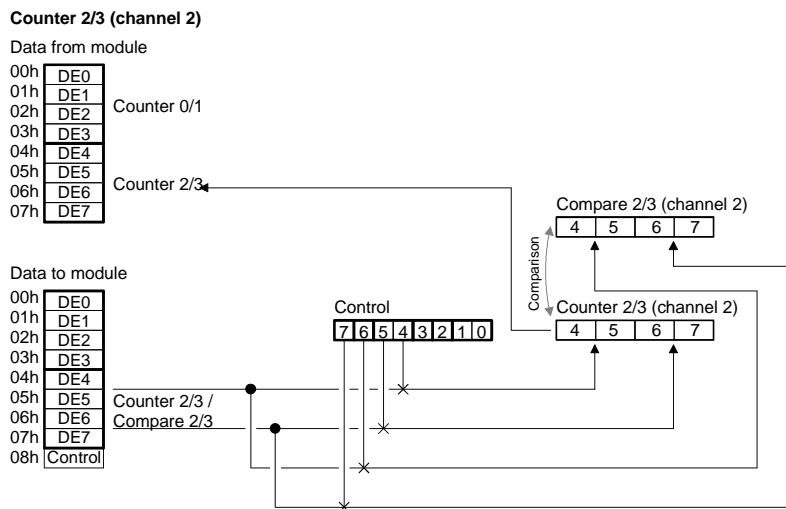
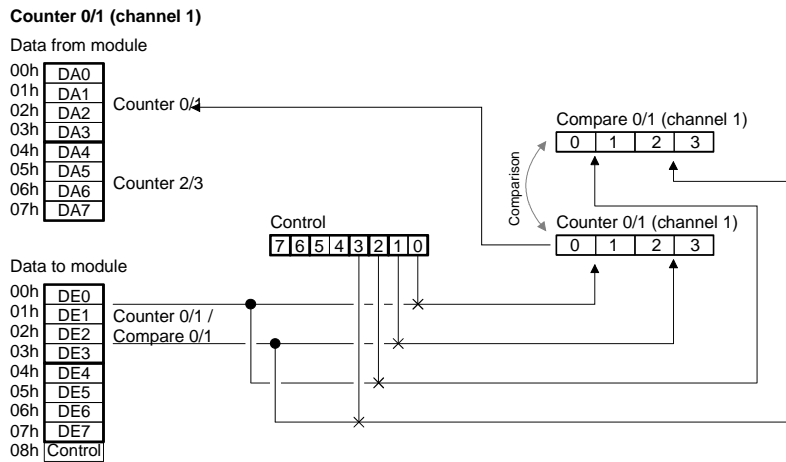
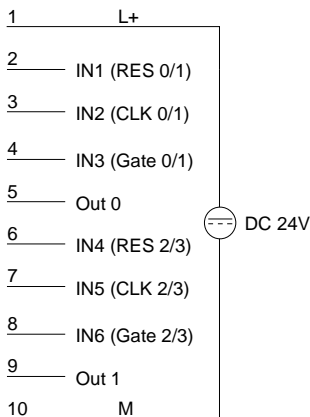
Stop by means of CONTROL = termination

**Mode 25**  
**One Shot, count up, with reset signal**

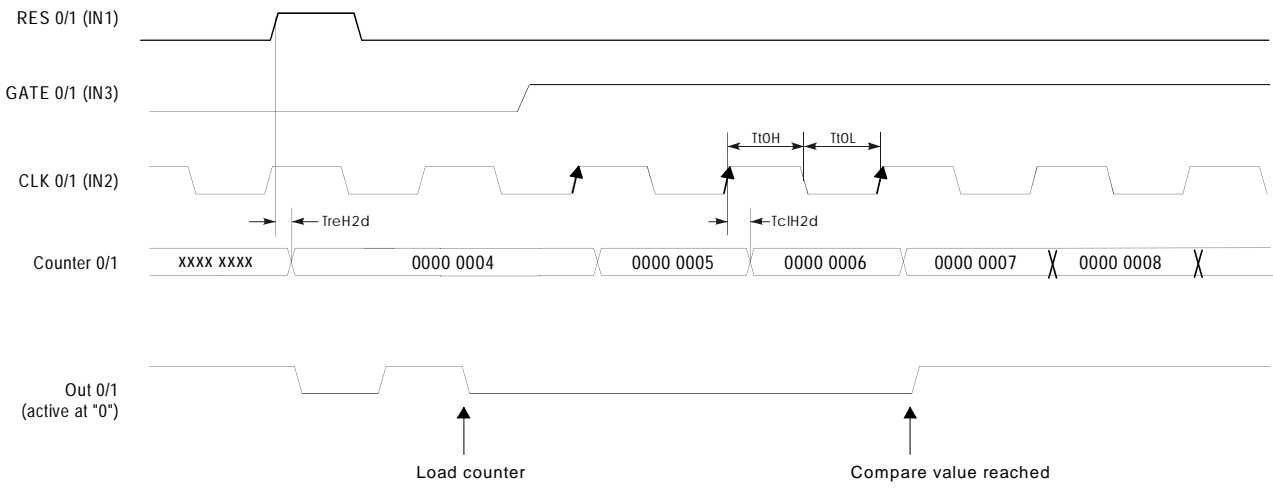
In mode 25 you may implement one 32Bit counter per channel, each one controlled by the signal applied to the GATE input. Every rising edge of the input clock increments the counter by 1 as long as the signal applied to GATE is HIGH. RES must be at a LOW level. A HIGH level at this input would clear the counter. OUT (active with 0) changes to LOW when the counter is loaded. OUT becomes HIGH when the value entered into COMPARE is reached.

**Mode 25 One Shot, count up, Reset**

**Pin assignment access to counter**



**Timing diagram** Example of counter 0/1 in mode 25:

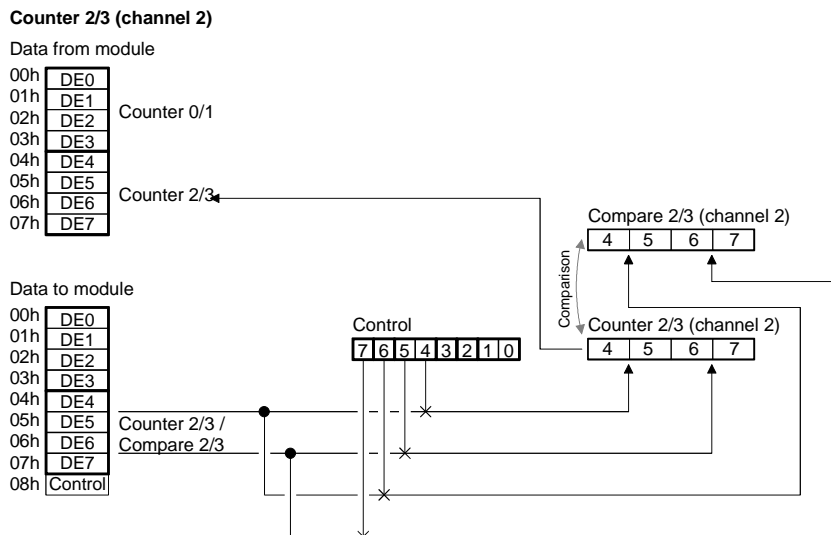
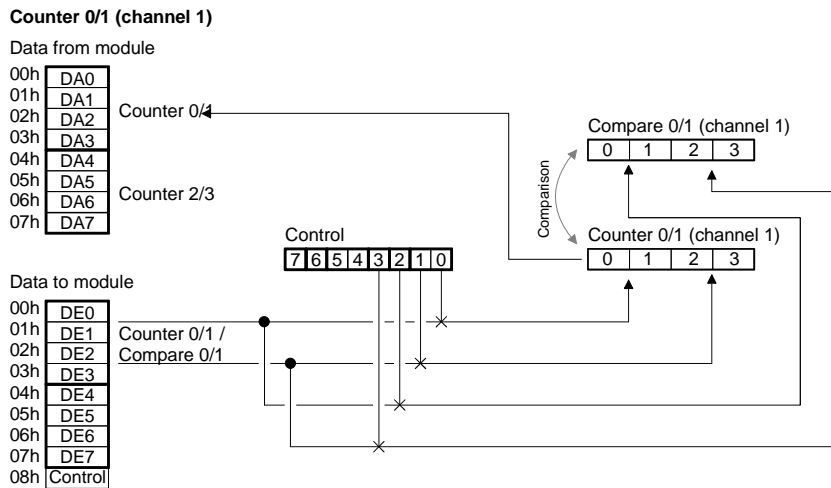
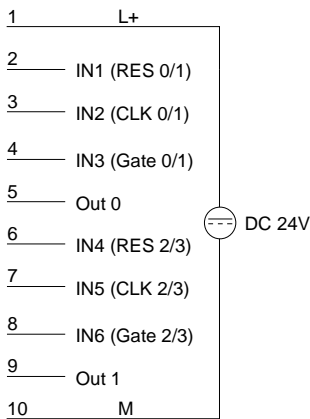


**Mode 26**  
**One Shot, count down, with reset signal**

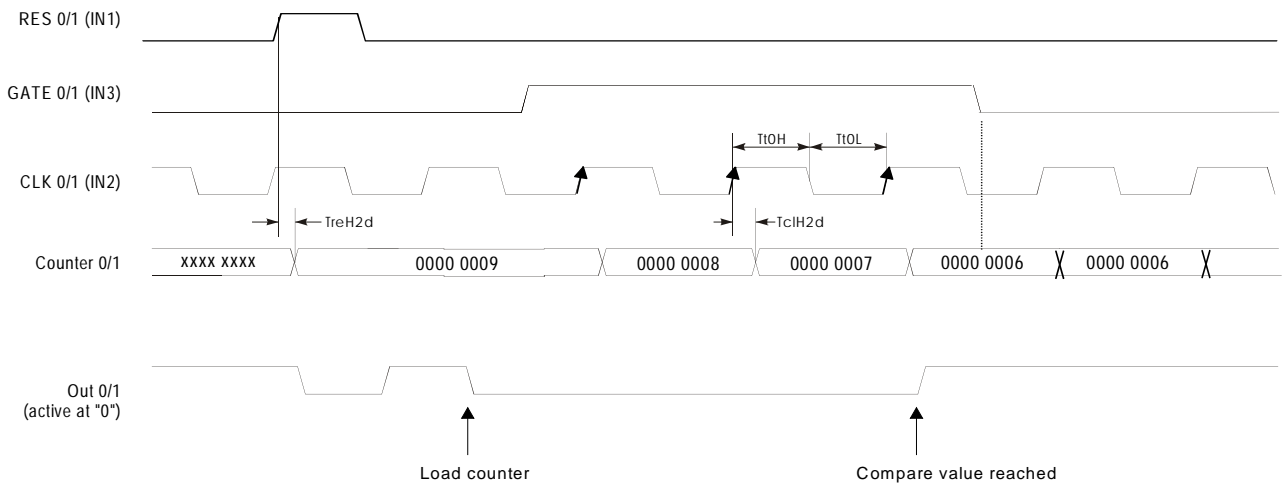
In mode 26 you may implement one 32Bit counter per channel, each one controlled by the signal applied to the GATE input. Every rising edge of the input clock decrements the counter by 1 as long as the signal applied to GATE is HIGH. RES must be at a LOW level. A HIGH level at this input would clear the counter. OUT (active 0) changes to LOW when the counter is loaded. OUT becomes HIGH when the value entered into COMPARE is reached.

**Mode 26 - One Shot, down, Reset**

**Pin assignment access to counter**



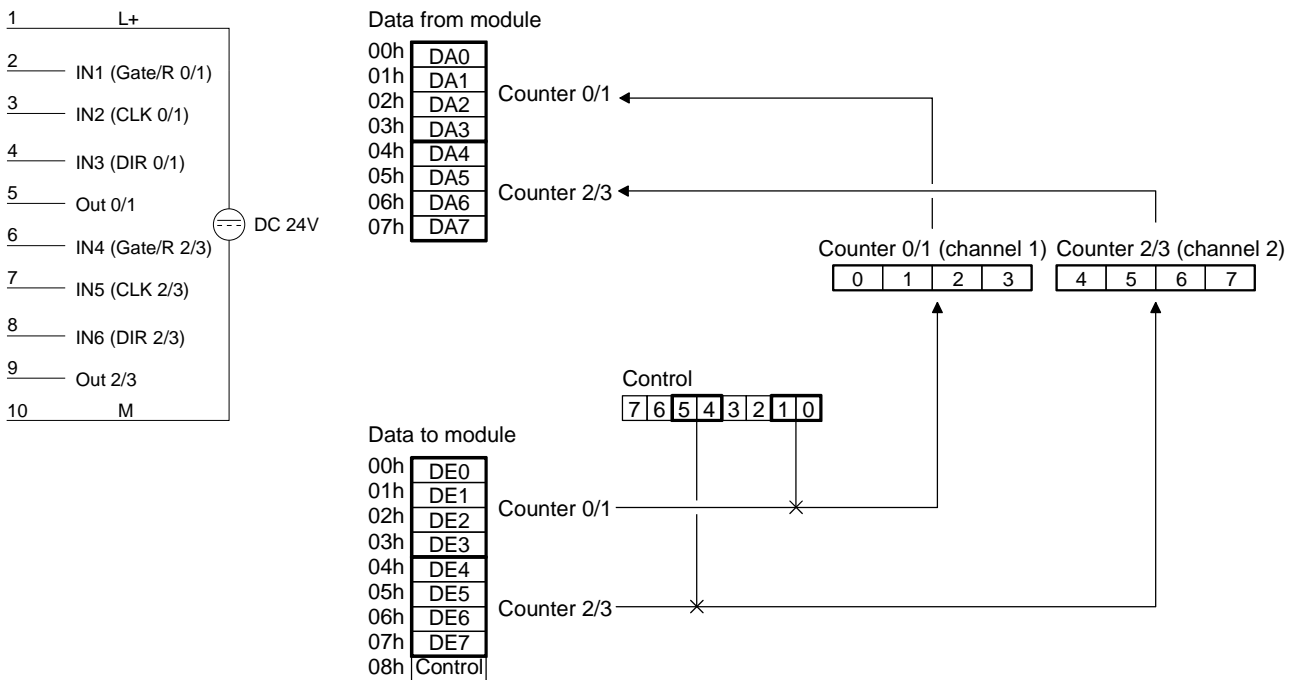
**Timing diagram** Example of counter 0/1 in mode 26:



**Mode 27**  
**32Bit counter**

In mode 27 two counters (16Bit) are combined to produce a 32Bit counter. You determine the direction by means of the DIR input (IN3 or IN6). Every rising or falling edge of the input clock signal increments or decrements the counter. The rising edge of the signal Gate/R resets the counter. During the count process, the signal Gate/R has to be HIGH. When the signal Gate/R becomes "0", the counter value remains valid. When the counter reaches zero, output OUT of the respective counter is active for a minimum period of 100ms, even if the counter should continue counting. If the counter stops at zero, the output remains active.

**Pin assignment**  
**access to counter**

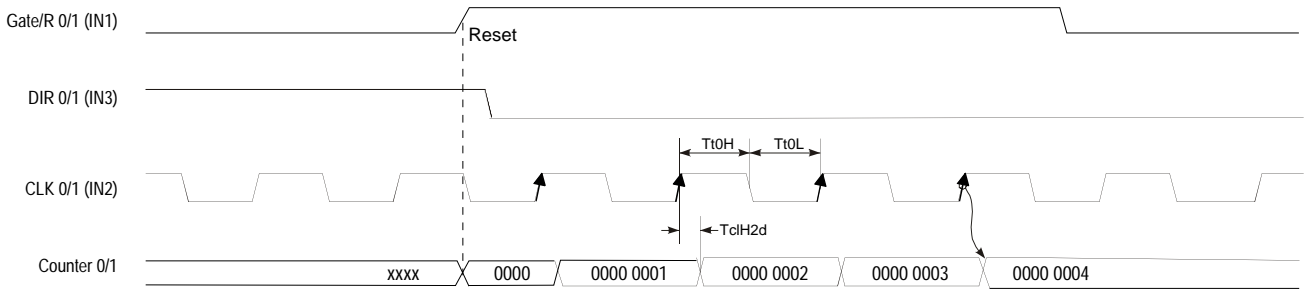




**Up counter**

In mode 27, a LOW level at the DIR input configures the counter for counting up.

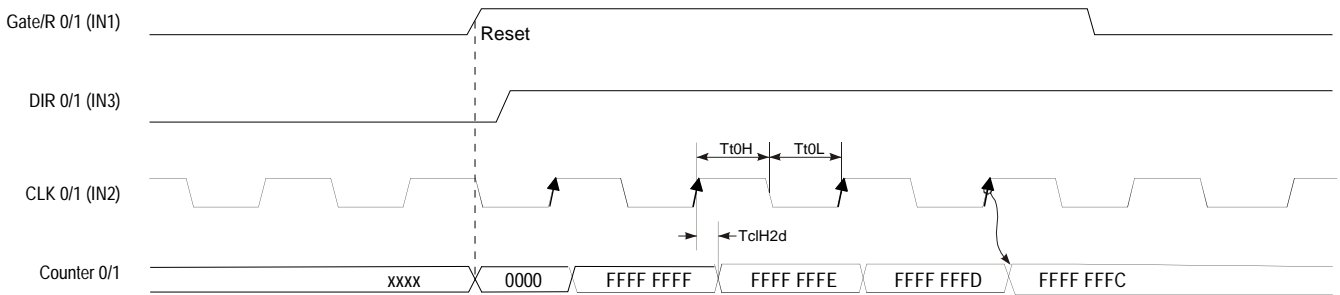
Timing diagram of the counter 0/1 example:



**Down counter**

In mode 27, a HIGH level at the DIR input configures the counter for counting down.

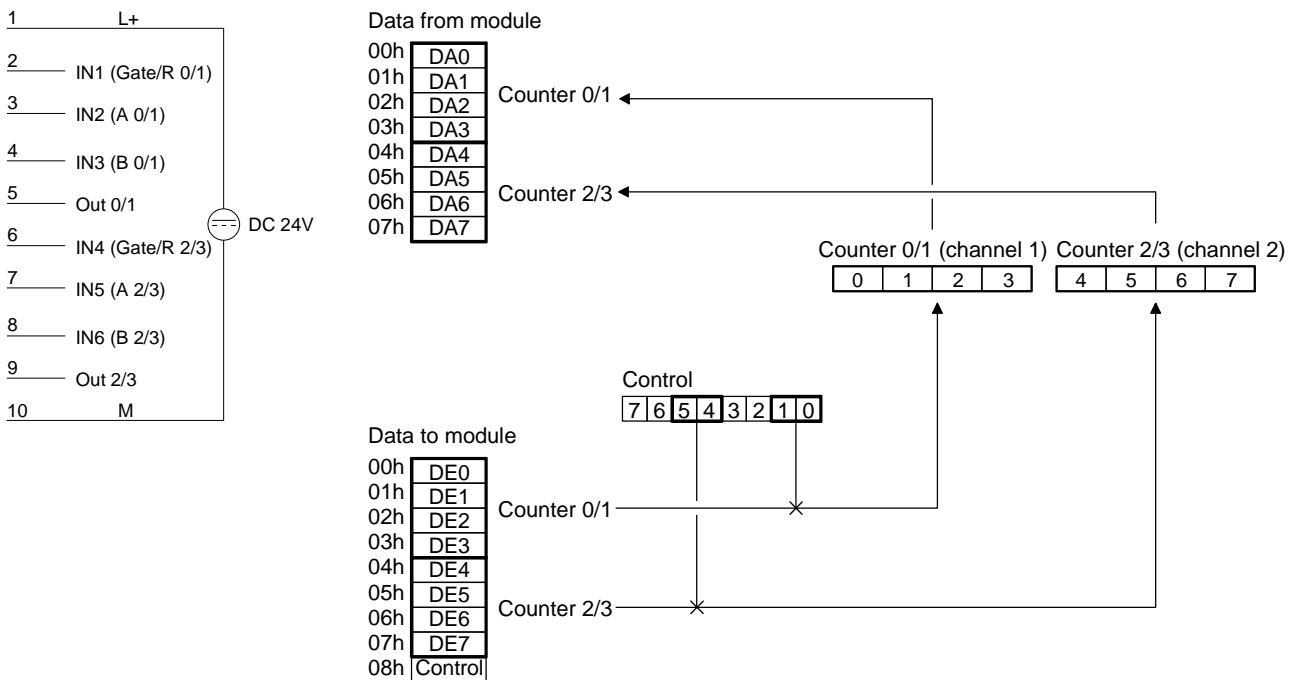
Timing diagram of the counter 0/1 example:



**Mode 28  
Encoder 1 edge**

In mode 28 you may configure an encoder for one of the channels. Depending on the direction of rotation this encoder will increment or decrement the internal counter with every falling edge. The rising edge of the signal Gate/R resets the counter. During the count process, the signal Gate/R has to be HIGH. When the signal Gate/R becomes "0", the counter value remains valid. When the counter reaches zero, output OUT of the respective counter is active for a minimum period of 100ms, even if the counter continues counting. If the counter stops at zero the output remains active.

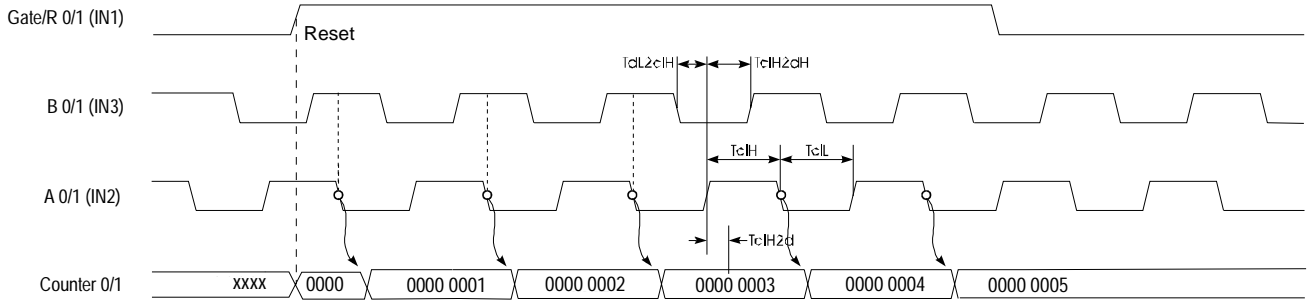
**Pin assignment  
access to counter**



**Up counter**

Every falling edge of the signal at input A increments the counter if input B is at HIGH level at this moment.

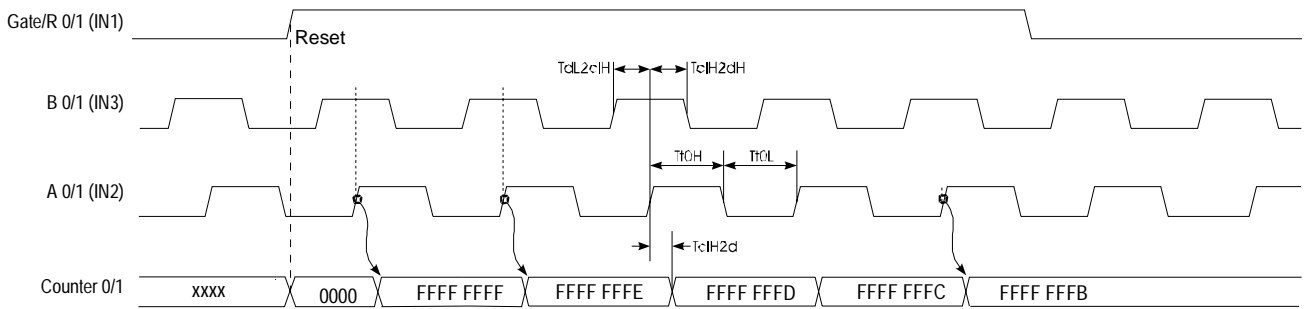
Timing diagram for the counter 0/1 example:



**Down counter**

Every rising edge of the signal at input A decrements the internal counter if input B is at HIGH level at this moment.

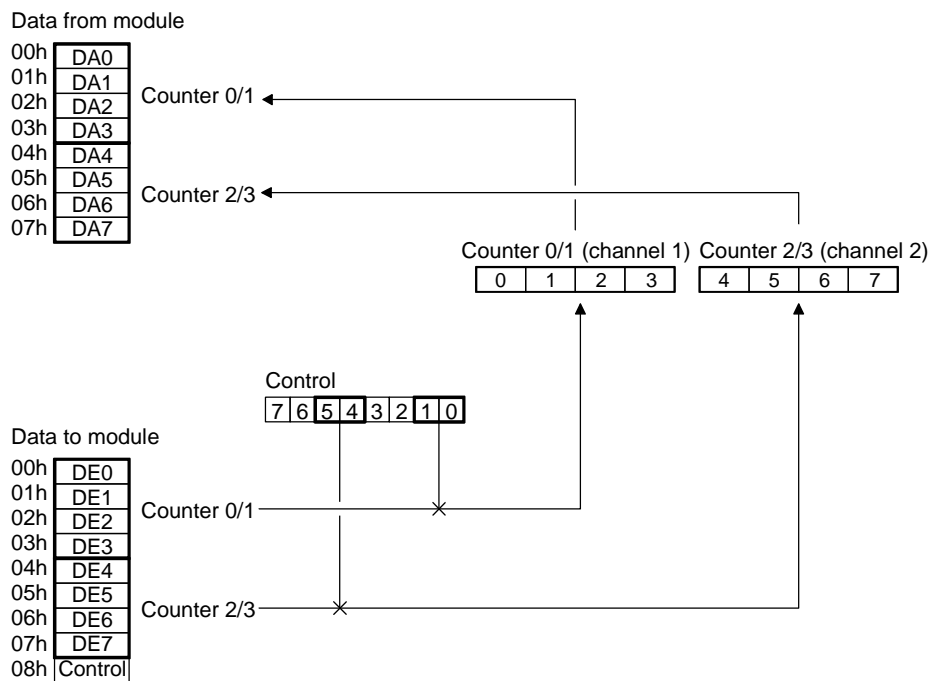
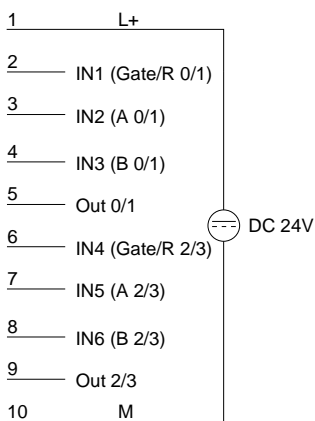
Timing diagram for the counter 0/1 example:



**Mode 29**  
**Encoder 2 edges**

Every rising or falling edge of the signal at input A changes the counter by 1. The direction of the count depends on the level of the signal applied to input B. The rising edge of the signal Gate/R resets the counter. During the count process, the signal Gate/R has to be HIGH. When the signal Gate/R becomes "0", the counter value remains valid. When the counter reaches zero, output OUT of the respective counter is active for a minimum period of 100ms, even if the counter continues counting. If the counter stops at zero the output remains active.

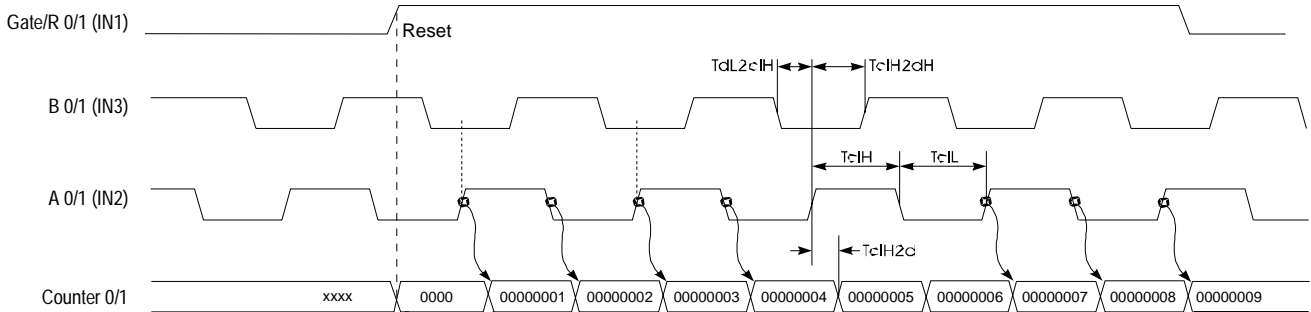
**Pin assignment**  
**access to counter**



**Up counter**

The counter is incremented by the rising edge of signal A if input B is at a LOW level or by the falling edge of input A when input B is at a HIGH level.

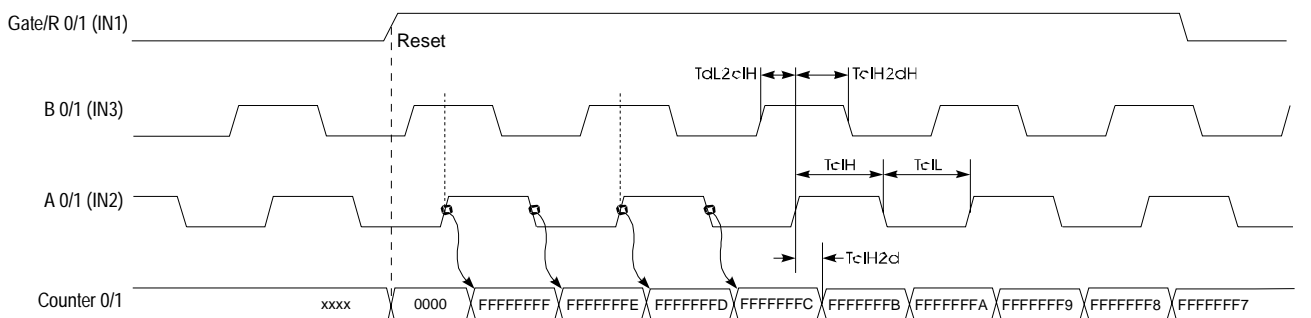
Timing diagram for the counter 0/1 example:



**Down-counter**

The counter is decremented by the rising edge of signal A if input B is at a HIGH level or by the falling edge of input A when input B is at a LOW level.

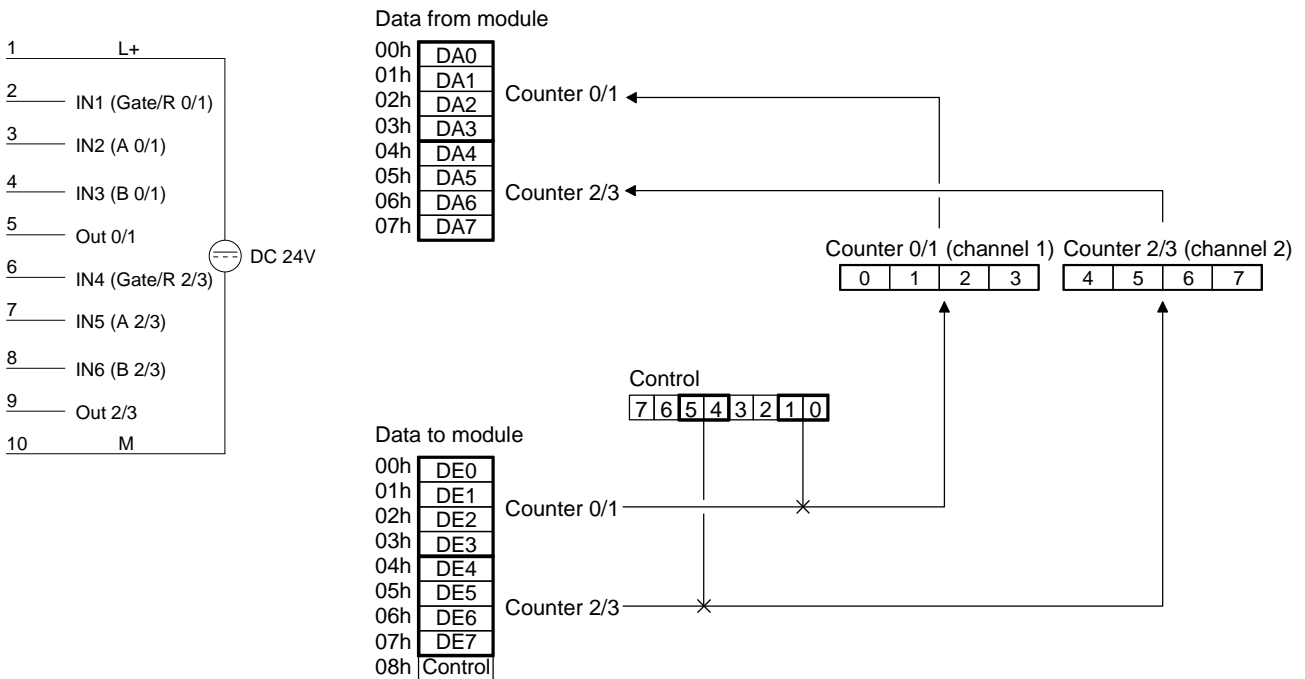
Timing diagram for the counter 0/1 example:



**Mode 30  
Encoder 4 edges**

Every rising or falling edge at inputs A or B increments or decrements the counter. The direction depends on the level applied to the other input (B or A). The rising edge of the signal Gate/R resets the counter. During the count process, the signal Gate/R has to be HIGH. When the signal Gate/R becomes "0", the counter value remains valid. When the counter reaches zero, output OUT of the respective counter is active for a minimum period of 100ms, even if the counter continues counting. If the counter stops at zero, the output remains active.

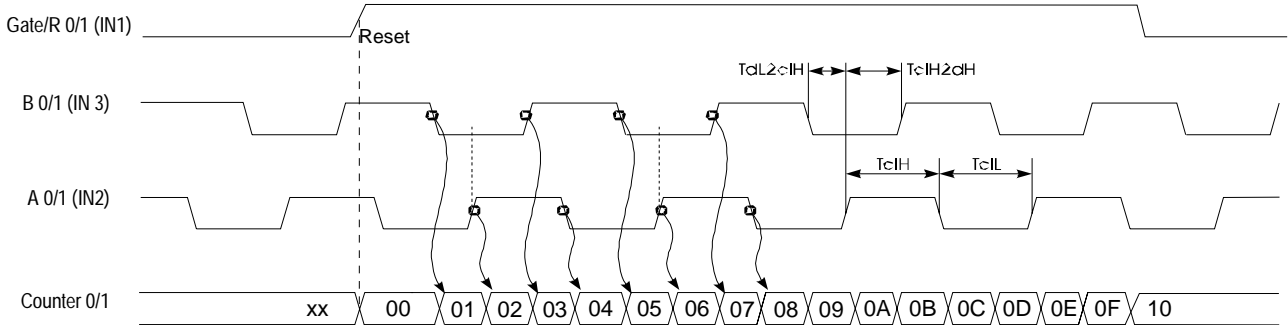
**Pin assignment  
access to counter**



**Up counter**

The counter is incremented when a rising edge is applied to B while input A is at a HIGH level or if a falling edge is applied to B when input A is at a LOW level. Alternatively it is also incremented when a rising edge is applied to A when input B is at a LOW level or by a falling edge at A when input B is at a HIGH level.

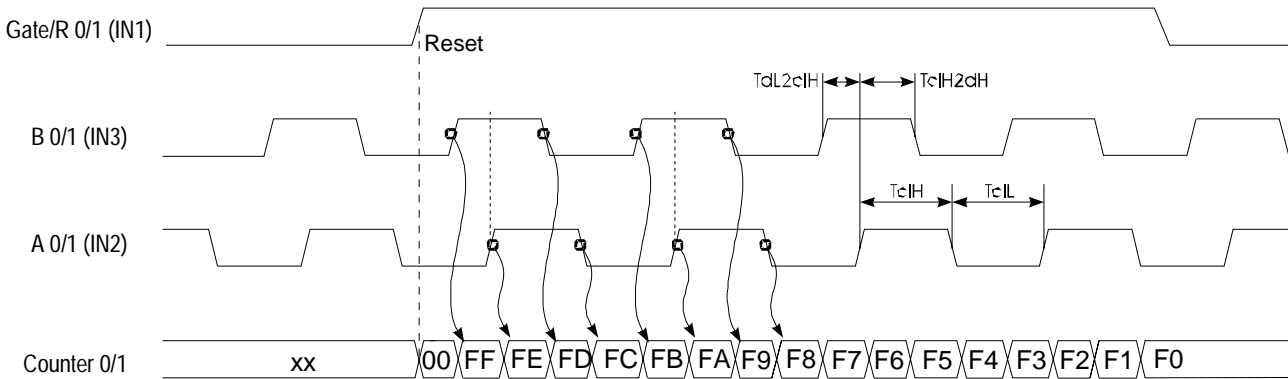
Timing diagram for the counter 0/1 example:



**Down counter**

The counter is decremented when a rising edge is applied to B while input A is at a LOW level or if a falling edge is applied to B when input A is at a HIGH level. Alternatively it is also decremented when a rising edge is applied to A when input B is at a HIGH level or by a falling edge at input A when input B is at a LOW level.

Timing diagram for counter 0/1 example:



## Technical data

### SSI module FM 250S

Electrical data	VIPA 250-1BS00
Number of channels	1
Number of outputs	2
Current consumption	200mA via backplane bus
Isolation	yes
SSI interface	Transducer supply voltage
Signal cable	RS422, isolated
Clock	RS422, isolated
Baudrate	configurable: 100 / 300 / 600kBaud (default: 300kBaud)
Signal voltage "0"	-5 ... 7V
Signal voltage "1"	13 ... 36V
Output stage	24V DC high side switch 0.5A
Ext. power supply	24V DC (18 ... 28.8V)
Status indicator	by means of LEDs located on the front
Programming specifications	
Input data	4Byte
Output data	4Byte, 8Byte buffer in the module
Parameter data	4Byte
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD)	25.4x76x76mm
Weight	100g



**Counter module**  
**FM 250**

Electrical data	VIPA 250-1BA00
Number of counters	2 resp. 4
Counter resolution	32Bit resp. 16Bit
Number of operating modes	26
Counter frequency	max. 1MHz
Current consumption	80mA via backplane bus
Isolation	yes
Output stage	DC 24V high side switch 0.5A
Ext. power supply	DC 24V (18 ... 28.8V)
Signal voltage "0"	-30 ... 5V
Signal voltage "1"	13 ... 30V
Status indicator	via LEDs located on the front
Programming specifications	
Input data	10Byte
Output data	10Byte
Parameter data	2Byte
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD)	25.4x76x76mm
Weight	100g



## Chapter 11 MotionControl Modules

### Outline

This chapter contains information about the installation, the data transfer and the operating modes of the MotionControl modules.

The following text describes:

- Installation
- Parameterization
- Data transfer
- Technical data

### Content

Topic	Page
<b>Chapter 11 MotionControl Modules</b> .....	<b>11-1</b>
System Overview.....	11-2
FM 253 - MotionControl Stepper.....	11-3
FM 253 - MotionControl Stepper - Construction.....	11-4
FM 253 - Connecting a drive.....	11-6
FM 253 - Data transfer >> FM 253.....	11-8
FM 253 - Parameterization .....	11-9
FM 253 - Operating modes.....	11-11
FM 253 - Data transfer >> CPU .....	11-15
FM 253 - Handling blocks .....	11-17
FM 254 - MotionControl Servo .....	11-23
FM 254 - MotionControl Servo - Construction.....	11-24
FM 254 - Connecting a drive with encoder.....	11-26
FM 254 - Summary of parameters and transfer values .....	11-28
FM 254 - Parameterization .....	11-29
FM 254 - Data transfer >> FM 254.....	11-30
FM 254 - Operating modes.....	11-31
FM 254 - Data transfer >> CPU .....	11-37
Technical data .....	11-38

## System Overview

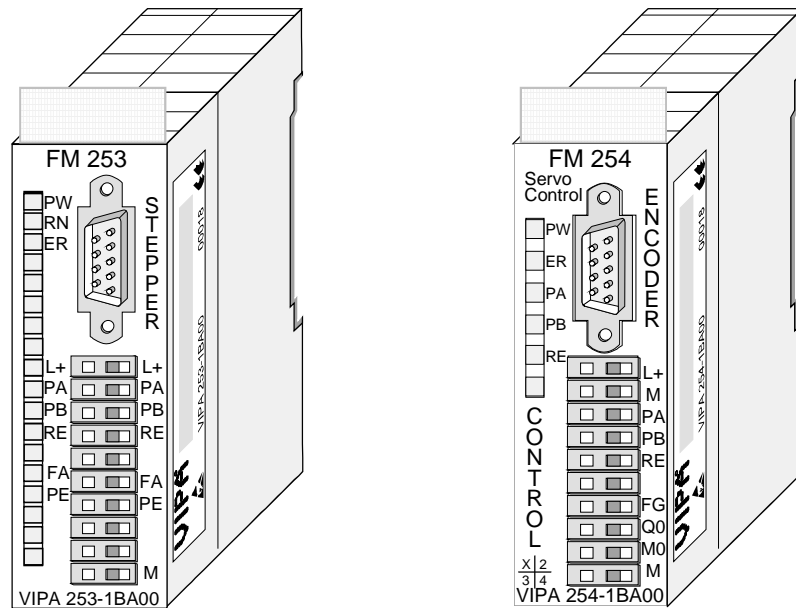
### General

The MotionControl modules described here are modules for machine drives with a high pulse rate.

The modules may be used for point-to-point positioning as well as for complex drive outlines with need for a high level of precision, dynamics and speed.

Depending on the module you may control stepper motors or servo drives.

### MotionControl modules



### Order data

Type	Order number	Description	Page
FM 253	VIPA 253-1BA00	MotionControl Module Stepper	11-3
FM 254	VIPA 254-1BA00	MotionControl Module Servo	11-23

## FM 253 - MotionControl Stepper

### Properties

The FM 253 is a positioning module for controlling a stepper motor.

The module works independently and is controlled via an according user application at the CPU.

The module has the following characteristics:

- Microprocessor controlled positioning module for controlling a 1axis drive with stepper motor.
- Operating round and linear axis
- Different operating modes
- The module works independently and is controlled via an user application at the System 200V.
- The parameterization data is stored in the internal Flash memory. There is no battery required.
- The module contains 3 inputs for connecting end switches and is able to control 2 outputs. The states of the in-/outputs are additionally shown via LEDs.

### Application areas

The module may be employed for simple positioning tasks as well as for complex drive outlines with a need for a high level of precision at the positioning.

Stepper motors are employed where a maximum torque at low rotational speed is required and the target position shall be reached and held without overshoot.

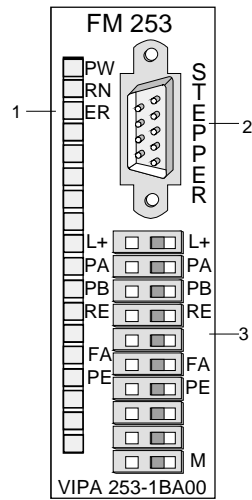
### Operating modes

The operating mode is preset via your application program. The module supports the following operating modes:

- Positioning operating absolute
- Positioning operating relative
- Reference run
- Permanent run axis
- Set position
- Edit motor parameters
- Delete errors
- Read inputs

## FM 253 - MotionControl Stepper - Construction

### Front view



- [1] LED Status monitor
- [2] Plug for drive
- [3] Connection for supply voltage, end switch and outputs

### Components

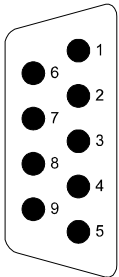
#### LEDs

The FM 253 has some LEDs at the front used for status monitoring. The usage and the according colors of these LEDs are shown in the following table:

Label	Color	Description
PW	Yellow	DC 24V supply voltage is applied
RN	Green	RUN: control active
ER	Red	Internal error
L+	Yellow	DC 24V supply voltage for outputs is applied
PA	Green	Limit value A overrun, input PA is set
PB	Green	Limit value B overrun, input PB is set
RE	Green	Reference point overrun
FA	Green	Drive in run
PE	Green	Drive reached position

**Stepper interface**

Via this interface your stepper motor is connected. The interface appears as 9pin D-type-plug and works with RS422 level. It has the following pin assignment:



*9pin D-type-plug*

Pin	Assignment
1	PULSE_P: (+) pulse output
2	DIR_P: (+) direction signal
3	ENABLE_P: (+) release signal
4	READY+: (+) readiness message
5	GND: ground
6	PULSE_N: (-) pulse output
7	DIR_N: (-) direction signal
8	ENABLE_N: (-) release signal
9	READY-: (-) readiness message

**Control interface**

The control interface provides connection possibilities for end switches and output elements. The interface has the following pin assignment:



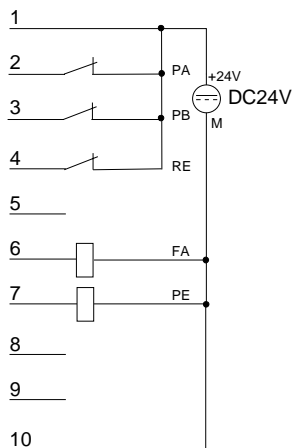
Pin	Assignment
1	Supply voltage DC 24V for outputs
2	Input: end switch PA
3	Input: end switch PB
4	Input: reference switch
5	reserved
6	Output: axis in motion
7	Output: position reached
8	reserved
9	reserved
10	Ground 24V

## FM 253 - Connecting a drive

### Connection stepper motor

The connection of a stepper motor is exclusively via the stepper interface.

### Connection of supply voltage, end switch and output units



### Voltage supply

The module itself is provided via the back plane bus. The deployment of the integrated digital outputs requires an additional voltage supply. The connection of an additional DC 24V supply voltage takes place via the clamps 1 and 10 of the control interface.

### Inputs for end switches

You may connect up to 3 end switches (opener) to the module.

At terminals 2 and 3 (PA and PB) you connect the end switches with which you limit the distance. As soon as one of these switches is operated, the drive is stopped immediately and may only be driven into the other direction.

Terminal 4 is for the connection of the reference switch which is responsible for the tuning with the FM 253 module.

### Outputs

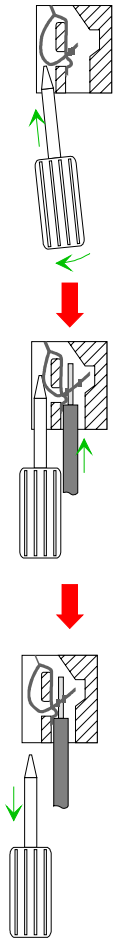
The module contains 2 outputs that are only controlled by the module:

- FA - drive in run (clamp 6)
- PE - drive reached position (clamp 7)

The states of the outputs are shown via the according LEDs.



## Cabling



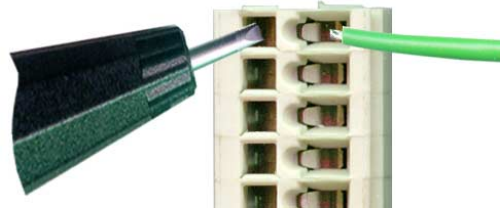
The end switches and the outputs are to connect at the control interface. Herefore a 10pin plug with CageClamp technology from WAGO is used. The cabling with CageClamps is very fast and in opposite to screw connections vibration secure.

You may connect cores with a core cross-section from  $0.08\text{mm}^2$  up to  $1.5\text{mm}^2$ .

The cabling is analog to the big CageClamps of the System 200V.

Push the spring in the square opening with a fitting screwdriver more inside and insert the core into the rectangular opening.

By releasing the screwdriver the core is securely fixed.



## FM 253 - Data transfer >> FM 253

### Drive data

The MotionControl Stepper module fetches a data block from the CPU cyclically and analyzes it.

The data block has a length of 16Byte and the following structure:

Byte No.	Content	Length
0-3	Scheduled position	4Byte
4-7	Scheduled frequency	4Byte
8-9	Reserved	
10	Mode	1Byte
11	Index	1Byte
12-15	Variable parameters	4Byte

Via the MODE-Byte the contents of the data block are specified. The following functions may be initiated via the MODE-Byte:

### Mode (Byte 10)

Bit 7 ... Bit 0	Preset in Byte	Response in Byte
00: Idle-Mode - no status change of the drive, serves for parameter changes	-	-
01: Positioning relative - driving the preset number of steps	0-3: rel. set position	-
02: Reference run - calibration of the drive		
03: Permanent run axis - drive runs with scheduled frequency	15: Parameter bits 4-7: set frequency	- -
04: Read inputs - responds with the end switches states	-	15: State
05: Motor parameters - transmits parameters depending on index	11: Index, 12-15: Parameter	-
06: Set position - sets the recent position in the module without moving the drive	0-3: Set position	-
07: Delete error - deletes the error bit activated with 1	14-15: Error bit	-
08: Positioning absolute - drive to scheduled position	0-3: abs. set position	-

### Parameter transfer (Mode = 05h)

Via **Index (Byte 11)** you set the parameter which value may be predefined via **Byte 12-15**. The value is transferred to the module by setting the **Mode 05h in Byte 10**.

More detailed information follows below.

## FM 253 - Parameterization

### Overview

The parameter data is transferred to the module together with the drive data in the 16Byte sized data block. For the parameterization you type the parameter to change in the **Index-Byte (Byte 11)** via the **Index-No.**

The new value is fixed in **Byte 12-15**.

As soon as you set the **Mode-Byte (Byte 10)** to **05h**, the parameter is transferred to the module.



Please regard, that new parameters are only taken over when there has been a mode change before. For this you switch into the IDLE-Mode (MODE-Byte 10 = 00h) after every parameter transfer.

### Store parameters in the Flash

The parameters that you transfer to the module are stored in the RAM. As long as the module is supplied with voltage, the parameters are preserved. Via the index no. 97h you also have the possibility to store the parameters in the internal Flash.

So the parameters are available again after PowerOn.

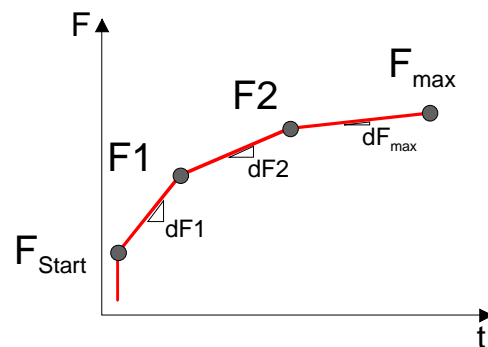
### Parameterization via FCs

You get FCs from VIPA that should make the deployment of the FM 253 easier. For example you may parameterize your module via the FCs 201 and 202.

The control of the drive functions via FC 200. Via this FC you may access all modes except "Set parameters".

### Context of the parameters

The following illustration shows the important contexts of the parameters. The assignment of the according index no. is to find in the table below.



**Set index at parameter**

Via the index no. you fix the parameter in Byte 11, where the value may be preset in Byte 12-15.

Index	Parameter	Unit	Value range	Default	Description
00h	Fstart	Hz	UINT32	200	Start frequency
01h	F1	Hz	UINT32	4000	Limit frequency 1
02h	dF1	Hz	UINT32	100	Acceleration of Fstart ⇒ F1
03h	F2	Hz	UINT32	10000	Limit frequency 2
04h	dF2	Hz	UINT32	60	Acceleration of F1 ⇒ F2
05h	Fmax	Hz	UINT32	30000	Maximum drive frequency
06h	dFmax	Hz	UINT32	40	Acceleration of F2 ⇒ Fmax
07h	Fpos	Hz	UINT32	30000	Frequency at positioning
08h	Fref	Hz	UINT32	1000	Frequency for reference run
10h	Fist	Hz	UINT32	-	Recent motor frequency (read only)
11h	Fsoll	Hz	UINT32	-	Recent set frequency (read only)
13h	FTarget	Hz	UINT32	-	Target frequency (read only)
97h				-	Store parameters in Flash
98h				-	Read parameters from Flash (State like after PowerON)
99h				-	Load default parameters

**Note!**

When setting parameters for the drive, you should remember the following rules:

- **dF1** should always be **smaller** than **Fstart**
- **dF2** should be the **half** of **dF1**
- **dFmax** should be the **half** of **dF2**

For this the following context appears:

$$4 \cdot dF_{\max} = 2 \cdot dF2 = dF1 < F_{\text{Start}}$$

Wrong inputs are partly corrected by the firmware of the module.

## FM 253 - Operating modes

### Overview

By setting according bits in the MODE-Byte you may set the following operating modes described below:

- IDLE-Mode
- Positioning relative / absolute
- Permanent run
- Set position
- Reference run

### IDLE-Mode

Default: Byte 10 = 00h

In the IDLE-Mode no state change of the drive occurs. For new data is only taken over by the module after an state change, you may initiate a mode change by jumping into the IDLE-Mode and back again.

Via the IDLE-Mode you may e.g. start a new order, for a mode change is recognized by the jump into the IDLE-Mode.

The operating mode IDLE should always be called when no action shall be initiated. For initiating an action you normally branch into another mode only for a short time and switch then back to the IDLE-Mode.

### Positioning relative

Default: Byte 10 = 01h, Byte 0-3 = relative set position

At the relative positioning a predefined number of steps is added to the recent position and then approached.

Herefore you have to predefine the position offset (number of steps) as relative scheduled position in Byte 0-3 and then set the **Mode (Byte10) to 01h.**

By setting the Byte 10 to 01h the relative positioning starts.

For acceleration and frequency of the drive, the values set in the parameters are used. If there are no presetting, the default values are used.

As long as the drive is operating, the output "**Axis in run**" is set. After reaching the position this output is cleared and the output "**Position reached**" is set.

### Positioning absolute

Default: Byte 10 = 08h, Byte 0-3 = absolute set position

At the absolute positioning an absolute scheduled position is approached.

Herefore you have to predefine the position (number of steps) as absolute scheduled position in Byte 0-3 and then set the **Mode (Byte 10) to 08h.**

By setting the Byte 10 to 08h the absolute positioning starts.

For acceleration and frequency of the drive, the values set in the parameters are used. If there are no presetting, the default values are used. As long as the drive is operating, the output "**Axis in run**" is set. After reaching the position this output is cleared and the output "**Position reached**" is set.

**Permanent run**

Default: Byte 10 = 03h, Byte 4-7 = Scheduled frequency

At permanent run the axis rotates with the set frequency until it is changed.

Herefore you have to predefine the rotational speed as set frequency in Byte 4-7 and then set **Mode (Byte10) to 03h**.

By setting Byte 10 to 03h the drive starts and rotates with the given frequency until a new frequency value is set.

A new frequency is only taken over at mode change. This is reachable by changing into the IDLE-Mode (Byte 10 = 00h) after the start-up of the drive. Now type the new scheduled frequency and set Byte 10 back to 03h. The drive is set to the new frequency immediately.

For acceleration of the drive, the values set in the parameters are used. If there are no presetting, the default values are used.

As long as the drive is operating, the output "**Axis in run**" is set. By presetting 00h as scheduled frequency (mode change required) the drive stops and the output is set back.

**Stop drive by permanent run and set frequency = 00h**

By setting a scheduled frequency of 00h in Byte 4-7 and the mode 03h in Byte 10 you may stop the drive at any time.

**Note!**

Please regard, that a frequency change is only recognized by the module via a mode change. This is also valid for stopping the drive. For a mode change, use the short time jump to the IDLE-Mode.

**Set position**

Default: Byte 10 = 06h, Byte 0-3: Position value

In the operating mode "Set position" you may assign a new value to the recent actual value.

Herefore you predefine the new value in Byte 0-3 and then set the MODE-Byte 10 to 06h.

## Reference run

Default: Byte 10 = 02h, Byte 15 = Control bits for reference run

The reference run supports the calibration of your drive system. The reference point should be inside the drive outline.

Before starting a reference run you have to specify the type of the reference run and the direction to run to in Byte 15.

By setting Byte 10 to 02h, the drive starts with its reference run.

As frequency the reference frequency set in the parameters are used. If there are no parameters, the default values are used.

There are 6 different possibilities for the reference drive that are predefined via Byte 15:

- Reference run to reference switch and delete position counter
- Reference run to reference switch and keep position counter
- Reference run to end switch B and delete position counter
- Reference run to end switch B and keep position counter
- Reference run to end switch A and delete position counter
- Reference run to end switch A and keep position counter

## Control bits for the reference run

The control bits in Byte 15 have the following assignment:

Byte 15	Parameter
Bit 0	1: Direction forward 0: Direction backward
Bit 1	1: delete position after reference run 0: keep position after reference run
Bit 2	Reference run to reference switch
Bit 3	Reference run to end switch B
Bit 4	Reference run to end switch A



### Note!

When starting a reference run, please regard, that you always have to set a direction via Bit 0 and that you may set only one bit in the Bits 2...4!

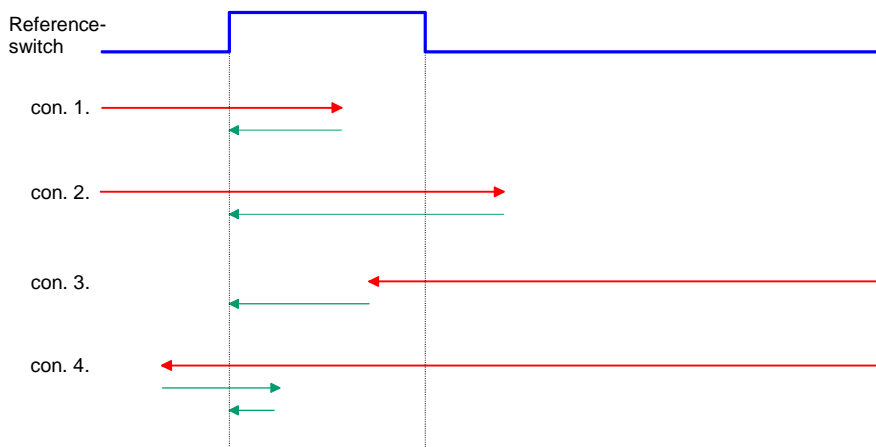
**Reference run to reference switch**

The reference run starts always with the speed predefined in FREF. The direction has to be preset in the variable parameter (Byte 15, Bit 0). As soon as the ascending edge of the reference switch is recognized, the motor slows down to FSTART.

Depending on the reference speed the drive may overrun the reference switch or not during slow down.

The following 4 drives to the reference switch are possible:

1. Motor comes from the left side, slows down inside the reference switch and drives backward with FSTART until the descending edge of the reference switch is recognized.
2. Motor comes from the left side, overruns the reference switch during slow down and drives backward with FSTART over the ascending edge until the descending edge of the reference switch is recognized.
3. Motor comes from the right side, slows down inside the reference switch and drives with FSTART until the descending edge of the reference switch is recognized.
4. Motor comes from the left side, overruns the reference switch during slow down, it changes the rotational direction and drives with FSTART until the ascending edge of the reference switch is recognized, switches the direction again and drives on until the descending edge of the reference switch is recognized.



**Reference run to end switch**

You may limit your distance via the end switches A and B.

At the reference run to end switch the drive starts and drives with the preset speed FREF and the predefined rotational direction until the according end switch gets active, stops abruptly, changes its rotational direction and drives with FSTART until the end switch is inactive again.



**Note!**

If you use the reference run to end switch, you have to regard, that there is enough space behind the end switch for the motor to slow down!



## FM 253 - Data transfer >> CPU

**Respond message** The MotionControl Stepper module sends a data block to the CPU cyclically that contains several information about the recent state of the drive. The data block has a length of 16Byte and the following structure:

Byte no.	Content	Length
0-3	actual position	4 Byte
4-7	actual frequency	4 Byte
8-9	error messages	2 Byte
10	actual mode	1 Byte
11	state	1 Byte
12-15	data of variables	4 Byte

### Actual position, actual frequency

Via this two parameters the actual position and frequency of your drive is always shown.

### Error messages

The recently recognized errors are monitored via the error bits of Byte 8-9. The errors remain active until the according Bits are set back. As long as an error is still valid, the according error bit is set again after the reset.

The following error messages are used:

#### *Error byte (Byte 8-9)*

Byte 9	Description
Bit 0	Error in the internal state administration
Bit 1	System has been booted (always after PowerON)
Bit 2	Error at proofing Flash parameters, motor parameters not valid
Bit 3	This function is not permitted during motor run
Bit 4	Motor is recently blocked
Bit 5	Error at positioning the motor
Bit 6	End switch is/was active
Bit 7	Frequency has been limited to FMAX
Byte 8	
0	General error at the motor

**Set back error messages**

For deleting an active error (Byte 8-9) you have to set the according error bit in the variables parameter (Byte 14-15) to "1".

As soon as you set the **Mode (Byte 10) to 07h**, the according errors in the module are set back. You may also set back several error messages at the same time. FFFFh in Byte 14-15 for example sets back all errors.

**Recent mode**

Here you always find the mode that your FM 253 has at the moment. The following modes may be shown:

*Mode (Byte 10)*

Byte	Mode
10	00h: IDEL 01h: Positioning relative 02h: Reference run 03h: Permanent run axis 04h: Read inputs 05h: Change motor parameters 06h: Set position 07h: Delete error 08h: Positioning absolute

**State**

The STATE-Byte shows you the state of the drive. The following state messages may be shown:

*State (Byte 11)*

Byte 11	State
Bit 0	1: Drive in run 0: Drive in stop
Bit 1	1: Direction forward 0: Direction backward
Bit 2	1: Drive in position 0: Drive not in position

**Read inputs**

For reading the inputs, the **Mode (Byte 10)** is set to **04h** and now the module shows the state of the end switches and the reference switch in the variables data (Byte 15).

*Inputs (Byte 15)*

Byte 15	Input
Bit 0	State PA end switch (1: operated, 0: not operated)
Bit 1	State PB end switch (1: operated, 0: not operated)
Bit 2	State RE reference switch (1: operated, 0: not operated)

## FM 253 - Handling blocks

### Overview

There are different handling blocks available with the FM 253 to make the usage of the module more comfortable. The following handling blocks are available for the FM 253 at this time:

Block	Description
FC 200	Control drive
FC 201	Adjustment of a parameter
FC 202	Adjustment of all drive parameters (Index 0...9)

### FC 200 Control drive

This FC serves the control of your drive by transferring the drive data to the module through setting the according mode.

With this FC you may transfer all modes except "Set parameters" and the according parameters to the module.

### Data transfer to FM 253 with SET\_MODE = 1

- Set the mode.
- Give data to the according parameters.
- Start the transfer by setting SET\_MODE to 1. When the mode is started, the module SET\_MODE is set back at the next cycle and shows the actual data of the FM 253.

### Data transfer to CPU with SET\_MODE = 0

At the call of the FC 200 with SET\_MODE = 0, the actual data of the FM 253 is shown via the labels ACT\_POSITION, ACT\_FREQUENCY, ACT\_MODE, ERROR, STATE and VAR\_DATA.

It is convenient to store the single values in a data block. In the following example we used DB5 for this purpose.

### Parameters

Address	Declaration	Name	Type	Start value	Comment
0.0	in	ADDRESS	INT		Set basic address
2.0	in	SET_POSITION	DINT		Transfer position values
6.0	in	SET_FREQUENCY	DINT		Transfer frequency at permanent run
10.0	in	VARIABLES	DWORD		Transfer variables at reference run
14.0	in	MODE	INT		Transfer mode to change
16.0	out	ACT_POSITION	DINT		Response actual position
20.0	out	ACT_FREQUENCY	DINT		Response actual frequency
24.0	out	ERROR	INT		Error word
26.0	out	ACT_MODE	INT		Response actual mode
28.0	out	STATE	BYTE		Response status bits
30.0	out	VAR_DATA	DWORD		Response variables
34.0	in_out	SET_MODE	BOOL		Start function

**ADDRESS** Start address from where on the FM 253 is stored in the CPU.

**SET\_POSITION** In mode 01, 06 and 08 you fix the scheduled position for the drive here.

**SET\_FREQUENZ** In mode 03 you fix the scheduled rotational speed as set frequency.

**VARIABLES** Fix here the control bits for the reference run (MODE = 02) and for setting the errors back (MODE = 07).

The control bits for the reference run have the following assignment:

*Control bits*

VARIABLE-Byte	Parameter
Bit 0	1: Direction forward 0: Direction backward
Bit 1	1: after reference run delete position 0: after reference run keep position
Bit 2	Reference run to reference switch
Bit 3	Reference run to end switch B
Bit 4	Reference run to end switch A

An overview over the error-bit-assignment follows below.

**MODE** With this parameter you transfer the mode to the FM 253. The following modes are possible:

*Mode*

Value	Description	Default in	Response in
00	Idle-Mode - no status change of the drive, serves for parameter changes	-	-
01	Positioning relative - driving the preset number of steps	SET_POSITION	-
02	Reference run - calibration of the drive	VARIABLES	-
03	Permanent run axis - drive runs with scheduled frequency	SET_FREQUENCY	-
04	Read inputs - responds with the end switches states	-	VAR_DATA
06	Set position - sets the recent position in the module without moving the drive	SET_POSITION	-
07	Delete error - deletes the error bit activated with 1	VARIABLES	-
08	Positioning absolute - drive to scheduled position	SET_POSITION	-

**ACT\_POSITION,  
ACT\_FREQUENCY** Via those parameters the recent actual position and actual frequency of your drive is shown.

**ERROR** Here you may find error messages if occurred. The errors remain active until the error cause is removed and the according bits are set back. The following error messages may occur:

*Error messages*

ERROR- Byte 1	Description
Bit 0	Error in the internal state administration
Bit 1	System booted (always after PowerON)
Bit 2	Error at validating the Flash parameters, motor parameters not valid
Bit 3	Function is not available during motor run
Bit 4	Motor is blocked
Bit 5	Error at positioning the motor
Bit 6	End switch is/was active
Bit 7	Frequency has been limited to FMAX
ERROR- Byte 0	
0	General error at the motor

The clearing of the error messages takes place via MODE = 07 and VARIABLE = Error bytes.

**ACT\_MODE** Responds the mode in which the module is at this moment.

**STATE** The STATE-Byte shows you information about the state of the drive. The following state messages may occur:

*State*

STATE- Byte	State
Bit 0	1: Drive in run 0: Drive in stop
Bit 1	1: Direction forward 0: Direction backward
Bit 2	1: Drive in position 0: Drive not in position

**VAR\_DATA**

In VAR\_DATA the state of the inputs is returned after you requested this by MODE = 04. For reading the inputs the **Mode 4** is set and now the module shows the state of the end switches and the reference switch in the variables data (Byte 15).

*Inputs*

VAR_DATA-Byte	Input
Bit 0	State PA end switch (1: operated, 0: not operated)
Bit 1	State PB end switch (1: operated, 0: not operated)
Bit 2	State RE reference switch (1: operated, 0: not operated)

**SET\_MODE**

After you defined the according parameters the data is transferred to your module via SET\_MODE = 1.

When the mode has been started, the module sets back again the SET\_MODE in the next cycle and returns the actual data of the FM 253.

**Example**

```

DB 5
DBD 0 Position          DINT    L#0      Position value
DBD 4 Frequency        DINT    L#0      Frequency for permanent run
DBW 8 reserve          WORD    W#16#0
DBW 10 MODE            INT     0        Mode
DBW 12 Index          INT     0        Index default
DBD 14 Variable_PARAM DWORD   DW#16#0  Var. for Ref.run/Param...
DBW 18 Reserve1       WORD    W#16#0
DBD 20 Act_Position   DINT    L#0      actual position
DBD 24 Act_Frequency  DINT    L#0      actual frequency
DBW 28 Error          INT     0        error monitor
DBW 30 ACT_Mode       INT     0        actual mode
DBW 32 State          BYTE    B#16#0  State response
DBD 34 VAR_DATA       DWORD   DW#16#0  Return parameter/data

CALL FC 200 //FC for Stepper module
ADDRESS      :=128 //Module address
SET_POSITION :=DB5.DBD 0 //DBD with position for abs/rel
SET_FREQUENCY:=DB5.DBD 4 //DBD with frequency for permanent run
VARIABLES    :=DB5.DBD14 //Delete data for Ref_Run/Del error
MODE         :=DB5.DBW10 //Mode default for new order
SET_MODE     :=M1.0 //Start order
ACT_POSITION :=DB5.DBD20 //actual position
ACT_FREQUENCY :=DB5.DBD24 //actual frequency
ERROR        :=DB5.DBW28 //Monitor error
ACT_MODE     :=DB5.DBW30 //actual mode
STATE        :=DB5.DBW32 //State bits from module
VAR_DATA     :=DB5.DBD34 //Return of values
                                     e.g. read inputs

```

**FC 201 -  
set a parameter**

With the FC 201 it is possible to set a parameter at the FM 253.

**Parameter**

Address	Declaration	Name	Type	Start value	Comment
0.0	in	ADDRESS	INT		Fixed basic address
2.0	in	INDEX	INT		Transfer INDEX for parameters
4.0	in	PARAMETER	DWORD		Transfer parameter value
	out				
8.0	in_out	SET_PARA	BOOL		Start parameter transfer

**ADDRESS**

Start address from where on the FM 253 is stored in the CPU.

**INDEX**

Via INDEX you fix the parameter where the value is set in PARAMETER.

Index	Parameter	Unit	Value range	Default	Description
00h	Fstart	Hz	UINT32	200	Start frequency
01h	F1	Hz	UINT32	4000	Limit frequency 1
02h	dF1	Hz	UINT32	100	Acceleration from Fstart ⇒ F1
03h	F2	Hz	UINT32	10000	Limit frequency 2
04h	dF2	Hz	UINT32	60	Acceleration from F1 ⇒ F2
05h	Fmax	Hz	UINT32	30000	Maximum drive frequency
06h	dFmax	Hz	UINT32	40	Acceleration from F2 ⇒ Fmax
07h	Fpos	Hz	UINT32	30000	Frequency at positioning
08h	Fref	Hz	UINT32	1000	Frequency for reference run
10h	Fist	Hz	UINT32	-	Actual motor frequency (read only)
11h	Fsoll	Hz	UINT32	-	Actual sched. frequency (read only)
13h	FTarget	Hz	UINT32	-	Target frequency (read only)
97h				-	Store parameters in Flash
98h				-	Read parameters from Flash (State like after PowerON)
99h				-	Load default parameters

**PARAMETER**

Here you type the value of the parameter specified via INDEX.

**SET\_PARA**

After you filled the according parameters, the parameter is transferred to your module via SET\_PARA = 1. After the transfer SET\_PARA is set back automatically.

**Parameterize FC 202 - FM 253** Via the FC 202 you may adjust all relevant parameters of the FM 253.

### Parameter

Address	Declaration	Name	Type	Start value	Comment
0.0	in	DATA_DB	BLOCK_DB		Data block with parameters
2.0	in	ADDRESS	INT		Module address
	out				
4.0	in_out	START	BOOL		Start parameter transfer
4.1	in_out	RUN	BOOL		Transfer single runs

**DATA\_DB** Please fix here the data block where your parameters are stored.  
The DB has the following structure:

```

DBD 0 Fstart      DINT  L#0  Start frequency
DBD 4 F1          DINT  L#0  Limit frequency 1
DBD 8 F2          DINT  L#0  Limit frequency 2
DBD 12 Fmax       DINT  L#0  Maximum drive frequency
DBD 16 dF1        DINT  L#0  Acceleration Fstart --> F1
DBD 20 dF2        DINT  L#0  Acceleration F1 --> F2
DBD 24 dFmax      DINT  L#0  Acceleration F2 --> Fmax
DBD 28 Fpos       DINT  L#0  Frequency at positioning
DBD 32 Fref       DINT  L#0  Frequency at reference run
DBD 36 StepRepeat DINT  L#0  Step between frequency calculation

```

**ADDRESS** Start address from where on the FM 253 is stored in the CPU.

**START** After you created the DB you may transfer your parameters to your module via START = 1.  
As soon as all parameters are transferred, START is set back again.

**RUN** This variable stores one cycle spreading state and it is responsible for the single parameter transfer.



## FM 254 - MotionControl Servo

### Features

- Microprocessor controlled positioning module for drives with an analog set point interface ( $\pm 10V$  control voltage)
- 7 operating modes
- closed-loop position control
- The module operates independently and it is controlled by means of an application program in the System 200V.
- Data is saved in Flash-RAM. No backup battery is required.

### Application areas

The positioning module can be employed for simple position control tasks as well as profile-controlled drive outlines that meet the most stringent requirements with respect to dynamics, accuracy and speed.

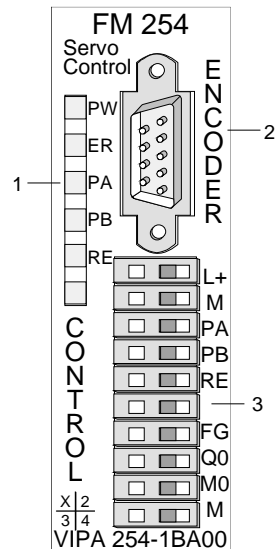
Due to the various modes of operation the module can also be employed for positioning applications on machines that employ very high clock rates.

Typical applications:

- Production and transportation equipment, transfer lines and assembly lines
- Presses
- Woodworking machines
- Handling equipment
- Feeder devices
- Packing machines
- Auxiliary actuators for lathes and milling machines

## FM 254 - MotionControl Servo - Construction

### Front view



- [1] LED status indicators
- [2] Encoder interface
- [3] Connector for supply voltage, drive, end switch and outputs

### Components

#### LEDs

The positioning module FM 254 has 6 status indicator LEDs.

The following table contains the description and the respective color of these LEDs.

Label	Color	Description
PW	Yellow	24V DC supply voltage is applied
ER	Red	internal error
PA	Green	Limit value A overrun, input PA is set
PB	Green	Limit value B overrun, input PB is set
RE	Green	Reference point overrun
FG	Green	Drive released



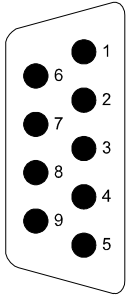
#### Note!

If the PW-LED is not on during operation, this may depend on a short circuit in the DC 24V voltage supply.

Please control also the connections of the encoder plug.

If the LED remains off even when you disconnect the encoder plug, the module has a defect.

**Encoder interface** *9pin D-type plug*



Pin	Assignment
1	+24V encoder power
2	+5V encoder power
3	R+ clock input null pulse
4	B+ clock input
5	A+ clock input
6	Ground encoder power
7	R- clock input null pulse
8	B- clock input
9	A- clock input

**Control interface**



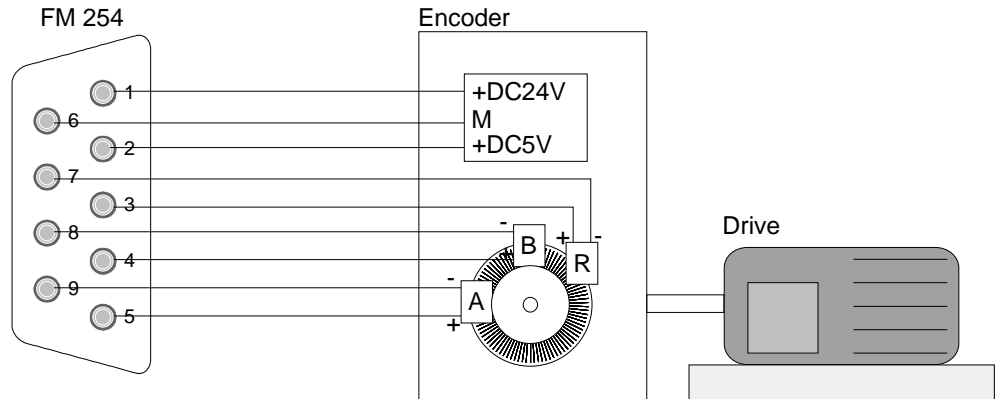
Pin	Assignment
1	DC 24V supply voltage
2	Ground 24V
3	Input for start switch (low active)
4	Input for end switch (low active)
5	Input for reference switch (low active)
6	reserved
7	Output regulator release
8	Analog output ground
9	Analog output +
10	Screen

## FM 254 - Connecting a drive with encoder

### Connection of an encoder

The encoder is wired to the 9pin D-type connector located at the front. The module supplies the encoder with the required DC 24V and DC 5V voltages.

The following figure shows the connection of an encoder:



### Connection of supply voltage, drive, end switch and outputs

#### Power supply

The module requires a power supply of DC 24V via pins 1 and 2.

#### End switches

You may connect up to 3 end switches (opener) to the module.

The end switches for the extremes of the distance are connected to terminals 3 and 4. The drive will be stopped immediately as soon as one of these switches is operated. In this situation may only be driven into the opposite direction.

The reference switch is connected to terminal 5. This is required to tune the drive to the positioning module.

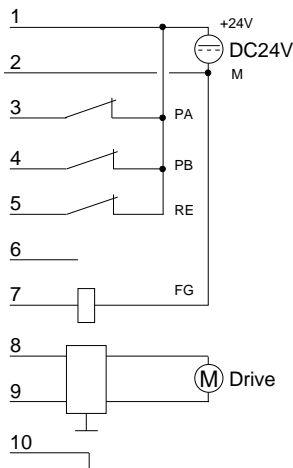
The end switch that stops the drive in the mode hardware-controlled run is also connected to terminal 5.

#### Outputs

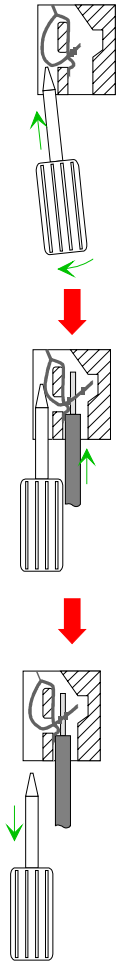
The module has 2 outputs that are controlled directly by the module. At present, however, only the output "Controller Enable" (pin 7) is available. The second output is intended for future expansion. You enable the output by setting bit 0 in the traversing data.

#### Drive

Pin 8 and 9 supply an analog signal for  $\pm 10V$  regulator control.



## Cabling



The drive and the end switches are to connect at the control interface. Herefore a 10pin plug with CageClamp technology from WAGO is used. The cabling with CageClamps is very fast and in opposite to screw connections vibration secure.

You may connect cores with a core cross-section from  $0.08\text{mm}^2$  up to  $1.5\text{mm}^2$ .

The cabling is analog to the big CageClamps of the System 200V.

Push the spring in the square opening with a fitting screwdriver more inside and insert the core into the rectangular opening.

By releasing the screwdriver the core is securely fixed.



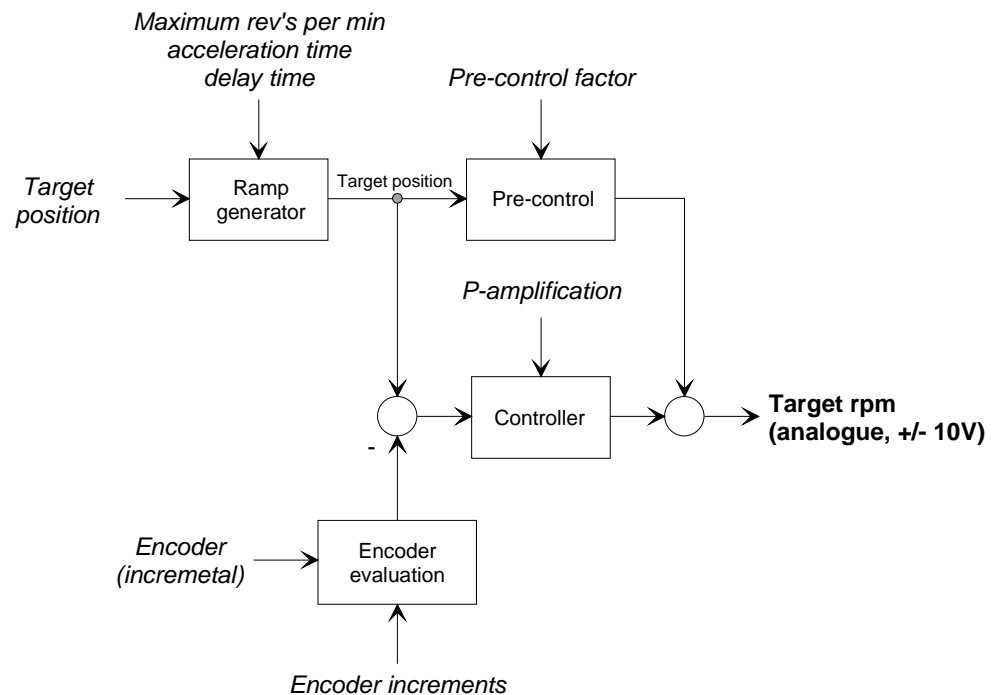
## FM 254 - Summary of parameters and transfer values

The following table lists all the parameters and transfer values. A block diagram depicts the interaction between the parameters.

### Overview

Value	Size	Unit	Physical range
Destination position Set position Actual position	32Bit	$\frac{1 \text{ rotation}}{2^{16}}$	0.0 ... 65535.9999 revs.
Maximum rpm.	16Bit	1/min	10 ... 300000 1/min
Acceleration time Delay time	16Bit	10 ms	10 ms ... 100 s
P-amplification	16Bit	0.1	0.0 ... 1000.0
Pre-control factor	16Bit	0.1	0.0 ... 1.0
Encoder increments	16Bit	1	10 ... 10000
Operating mode	16Bit	binary coding	

### Block diagram



## FM 254 - Parameterization

When commissioning the MotionControl Servo module it requires 16Byte of parameter data. These have the following structure:

### Parameter data (write only)

Byte no.	Name	Length	Range	Unit
1, 0	Maximum rotat. speed	2Byte	10.....1000	1/min
3, 2	reserved	2Byte	-	-
5, 4	reserved	2Byte	-	-
7, 6	P-amplification	2Byte	0.0 ... 1000.0	0.1
9, 8	Pre-control factor	2Byte	0.0 ... 1.0	0.1
11, 10	Encoder increments	2Byte	10 ... 10000	1
13, 12	Reference rot. speed	2Byte	10 ... 1000	1/min
14	Pos. reached window	1Byte	0.....255	1INK
15	Drag distance	1Byte	4.....1020	4INK

### Parameter description

#### *Maximum rotational speed*

Defines the maximum rotations for your drive.

#### *P-amplification, Pre-control factor*

These values control the regulation properties.

#### *Encoder increments*

This parameter matches your MotionControl Servo module to the encoder.

#### *Reference rotational speed*

This value for the rotational speed is used for the reference run that is required by the MotionControl Servo module to re-acquire parameters for the control path.

#### *Pos-reached-window*

When the target position has been reached, this position is maintained by continuous control of the drive. The drive is never stopped.

You can specify a window by entering certain increments into the *Pos-reached-window*. These define the tolerance by which the actual value may differ from the target position before the drive is controlled, i.e. when the drive is stationary.

#### *Drag distance*

This parameter defines the drag error or the difference between the actual and the set value, which causes the drive to be stopped.

## FM 254 - Data transfer >> FM 254

**Traversing data** The CPU can control the MotionControl Servo module by writing the following values into the FM 254 module:

Byte no.	Name	Length	Range	Unit
3, 2, 1, 0	Target position	4Byte	32 Bit Integer	Encoder increments
5, 4	Control bytes	2Byte		
7, 6	Rot. speed	2Byte	10.....6000	1/min
9, 8	Acceleration time	2Byte	1...10000	10ms
13, 12, 11, 10	Parameter field	4Byte		
15, 14	Field identifier	2Byte		

### Control bytes (Byte 4 and Byte 5)

Byte	Bit 7 ... Bit 0
4	Bit 0: Enable Bit 1: Operating mode reference run positive Bit 2: Operating mode reference run negative Bit 3: Operating mode hardware-controlled run positive Bit 4: Operating mode hardware-controlled run negative Bit 5: Operating mode incremental dimension Bit 6: Operating mode infinite incremental dimension Bit 7: Taking over target position
5	Bit 0: reserved Bit 1: Non-maintained command mode direction of rotation pos. Bit 2: Non-maintained command mode direction of rotation neg. Bit 7 ... Bit 3: reserved

These operating modes are described below.

### Parameter field and Field identifiers (Byte 10 ... Byte 14)

You can send additional parameters with the traversing data to the MotionControl Servo module by specifying a field identifier. The parameters for the respective field identifier must be entered into the parameter field (Byte 10...13).

The FM 254 will use the default settings shown below if you do not transfer any field identifiers.

Field ident.	Description	Range	Unit	Default setting
FF01h	Software end switch (+)	32Bit Integer	Encoder increments	7FFF.FFFF
FF02h	Software start switch(-)	32Bit Integer	Encoder increments	8000.0001
FF03h	Rot. speed at non-maintained command mode	10.....6000	1/min	Reference rot. speed
FF04h	Delay time	1...10000	10ms	Acceleration time



## FM 254 - Operating modes

### Overview

The following operating modes can be selected by setting the respective bit in the control byte:

- Positioning operation (positioning to an absolute target position)
- Reference run (system calibration)
- Hardware run (drive to reference switch)
- Incremental run (use addition to approach a relative target)
- Infinite incremental run (relative traversal without counter overflow)
- Non-maintained command mode

---

### Positioning mode

#### Operation

During the positioning operation the absolute target position is only transferred to the FM 254, if the bit "Taking over target position" is set.

If a new position is specified with the enable bit set, the drive moves to the respective position  $\pm$  POS-REACHED-WINDOW with the values that were previously specified for the rotational speed and the acceleration/delay and sets the "Position reached"-Bit. After transferring the parameters for the traversal, you can start the drive by setting the enable bit. During the traversal the module indicates the direction of rotation by setting bit 1 or 2.

Should the deviation between set and actual position exceed the window specified for the drag error, the positioning operation is terminated and the motor is stopped. The program is notified by means of an active drag error bit 0 in Byte 5. You can clear the drag error bit by resetting the enable bit. This also sets the set position to the actual position.

The drive is also stopped if soft- or hardware switches are passed that terminate the traversal distance.

The operation can be continued at any time by setting the enable bit.

The acceleration/delay time can be modified before a new command is issued.

It is always possible to specify a new value for the rotational speed by modifying the traversing data. If the rotational speed is changed while movement is taking place, the new value is attained respecting the current acceleration/delay times.

#### Control bytes

The control bytes that you use to specify this operating mode are an integral part of the traversal data.

A general description of the traversal data is available on pages 11-30.

Byte	Bit 7 ... Bit 0
4	Bit 0: enable (drive is started) Bit 6 ... Bit 1: 0 Bit 7: irrelevant
5	Irrelevant

## Reference run

### Operation

The reference run calibrates your drive system. The point of reference should be located on the path of traversal.

Start the reference run:

- Set the enable bit.
- Release the reference run by means of the bit "Reference run positive" or "Reference run negative".
  - The drive will travel to the point of reference using the reference rotational speed specified in the parameter set.
  - As soon as the point of reference is passed, the reference switch is operated (LED RE is turned off).
  - The position of the point of reference is recorded in memory.
  - The drive is reversed up to the next encoder zero pulse.

This concludes the reference run and the bit "Reference detected" is set.



### Note!

Please remember that a set position is not required for operating mode "Reference run". The set position is ignored.

### Control bytes

The control bytes that you use to select this operating mode are included in the traversing data.

A general description of the traversal data is available on pages 11-30.

Byte	Bit 7 ... Bit 0
4	Bit 0: enable (drive is started) Bit 2 ... Bit 1: 01: reference run positive 10: reference run negative Bit 6 ... Bit 3: 0 Bit 7: irrelevant
5	Irrelevant

---

## Hardware run

**Operation**

This mode is only used to approach a target position until the drive is stopped by an overrun end switch. The end switch must be connected to the reference switch input.

The traversal is governed by the values that were specified for rotational speed and acceleration or delay times. After the end switch is reached the respective position is stored internally and the drive is stopped with the specified delay time.

When the drive has stopped, it is reversed to the position of the end switch where it is stopped finally. At this point bit 3 is set to indicate "Position reached". For the reverse movement the MotionControl Servo module uses the reference rotational speed specified in the parameterization.

A new traversal can be initiated by toggling the bits "enable" and "HW ref. positive".

The acceleration/delay time can be modified before a new job is initiated.

If the rotational speed is altered when during the traversal, the new value is achieved by means of the current acceleration/delay time values.

**Note!**

Please remember that a set position is not required for operating mode "Hardware run". The set position is ignored.

**Control bytes**

The control bytes that you use to select this operating mode are included in the traversing data.

A general description of the traversal data is available on pages 11-30.

Byte	Bit 7 ... Bit 0
4	Bit 0: enable (drive is started) Bit 2 ... Bit 1: 0 Bit 4 ... Bit 3: 01: Hardware run positive 10: Hardware run negative Bit 6 ... 5: 0 Bit 7: irrelevant
5	Irrelevant

---

## Incremental run

**Operation**

The incremental mode makes use of relative positions, i.e. the value supplied as set position is added to the actual position.

When the enable bit is set, the drive travels in a positive or negative direction for the specified relative value. The drive uses the predefined values for rotational speed and acceleration to travel to the new position. If the position is negative the drive will be reversed.

You can modify the acceleration/delay time before you initiate a new job.

If the rotational speed is altered when during the traversal, the new value is achieved by means of the current acceleration/delay time values.

**Control bytes**

The control bytes that you use to select this operating mode are included in the traversing data.

A general description of the traversal data is available on pages 11-30.

Byte	Bit 7 ... Bit 0
4	Bit 0: enable (drive is started) Bit 4 ... Bit 1: 0 Bit 5: 1 (Incremental run) Bit 6: 0 Bit 7: irrelevant
5	Irrelevant

---

## Infinite incremental mode

### Operation

In this mode the position supplied as a value is approached as a relative position when enabled. When the position is reached, the set and the actual position are set to zero. You can use this mode to move the drive in one direction without counter overflow condition.

You can modify the acceleration/delay time before you initiate a new job.

You may specify a new value for the rotational speed at any time. If the rotational speed is altered during the traversal, the new value is achieved by means of the current acceleration/delay time values.

### Control bytes

The control bytes that you use to select this operating mode are included in the traversing data.

A general description of the traversal data is available on pages 11-30.

Byte	Bit 7 ... Bit 0
4	Bit 0: enable (drive is started) Bit 5 ... Bit 1: 0 Bit 6: 1 (Infinite incremental run) Bit 7: irrelevant
5	Irrelevant

## Non-maintained command mode

### Operation

The drive is released by setting Bit 0 in Byte 4 (enable) with before opposed rotational speed and acceleration. By setting Bit 1 or Bit 2 in Byte 5, a rotation direction is given and the drive starts. The drive stops as soon as Bit 1 or Bit 2 of Byte 5 is set back.

### Control bytes

The control bytes that you use to select this operating mode are included in the traversing data.

A general description of the traversal data is available on pages 11-30.

Byte	Bit 7 ... Bit 0
4	Bit 0: enable (drive is started)
5	Bit 0: V1.08 - Reset counter at non-maintained command mode (edge 0 after 1 sets back the actual position to zero) *1) Bit 1: 1 direction of rotation positive Bit 2: 1 direction of rotation negative



### Note!

\*1) The reset of the counter may only be executed in the non-maintained command mode. During positioning mode the regulator would throw a drag error because of the jumping actual value.

## FM 254 - Data transfer >> CPU

The following values are transferred cyclically by the MotionControl Servo module to the CPU and stored.

Byte no.	Name	Length	Range	Unit
3, 2, 1, 0	Set position	4Byte	32Bit Integer	Encoder increments
7, 6, 5, 4	Actual position	4Byte	32Bit Integer	Encoder increments
9, 8	Set rotational speed	2Byte	10.....6000	1/min
11, 10	Operating mode	2Byte	binary coded	
13, 12	reserved	2Byte	-	-
15, 14	Reply field identifier	2Byte		hex

### Operating state

Byte	Bit 7 ... Bit 0
10	Bit 0: enable issued Bit 1: clockwise rotation Bit 2: anticlockwise rotation Bit 3: position reached Bit 4: HW start switch operated Bit 5: HW end switch operated Bit 6: HW reference switch operated Bit 7: Reference detected
11	Bit 0: Drag error detected Bit 4: SW end switch anticlockwise rotation Bit 5: SW end switch clockwise rotation Bit 7 ... 1: irrelevant

### Example

If the MotionControl Servo module was addressed starting at peripheral address PY128 in your CPU, you may obtain the "set position" from PY128 to PY131.

Other values follow these values in the peripheral area in accordance with the list above.

For example, the 2Byte for the "Operating state" are located at PY138...PY139.

## Technical data

### MotionControl Stepper FM 253

Electrical data	VIPA 253-1BA00
Number of axis	1
Voltage supply	DC 24V (20.4 ... 28.8) via front from ext. power supply
Current consumption backplane bus	typ. 320mA, max. 500mA
Status monitor	via LEDs at the frontside
Connectors / Interfaces	
"Drive"-Interface	Output for pulse, direction and release with RS422
Max. Impulse frequency	200kHz
Digital inputs	
Number	3
Function	2 end switch, reference switch
Signal voltage "0"	0 ... 5V
Signal voltage "1"	15 ... 28.8V
Digital outputs	
Number	2
Function	"axis in motion", "position reached"
Output current	1A protected against sustained short circuits
Potential separation	yes
Programming data	
Input data	16Byte
Output data	16Byte
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	80g



**MotionControl**  
**Servo module**  
**FM 254**

Electrical data	VIPA 254-1BA00
Voltage supply	DC 24V (20.4 ... 28.8) via front from ext. power supply
Current consumption	200mA
Current consumption backplane bus	100mA
Status indicator	via LEDs on the frontside
Connectors / interfaces	
Encoder	Incremental encoder
Signal voltages	5V as per RS 422
Supply voltage	5.2V / 300mA 24V / 300mA
Input frequency and line length	1MHz max. with 10m screened line 500kHz max. with 35m screened line
Control	
Set point output	-10 ... +10V
Digital inputs	
Number	3
Supply voltage	DC 24V
Digital outputs	
Number	1
Potential separation	no
Output current	0.5A
Lamp load	5W
Programming data	
Input data	16Byte
Output data	16Byte
Parameter data	16Byte
Diagnostic data	-
Dimensions and Weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	80g



## Chapter 12 Power supplies

**Overview** This chapter contains descriptions of the System 200V power supplies.

Below follows a description of the:

- Power supply 2A
- Power supply 2A with terminal module
- Installation and wiring
- Technical data

<b>Contents</b>	<b>Topic</b>	<b>Page</b>
	<b>Chapter 12 Power supplies .....</b>	<b>12-1</b>
	Safety precautions .....	12-2
	System overview .....	12-3
	PS 207/2 - Power supply - Construction .....	12-4
	PS 207/2CM - Power supply with Clamps - Construction .....	12-6
	Installation .....	12-8
	Wiring .....	12-9
	Technical data .....	12-10

## Safety precautions

### Appropriate use

The power supplies were designed and constructed:

- to supply DC 24V to the System 200V components
- to be installed on a t-rail along with System 200V components
- to operate as DC 24V stand-alone power supply
- for installation in a cabinet with sufficient ventilation
- for industrial applications

**The following precautions apply to applications employing the System 200V power supplies.**



### Danger!

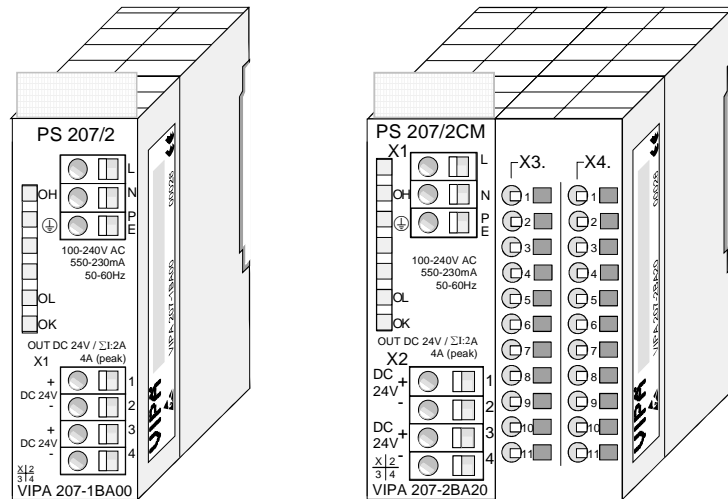
- The power supplies must be installed in protected environments that are only accessible to properly qualified maintenance staff!
- The power supplies are not certified for applications in explosive environments (EX-zone)!
- You have to disconnect the power supply from the main power source before commencing installation or maintenance work, i.e. before you start to work on a power supply or the supply cable the main supply line must be disconnected (disconnect plugs, on permanent installations the respective fuse has to be turned off)!
- Only properly qualified electrical staff is allowed to install, connect and/or modify electrical equipment!
- To provide a sufficient level of ventilation and cooling to the power supply components whilst maintaining the compact construction it was not possible to protect the unit from incorrect handling and a proper level of fire protection. For this reason the required level of fire protection must be provided by the environment where the power supply is installed (e.g. installation in a switchboard that satisfies the fire protection rules and regulations)!
- Please adhere to the national rules and regulations of the location and/or country where the units are installed (installation, safety precautions, EMC ...).

## System overview

The System 200V power supplies are provided with a wide-range input that is connectable to AC 100 ... 240V. The output voltage is DC 24V at 2A/48W.

Since all inputs and outputs are located on the front of the unit and since the enclosure is isolated from the backplane bus you may install the power supply along with the System 200V on the same t-rail or you can use it as a separate external power supply.

The following power supplies are currently available:



### Order data

Order number	Description
VIPA 207-1BA00	Power supply PS 207/2 primary AC 100...240V, secondary DC 24V, 2A, 48W
VIPA 207-2BA20	Power supply PS 207/2CM primary AC 100...240V, secondary DC 24V, 2A, 48W with terminal module 2x11 clamps

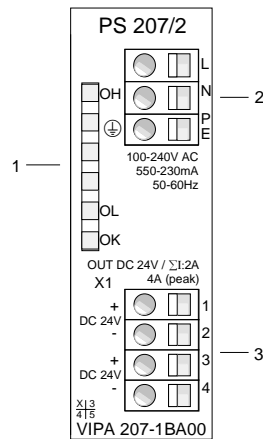
## PS 207/2 - Power supply - Construction

### Properties

The power supply is distinguished by the following properties:

- Wide-range input AC 100...240V without manual intervention
- Output voltage DC 24V, 2A, 48W
- Can be installed on a t-rail together with other System 200V components or as stand-alone device
- Protection from short-circuits, overload and open circuits
- Typically 90% efficiency at  $I_{rated}$

### Construction



- [1] LED status indicator
- [2] AC IN 100 ... 240V
- [3] DC OUT 24V, 2A, 48W

### LEDs

The front of the power supply carries 3 LEDs for troubleshooting purposes. The following table lists the significance and the respective color.

Name	Color	Description
OH	red	Overheat: turned on by excessive temperatures
OL	yellow	Overload: turned on when the total current exceeds the maximum capacity of app. 3A.
OK	green	Turned on when the power supply operates properly and supplies DC 24V power.



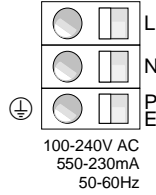
### Note!

Only one LED is on at unit operation.

When all the LEDs are extinguished while the power supply is operational, a short circuit is present or the power supply has failed.

**Connector wiring**

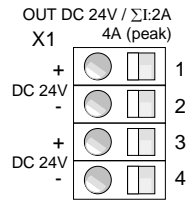
**Input voltage  
INPUT  
AC 100...240V**



The power supply must be connected to a source of AC power via the input connector.

A fuse protects the input from overloads.

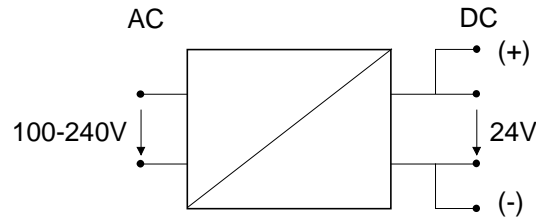
**Output voltage  
OUTPUT  
DC 24V, 2A**



Two connectors are provided for connection to System 200V modules that require an external source of DC 24V .

Both outputs are protected against short circuits and have an output voltage of DC 24V with a total current of 2A max.

**Block diagram**



**Danger!**

- You need to disconnect the power supply from the main power source before commencing installation or maintenance work, i.e. before you start to work on a power supply or the supply cable, the main supply line must be disconnected (disconnect plugs, on permanent installations, the respective fuse has to be turned off)!
- Only properly qualified electrical staff is allowed to install, connect and/or modify electrical equipment!

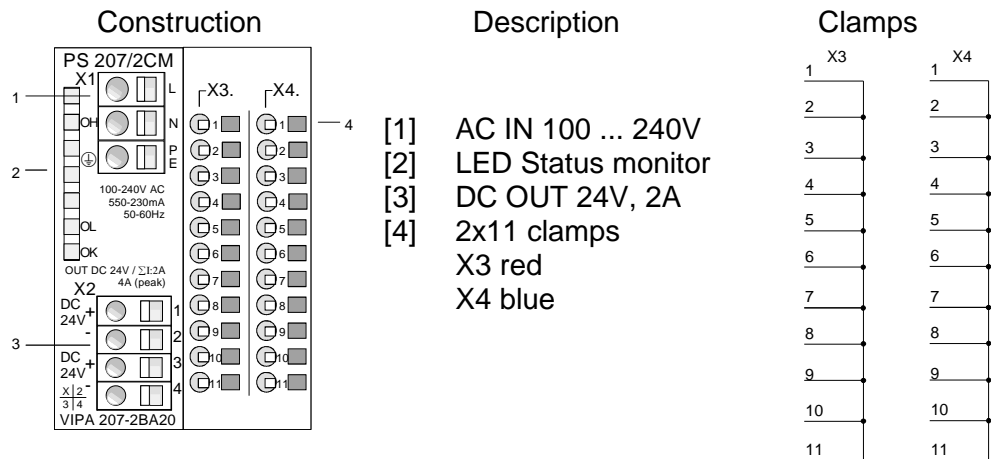
# PS 207/2CM - Power supply with Clamps - Construction

## Properties

The power supply is distinguished by the following properties:

- Wide-range input AC 100...240V without manual intervention
- Output voltage DC 24V, 2A, 48W
- Installable on a t-rail together with other System 200V components or as stand-alone device
- Protection from short-circuits, overload and open circuits
- Typically 90% efficiency at  $I_{rated}$
- Terminal module with 2x11 clamps

## Construction



## LED's

The front of the power supply carries 3 LEDs for troubleshooting purposes. The following table lists the significance and the respective color.

Name	Color	Description
OH	red	Overheat: turned on by excessive temperatures
OL	yellow	Overload: turned on when the total current exceeds the maximum capacity of app. 3A.
OK	green	Turned on when the power supply operates properly and supplies DC 24V power.



### Note!

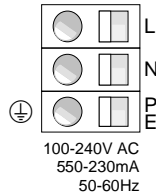
Only one LED is on at unit operation.

When all the LEDs are extinguished while the power supply is operational, a short circuit is present or the power supply has failed.



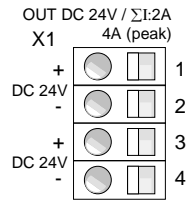
**Connector wiring**

**Input voltage  
INPUT  
AC 100...240V**



The power supply has to be connected to a source of AC power via the input connector. A fuse protects the input from overloads.

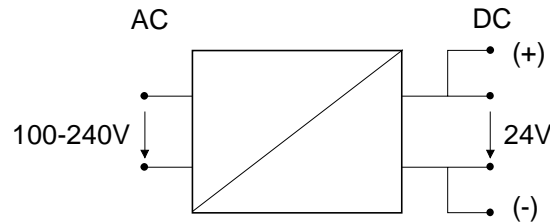
**Output voltage  
OUTPUT  
DC 24V, 4A**



Two connectors are provided for connection to System 200V modules that require an external source of DC 24V.

Both outputs are protected against short circuits protected and have an output voltage of DC 24V with a total current of max. 2A.

**Block diagram**



**Danger!**

- You need to disconnect the power supply from the main power source before commencing installation or maintenance work, i.e. before you start to work on a power supply or the supply cable the main supply line has to be disconnected (disconnect plugs, on permanent installations the respective fuse must be turned off)!
- Only properly qualified electrical staff is allowed to install, connect and/or modify electrical equipment!

## Installation

### Installation

The power supplies may be installed by two different methods:

- You may install the power supply along with System 200V modules on the same T-rail. In this case the power supply can only be installed at one end of your System 200V since the backplane bus would otherwise be interrupted.  
The power supplies are not connected to the backplane bus.
- Installed as stand-alone power supply on a T-rail.

Please ensure proper and sufficient ventilation for the power supply when you select the installation location.



### Danger!

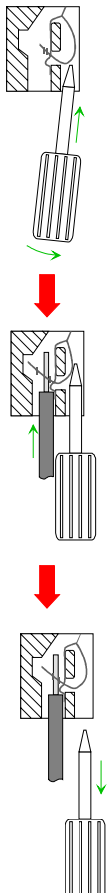
- The power supplies have to be installed in protected environments that are only accessible to properly qualified maintenance staff!
- You need to disconnect the power supply from the main power source before commencing installation or maintenance work, i.e. before you start to work on a power supply or the supply cable, the main supply line must be disconnected (disconnect plugs, on permanent installations, the respective fuse must be turned off)!
- Only properly qualified electrical staff is allowed to install, connect and/or modify electrical equipment!
- To provide a sufficient level of ventilation and cooling to the power supply components whilst maintaining the compact construction, it was not possible to protect the unit from incorrect handling and a proper level of fire protection. For this reason the required level of fire protection must be provided by the environment where the power supply is installed (e.g. installation in a switchboard that satisfies the fire protection rules and regulations)!
- Please adhere to the national rules and regulations of the location and/or country where the units are installed (installation, safety precautions, EMC ...).

## Wiring

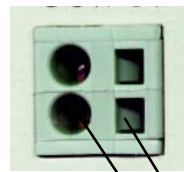
### Wiring

The connections to the power supply are provided by spring-clip terminals. The terminals are able to accommodate wires of a diameter from  $0.08\text{mm}^2$  to  $2.5\text{mm}^2$ . You may use flexible multi-strand wires as well as solid conductors.

### Wiring by means of spring-clip terminals



Connect cables to the spring-clip contacts as follows:



Square opening for screwdriver

Round opening for wires

The sequence shown on the left explains the steps that you have to follow to wire the power supply.

- Insert a suitable screwdriver at a slight angle into the square hole as shown.
- Push and hold the screwdriver in the opposite direction to open the spring contact.
- Insert the stripped end of the interconnecting wire into the round hole. You may use wires of a diameter of  $0.08\text{mm}^2$  to  $2.5\text{mm}^2$ .
- When you remove the screwdriver the inserted wire is clamped and connected securely by the spring-clip contact.



### Danger!

- You have to disconnect the power supply from the main power source before commencing installation or maintenance work, i.e. before you start to work on a power supply or the supply cable, the main supply line has to be disconnected (disconnect plugs, on permanent installations, the respective fuse must be turned off)!
- Only properly qualified electrical staff is allowed to install, connect and/or modify electrical equipment!

## Technical data

### Power supply

#### PS 207, 2A, 48W

Electrical data	VIPA 207-1BA00
Rated input voltage	AC 100...240V
Frequency	50Hz / 60Hz
Rated input current	0.24A / AC 230V
- power on surge	max. 30A
Buffer time (at a mains voltage AC $\geq 150V$ )	min. 10 ms
Rated output voltage	DC 24V $\pm$ 5 %
- residual ripple	< 100 mV <sub>ss</sub> incl. spikes
- open-circuit protection	yes
Rated output current	2A (48W); 4A (peak)
Efficiency	typ. 90% at I <sub>rated</sub>
Power dissipation	5W at nominal load
No load power	1.5W
Status indicators (LED)	via LEDs located on the front
Operating conditions	
Operating temperature	0°C...55°C (linear derating from 40°C to 55°C with 1.3W/°C)
Storage	- 25°C...+ 85°C
EMC	DIN EN 61000 / Part 4-8
Certification/CE	yes
General protection	Short circuit; overload; over temperature
Installation	IP 20
Terminals	DIN rail Spring-clip Input L, N, PE Output 2xDC 24V in parallel
Dimensions and Weight	
Dimensions (WxHxD)	25.4x76x76mm
Weight	250 g
Order data	
AC 100V-240V	VIPA 207-1BA00
DC 24V / 2A	

**Power supply**  
**PS 207/2CM,**  
**2A, 48W**

Electrical data	VIPA 207-2BA20
Rated input voltage	AC 100...240V
Frequency	50Hz / 60Hz
Rated input current	0.5A / AC 230V
- power on surge	max. 30A
Buffer time (at a mains voltage AC $\geq 150V$ )	min.10ms
Rated output voltage	DC 24V $\pm$ 5 %
- residual ripple	< 100 mV <sub>ss</sub> incl. spikes
- open circuit protection	yes
Rated output current	2A (48W);3A (peak)
Efficiency	typ. 90% at I <sub>rated</sub>
Power dissipation	5W at the nominal load
No load power	1.5W
Status indicators (LED)	via LEDs located on the front
Operating conditions	
Operating temperature	0°C...55°C (linear derating from 40°C to 55°C with 1.2W/°C)
Storage	- 25°C...+ 85°C
EMC	DIN EN 61000 / Part 4-8
Certification/CE	yes
General protection	Short circuit; overload; over temperature
Installation	IP 20
Terminals	DIN-rail Spring-clip Input L, N, PE Output 2xDC 24V in parallel
Terminal module	
Number of rows	2
Number of clamps per row	11
Maximum clamp current	10A
Dimensions and Weight	
Dimensions (WxHxD)	50.8x76x76mm
Weight	300 g
Ordering details	
AC 100V - 240V DC 24V / 2A	VIPA 207-2BA20



## Chapter 13 Digital input modules

### Overview

This chapter contains a description of the construction and the operating of the VIPA digital input modules.

Below follows a description of:

- A system overview of the digital input modules
- Properties
- Constructions
- Interfacing and schematic diagram
- Technical data

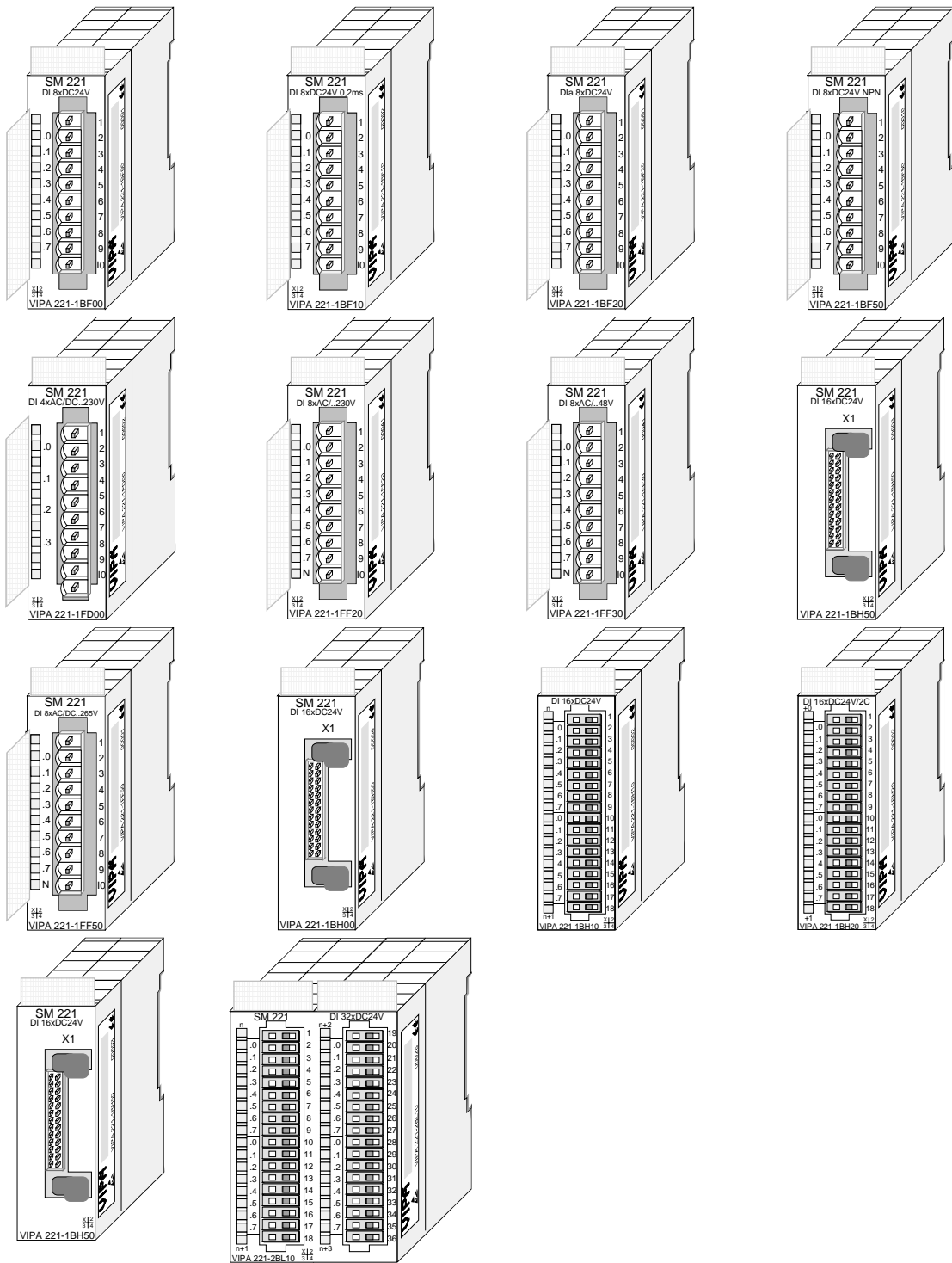
### Contents

Topic	Page
<b>Chapter 13 Digital input modules .....</b>	<b>13-1</b>
System overview .....	13-2
DI 8xDC 24V .....	13-4
DI 8xDC 24V 0.2ms .....	13-6
DIa 8xDC 24V .....	13-8
DI 8xDC 24V NPN .....	13-10
DI 4xAC/DC 90...230V .....	13-12
DI 8xAC/DC 60...230V .....	13-14
DI 8xAC/DC 24...48V .....	13-16
DI 8xAC 240V .....	13-18
DI 8xAC/DC 180...265V .....	13-20
DI 16xDC 24V with UB4x .....	13-22
DI 16xDC 24V .....	13-24
DI 16xDC24V/1C .....	13-26
DI 16xDC 24V NPN .....	13-36
DI 32xDC 24V .....	13-38

# System overview

## Input modules SM 221

Here follows a summary of the digital input modules that are currently available from VIPA:





**Order data  
input modules**

Type	Order number	Page
DI 8xDC 24V	VIPA 221-1BF00	13-4
DI 8xDC 24V 0.2ms	VIPA 221-1BF10	13-6
DIa 8xDC 24V	VIPA 221-1BF20	13-8
DI 8xDC 24V NPN	VIPA 221-1BF50	13-10
DI 4xAC/DC 90...230V	VIPA 221-1FD00	13-12
DI 8xAC/DC 60...230V	VIPA 221-1FF20	13-14
DI 8xAC/DC 24...48V	VIPA 221-1FF30	13-16
DI 8xAC 240V	VIPA 221-1FF40	13-18
DI 8xAC/DC 180...265V	VIPA 221-1FF50	13-20
DI 16xDC 24V with UB4x	VIPA 221-1BH00	13-22
DI 16xDC 24V	VIPA 221-1BH10	13-24
DI 16xDC 24V/1C	VIPA 221-1BH20	13-26
DI 16xDC 24V NPN	VIPA 221-1BH50	13-36
DI 32xDC 24V	VIPA 221-2BL10	13-38

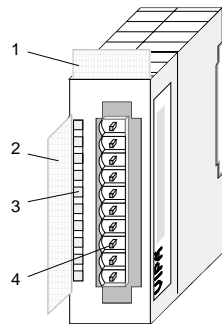
## DI 8xDC 24V

**Order data** DI 8xDC 24V VIPA 221-1BF00

**Description** The digital input accepts binary control signals from the process and provides an electrically isolated interface to the central bus system. The module has 8 channels, each one with a light emitting diode to indicate the status of the channel.

- Properties**
- 8 floating inputs, isolated from the backplane bus
  - DC 24V nominal input voltage
  - Suitable for standard switches and proximity switches
  - Status indicator for each channel by means of an LED

**Construction**

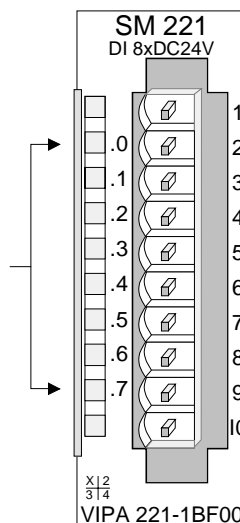


- [1] Label for module description
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Connector edge

**Status indicator pin assignment**

LED	Description	Pin	Assignment
-----	-------------	-----	------------

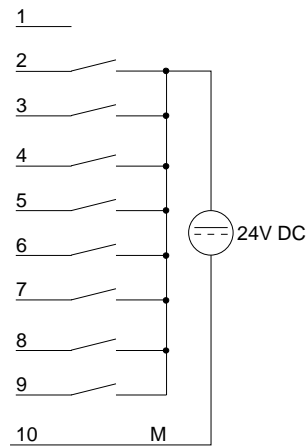
.0... .7	LEDs (green) E.0 to E.7 A "1" signal level is recognized as of app. 15V and the respective LED is turned on		
----------	---	--	--



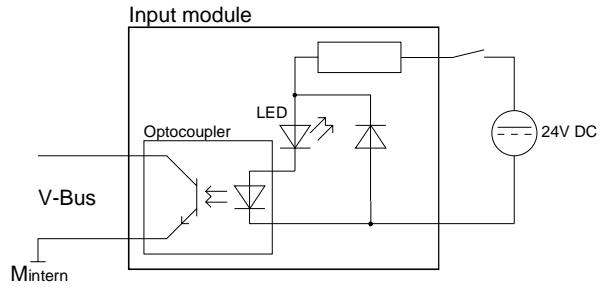
1	not connected
2	Input E.0
3	Input E.1
4	Input E.2
5	Input E.3
6	Input E.4
7	Input E.5
8	Input E.6
9	Input E.7
10	Ground

**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 221-1BF00
Number of inputs	8
Nominal input voltage	DC 24V (18 ... 28.8V)
Signal voltage "0"	0 ... 5V
Signal voltage "1"	15 ... 28.8V
Input filter time delay	3ms
Input current	typ. 7mA
Power supply	5V via backplane bus
Current consumption via backp. bus	20mA
Isolation	500Vrms (field voltage - backplane bus)
Status indicator	via LEDs located on the front
Programming specifications	
Input data	1Byte
Output data	-
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	50g

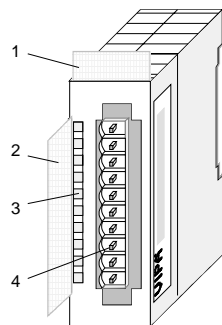
## DI 8xDC 24V 0.2ms

**Order data** DI 8xDC 24V 0.2ms VIPA 221-1BF10

**Description** The digital input accepts binary control signals from the process level and provides an electrically isolated interface to the central bus system. The module has 8 channels, each one with a light emitting diode to indicate the status of the channel.

- Properties**
- 8 floating inputs, isolated from the backplane bus
  - Delay time 0.2ms
  - DC 24V nominal input voltage
  - Suitable for standard switches and proximity switches
  - Status indicator for each channel by means of an LED

**Construction**

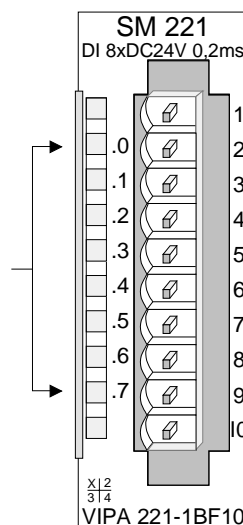


- [1] Label for module description
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

**Status indicator pin assignment**

LED	Description	Pin	Assignment
-----	-------------	-----	------------

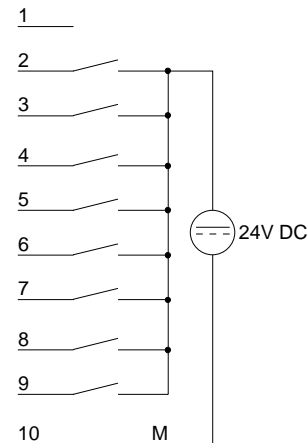
.0... .7	LEDs (green) E.0 to E.7 A "1" signal level is recognized as of app. 15V and the respective LED is turned on		
----------	---	--	--



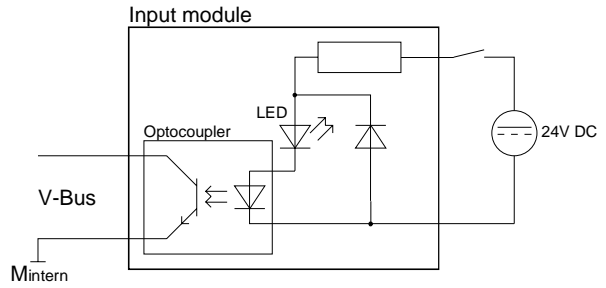
1	not connected
2	Input E.0
3	Input E.1
4	Input E.2
5	Input E.3
6	Input E.4
7	Input E.5
8	Input E.6
9	Input E.7
10	Ground

**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 221-1BF10
Number of inputs	8
Nominal input voltage	DC 24V (18 ... 28.8V)
Signal voltage "0"	0 ... 5V
Signal voltage "1"	15 ... 28.8V
Input filter time delay	0.2ms
Input current	typ. 7mA
Power supply	5V via backplane bus
Current consumption via backplane bus	20mA
Isolation	500Vrms (field voltage - backplane bus)
Status indicator	via LEDs located on the front
Programming specifications	
Input data	1Byte
Output data	-
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	50g

## D1a 8xDC 24V

**Order data** D1a 8xDC 24V VIPA 221-1BF20

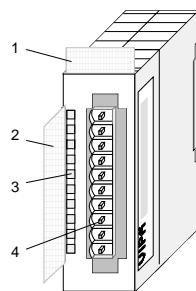
**Description** The digital alarm input module accepts the binary control signals from the process level and provides an electrically isolated interface to the central bus system.

All inputs are configurable as alarms. With the rising edge of the input, the alarm is activated. The alarm calls the OB 40 in the CPU. If this OB isn't available, the OB85 is called. If this OB is also not programmed, the CPU switches to STOP.

The module has 8 channels, each one with a light emitting diode to indicate the status of the channel.

- Properties**
- 8 alarm inputs, isolated from the backplane bus
  - nominal input voltage DC 24V
  - suited for urgent signals (switches and proximity switches)
  - Status indicator for each channel by means of an LED

**Construction**

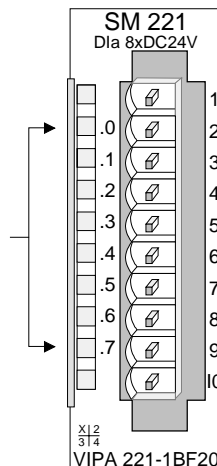


- [1] Label for module description
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

**Status indicator pin assignment**

**LED Description**

.0... .7 LEDs (green)  
E.0 to E.7  
A "1" signal level is recognized as of app. 15V and the respective LED is turned on

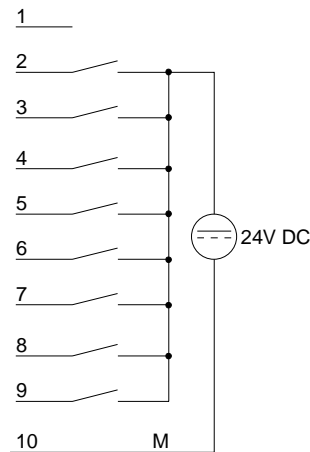


**Pin Assignment**

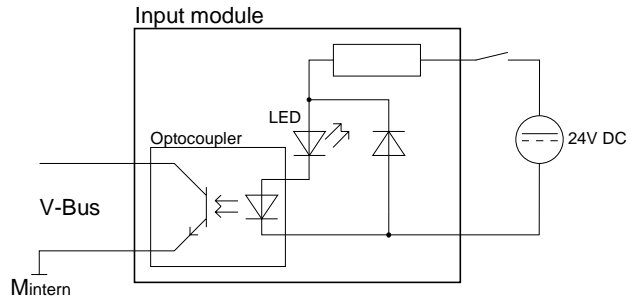
- 1 not connected
- 2 Input E.0
- 3 Input E.1
- 4 Input E.2
- 5 Input E.3
- 6 Input E.4
- 7 Input E.5
- 8 Input E.6
- 9 Input E.7
- 10 Ground

**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Note!**

The module may be deployed in the System 200V starting from CPU firmware versions:

- CPU 21x: Version 2.2.1
- CPU 24x: Version 3.0.6

The deployment with lower firmware versions causes error messages and a CPU switch to STOP!

**Technical data**

Electrical data	VIPA 221-1BF20
Number of alarm inputs	8
Nominal input voltage	DC 24V (18 ... 28.8V)
Signal voltage "0"	0 ... 5V
Signal voltage "1"	15 ... 28.8V
Input filter time delay	3ms
Input current	typ. 7mA
Power supply	5V via backplane bus
Current consumption via backplane bus	140mA
Isolation	500Vrms (field voltage - backplane bus)
Status indicator	via LEDs located on the front
Programming specifications	
Input data	1Byte
Output data	-
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	50g

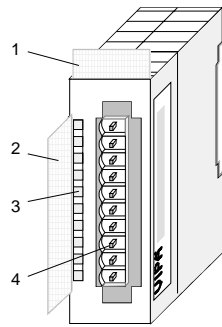
## DI 8xDC 24V NPN

**Order data** DI 8xDC 24V NPN VIPA 221-1BF50

**Description** The digital input accepts binary control signals from the process and provides an electrically isolated interface to the central bus system. The module has 8 channels, each one with a light emitting diode to indicate the status of the channel. The input becomes active when it is connected to ground.

- Properties**
- 8 floating inputs, isolated from the backplane bus
  - Active low input (signal level "1" when input is at ground)
  - DC 24V nominal input voltage
  - Suitable for standard switches and proximity switches
  - Status indicator for each channel by means of an LED

**Construction**

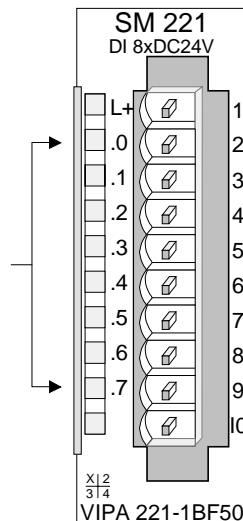


- [1] Label for module description
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

**Status indicator pin assignment**

<b>LED</b>	<b>Description</b>	<b>Pin</b>	<b>Assignment</b>
------------	--------------------	------------	-------------------

.0... .7 LEDs (green)  
E.0 to E.7  
when an input is at ground a "1" is detected and the respective LED is turned on

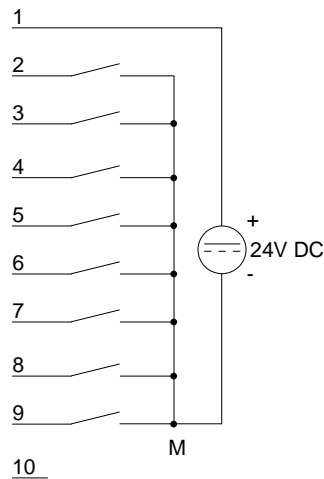


1	+DC 24V	2	Input E.0
3	Input E.1	4	Input E.2
5	Input E.3	6	Input E.4
7	Input E.5	8	Input E.6
9	Input E.7 / Ground	10	reserved

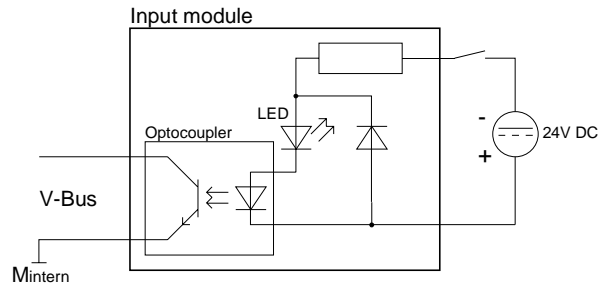


**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 221-1BF50
Number of inputs	8
Nominal input voltage	DC 24V (18 ... 28.8V)
Signal voltage "0"	15 ... 28.8V
Signal voltage "1"	0 ... 5V
Input filter time delay	3ms
Input current	typ. 7mA
Power supply	5V via backplane bus
Current consumption via backplane bus	20mA
Isolation	500Vrms (field voltage - backplane bus)
Status indicator	via LEDs located on the front
Programming specifications	
Input data	1Byte
Output data	-
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	50g

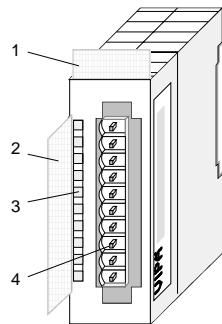
## DI 4xAC/DC 90...230V

**Order data** DI 4xAC/DC 90...230V VIPA 221-1FD00

**Description** The digital input accepts binary control signals from the process and provides an electrically isolated interface to the central bus system. The module has 4 channels and the respective status is displayed by means of LEDs.

- Properties**
- 4 floating inputs, isolated from the backplane bus and from each other
  - Status indicator for each channel by means of an LED
  - Nominal input voltage 90 ... 230V AC/DC

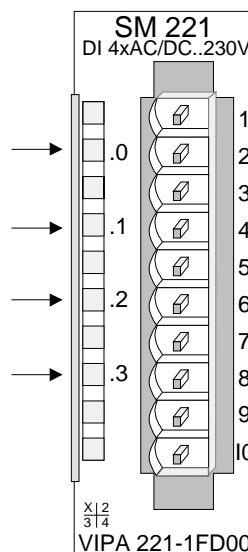
**Construction**



- [1] Label for module description
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

**Status indicator pin assignment**

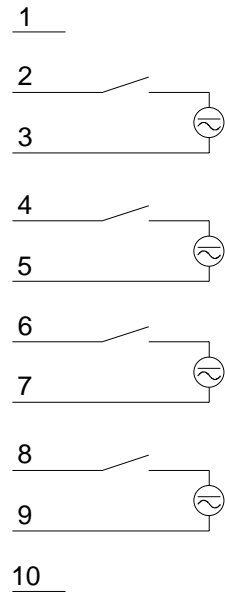
LED	Description
.0	LEDs (green)
.1	E.0 to E.3
.2	from app. DC 80V or AC 65V (50Hz) a signal "1" is detected and the respective LED is turned on
.3	



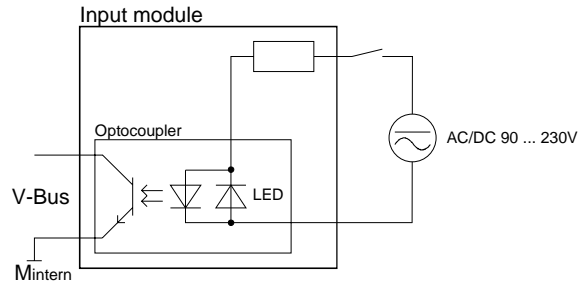
Pin	Assignment
1	not connected
2	E.0
3	Neutral conductor E.0
4	E.1
5	Neutral conductor E.1
6	E.2
7	Neutral conductor E.2
8	E.3
9	Neutral conductor E.3
10	not connected

**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 221-1FD00
Number of inputs	4
Nominal input voltage	AC/DC 90 ... 230V
Signal voltage "0"	AC/DC 0 ... 35V
Signal voltage "1"	AC/DC 90 ... 230V
Input filter time delay	25ms
Frequency of input voltage	50 ... 60Hz
Input resistance	136kΩ
Power supply	5V via backplane bus
Current consumption via backplane bus	80mA
Isolation	500Vrms (field voltage - backplane bus)
Status indicator	via LEDs located on the front
<b>Programming specifications</b>	
Input data	1Byte (Bit 0 ... Bit 3)
Output data	-
Parameter data	-
Diagnostic data	-
<b>Dimensions and weight</b>	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	50g

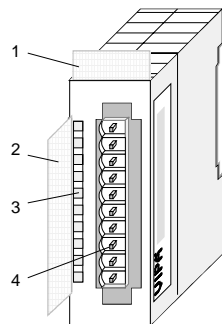
## DI 8xAC/DC 60...230V

**Order data** DI 8xAC/DC 60...230V VIPA 221-1FF20

**Description** The digital input accepts binary control signals from the process and provides an electrically isolated interface to the central bus system. The module has 8 channels, each one with a light emitting diode to indicate the status of the channel.

- Properties**
- 8 inputs, isolated from the backplane bus
  - Nominal input voltage 60 ... 230V AC/DC
  - Status indicator for each channel by means of an LED

**Construction**



- [1] Label for module description
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

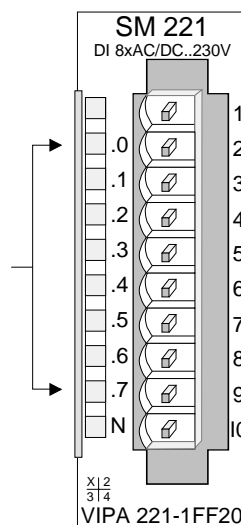
**Status indicator pin assignment**

**LED Description**

.0... .7 LEDs (green)  
E.0 to E.7  
from app. DC 55V or AC 45V (50Hz) a signal "1" is detected and the respective LED is turned on

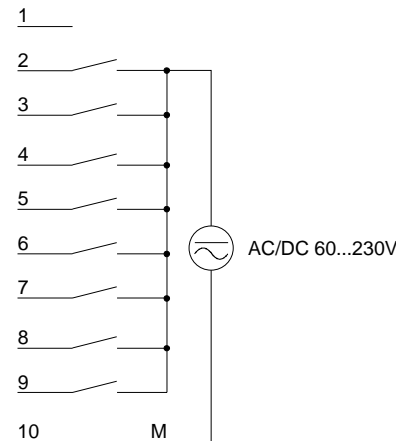
**Pin Assignment**

- 1 not connected
- 2 Input E.0
- 3 Input E.1
- 4 Input E.2
- 5 Input E.3
- 6 Input E.4
- 7 Input E.5
- 8 Input E.6
- 9 Input E.7
- 10 Neutral conductor

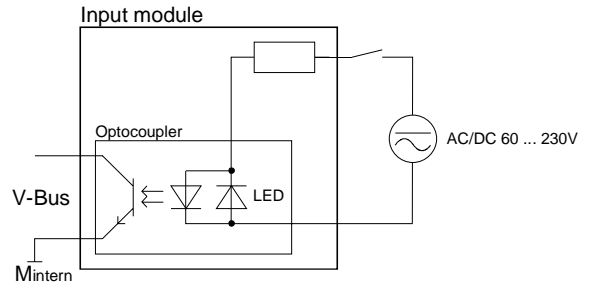


**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 221-1FF20
Number of inputs	8
Nominal input voltage	AC/DC 60 ... 230V
Signal voltage "0"	AC/DC 0 ... 35V
Signal voltage "1"	AC/DC 60 ... 230V
Input filter time delay	25ms
Frequency of input voltage	50 ... 60Hz
Input resistance	136kΩ
Power supply	5V via backplane bus
Current consumption via backplane bus	80mA
Isolation	500Vrms (field voltage - backplane bus)
Status indicator	via LEDs located on the front
Programming specifications	
Input data	1Byte
Output data	-
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	50g

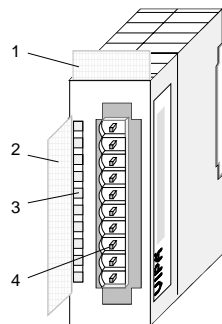
## DI 8xAC/DC 24...48V

**Order data** DI 8xAC/DC 24...48V VIPA 221-1FF30

**Description** The digital input accepts binary control signals from the process and provides an electrically isolated interface to the central bus system. The module has 8 channels, each one with a light emitting diode to indicate the status of the channel.

- Properties**
- 8 floating inputs, isolated from the backplane bus
  - Nominal input voltage AC/DC 24 ... 48V
  - Status indicator for each channel by means of an LED

**Construction**



- [1] Label for module description
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

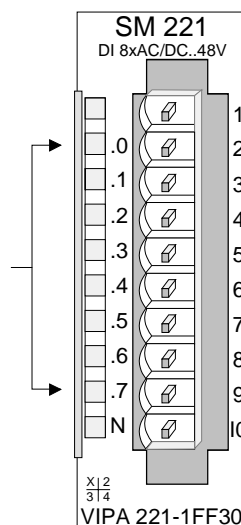
**Status indicator pin assignment**

**LED Description**

.0... .7 LEDs (green)  
E.0 to E.7  
from app. DC 14V or AC 12V (50Hz) a signal "1" is detected and the respective LED is turned on

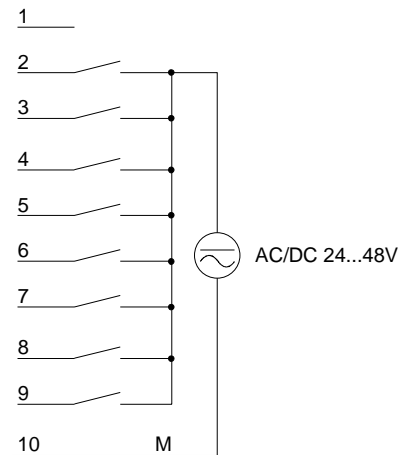
**Pin Assignment**

- 1 not connected
- 2 Input E.0
- 3 Input E.1
- 4 Input E.2
- 5 Input E.3
- 6 Input E.4
- 7 Input E.5
- 8 Input E.6
- 9 Input E.7
- 10 Neutral conductor

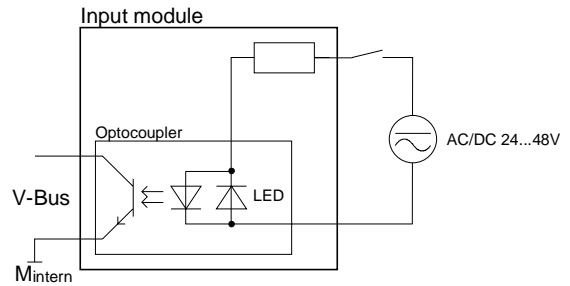


**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 221-1FF30
Number of inputs	8
Nominal input voltage	AC/DC 24 ... 48V
Signal voltage "0"	AC/DC 0 ... 8V
Signal voltage "1"	AC/DC 18 ... 48V
Input filter time delay	25ms
Frequency of input voltage	50 ... 60Hz
Input resistance	16.4kΩ
Power supply	5V via backplane bus
Current consumption via backplane bus	80mA
Isolation	500Vrms (field voltage - backplane bus)
Status indicator	via LEDs located on the front
Programming specifications	
Input data	1Byte
Output data	-
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	50g

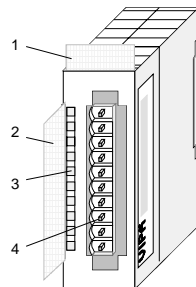
# DI 8xAC 240V

**Order data** DI 8xAC 240V VIPA 221-1FF40

**Description** The digital input accepts binary control signals from the process and provides an electrically isolated interface to the central bus system. The module has 8 channels, each one with a light emitting diode to indicate the status of the channel. In a defined voltage range, the signal state of the respective input is not modified (Hysteresis).

- Properties**
- 8 floating inputs, isolated from the backplane bus
  - Nominal input voltage AC 240V
  - Status indicator for each channel by means of an LED
  - Hysteresis
  - Current consumption 20mA per channel

**Construction**

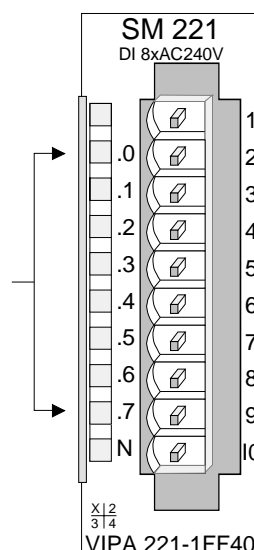


- [1] Label for module description
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

**Status indicator pin assignment**

LED	Description	Pin	Assignment
-----	-------------	-----	------------

.0... .7	LEDs (green) E.0 to E.7 from app. AC 190 V (50Hz) the signal "1" is detected and the respective LED is turned on		
----------	--	--	--

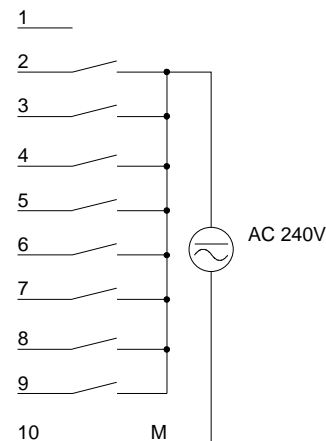


1	not connected
2	Input E.0
3	Input E.1
4	Input E.2
5	Input E.3
6	Input E.4
7	Input E.5
8	Input E.6
9	Input E.7
10	Neutral conductor

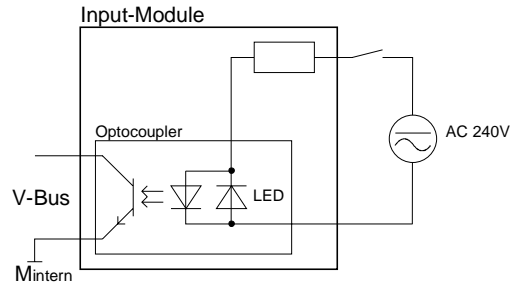


**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Note!**

This module is specified for voltages of max. AC 260V.

If inductive loads occur on the network, this load has to be filtered either directly at the module or at the according device, for example by using a snubber network.

**Technical data**

Electrical data	VIPA 221-1FF40
Number of inputs	8
Nominal input voltage	AC 240V
Current consumption per channel	20mA
Signal voltage "0"	AC 0 ... 70V
Hysteresis	AC 90 ... 160V
Signal voltage "1"	AC 190 ... 260V
Input filter time delay	25ms
Frequency of input voltage	50Hz
Input resistance	136kΩ
Power supply	5V via backplane bus
Current consumption via backplane bus	80mA
Isolation	500Vrms (field voltage - backplane bus)
Status indicator	via LEDs located on the front
<b>Programming specifications</b>	
Input data	1Byte
Output data	-
Parameter data	-
Diagnostic data	-
<b>Dimensions and weight</b>	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	ca. 50g

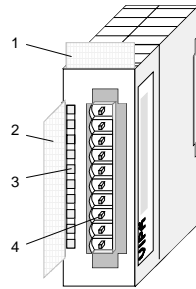
## DI 8xAC/DC 180...265V

**Order data** DI 8xAC/DC 180...265V VIPA 221-1FF50

**Description** The digital input accepts binary control signals from the process and provides an electrically isolated interface to the central bus system. The module has 8 channels, each one with a light emitting diode to indicate the status of the channel.

- Properties**
- 8 floating inputs, isolated from the backplane bus
  - Nominal input voltage AC/DC 180...265V
  - Status indicator for each channel by means of an LED

**Construction**

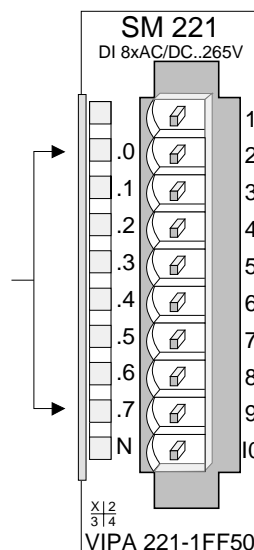


- [1] Label for module description
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

**Status indicator pin assignment**

LED	Description	Pin	Assignment
-----	-------------	-----	------------

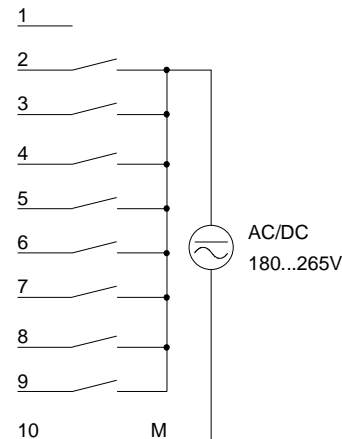
.0... .7	LEDs (green) E.0 to E.7 from app. DC 150V resp. AC 170V (50Hz) the signal "1" is detected and the respective LED is turned on		
----------	---	--	--



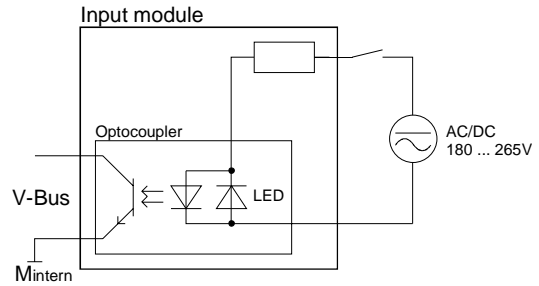
1	not connected
2	Input E.0
3	Input E.1
4	Input E.2
5	Input E.3
6	Input E.4
7	Input E.5
8	Input E.6
9	Input E.7
10	Neutral conductor

**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 221-1FF50
Number of inputs	8
Nominal input voltage	AC/DC 180...265V
Signal voltage "0"	AC/DC 0 ...150V
Signal voltage "1"	AC/DC 180 ... 265V
Input filter time delay	25ms
Frequency of input voltage	50 ... 60Hz
Input resistance	136kΩ
Power supply	5V via backplane bus
Current consumption via backplane bus	30mA
Isolation	500Vrms (field voltage - backplane bus)
Status indicator	via LEDs located on the front
Programming specifications	
Input data	1Byte
Output data	-
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	50g

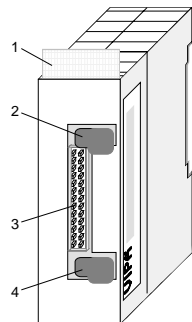
## DI 16xDC 24V with UB4x

**Order data** DI 16xDC 24V VIPA 221-1BH00

**Description** The digital input accepts binary control signals from the process and provides an electrically isolated interface to the central bus system. This module requires a UB4x-converter. It has 16 channels that indicate the respective status via LEDs on the UB4x. The module has to be connected to the converter module (DEA-UB4x) by means of a flattened round cable (DEA-KB91C).

- Properties**
- 16 inputs, isolated from the backplane bus
  - DC 24V nominal input voltage
  - Suitable for standard switches and proximity switches
  - Status indicator for each channel by means of a LED located on the conversion module UB4x

**Construction**



- [1] Label for module description
- [2] Clip
- [3] Recessed connector for the interface to a conversion module UB4x via the flattened round cable
- [4] Clip

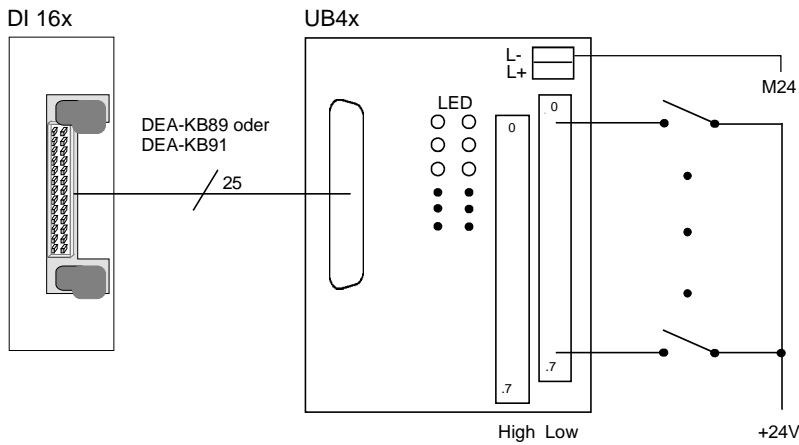
**Status indicator on UB4x**

LED	Description
0... .15	LEDs (yellow) E.0 to E.7 High E.0 to E.7 Low A "1" signal level is recognized as of app. 15V and the respective LED is turned on
L+ L-	LED (green) Supply voltage available

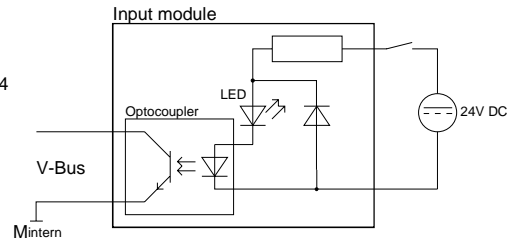
**Pin assignment module**

Connector	Pin	Assignment
	23...26	Supply voltage +DC 24V
	22	Input E.0
	21	Input E.1
	.	.
	.	.
	.	.
	8	Input E.14
	7	Input E.15
	1...6	Supply voltage Ground

Interface to UB4x



Schematic diagram module



Technical data

Electrical data	VIPA 221-1BH00
Number of inputs	16
Nominal input voltage	DC 24V (18 ... 28.8V)
Signal voltage "0"	0 ... 5V
Signal voltage "1"	15 ... 28.8V
Input filter time delay	3ms
Input current	typ. 7mA
Power supply	5V via backplane bus
Current consumption via backplane bus	20mA
Isolation	500Vrms (field voltage - backplane bus)
Status indicator	via LEDs located on the UB4x
Programming specifications	
Input data	2Byte
Output data	-
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	50g

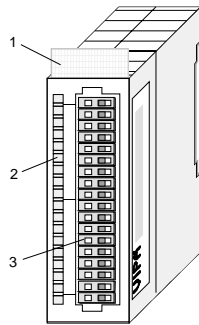
# DI 16xDC 24V

**Order data** DI 16xDC 24V VIPA 221-1BH10

**Description** The digital input accepts binary control signals from the process and provides an electrically isolated interface to the central bus system. It has 16 channels that indicate the respective status by means of LEDs.

- Properties**
- 16 inputs, isolated from the backplane bus
  - DC 24V nominal input voltage
  - Suitable for standard switches and proximity switches
  - Status indicator for each channel by means of an LED

**Construction**

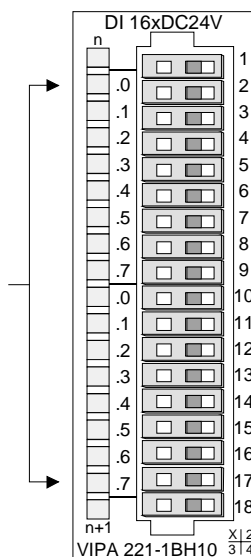


- [1] Label for module description
- [2] LED status indicator
- [3] Edge connector

**Status indicator connector assignment**

LED	Description	Pin	Assignment
-----	-------------	-----	------------

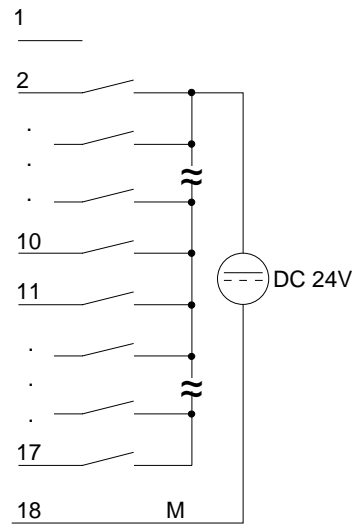
.0 ... .7	LEDs (green) E.0 to E.7 (per byte) A "1" signal level is recognized as of app. 15V and the respective LED is turned on		
-----------	--	--	--



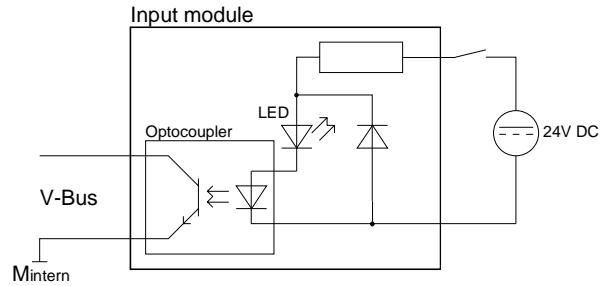
1	not connected
2	Input E.0
3	Input E.1
4	Input E.2
.	.
.	.
.	.
.	.
.	.
.	.
.	.
.	.
15	Input E.13
16	Input E.14
17	Input E.15
18	Ground

**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 221-1BH10
Number of inputs	16
Nominal input voltage	DC 24V (18 ... 28.8V)
Signal voltage "0"	0 ... 5V
Signal voltage "1"	15 ... 28.8V
Input filter time delay	3ms
Input current	typ. 7mA
Power supply	5V via backplane bus
Current consumption via backplane bus	30mA
Isolation	500Vrms (field voltage - backplane bus)
Status indicator	via LEDs located on the front
Programming specifications	
Input data	2Byte
Output data	-
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	50g

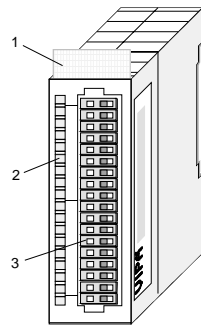
# DI 16xDC24V/1C

**Order data** DI 16xDC24V/1C VIPA 221-1BH20

**Description** The digital input accepts binary control signals from the process and provides an electrically isolated interface to the central bus system. It has 16 channels that indicate the respective status by means of LEDs. Additionally, the first two channels may head for counters.

- Properties**
- 16 inputs, isolated from the backplane bus
  - 2 inputs of this configurable as one counter, frequency or period measurement
  - Suitable for standard switches and proximity switches
  - Status indicator for each channel by means of an LED

**Construction**

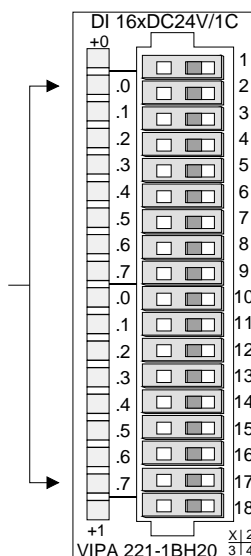


- [1] Label for module name
- [2] LED status indicator
- [3] Edge connector

**Status indicator connector assignment**

LED	Description	Pin	Assignment
-----	-------------	-----	------------

.0 ... .7	LEDs (green) E.0 to E.7 (per byte) A "1" signal level is recognized as of app. 15V and the respective LED is turned on		
-----------	--	--	--

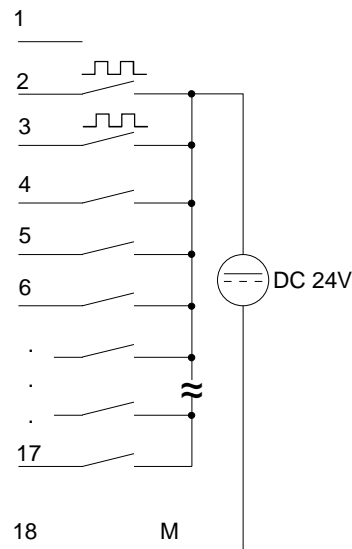


1	not connected
2	Input E.0 / Counter (A)
3	Input E.1 / Counter (B)
4	Input E.2
.	.
.	.
.	.
15	Input E.13
16	Input E.14
17	Input E.15
18	Ground

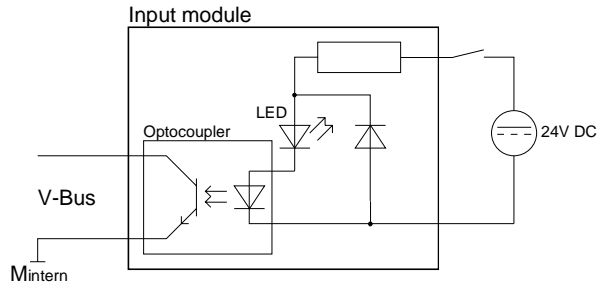


**Circuit and schematic diagram**

**Wiring diagram**



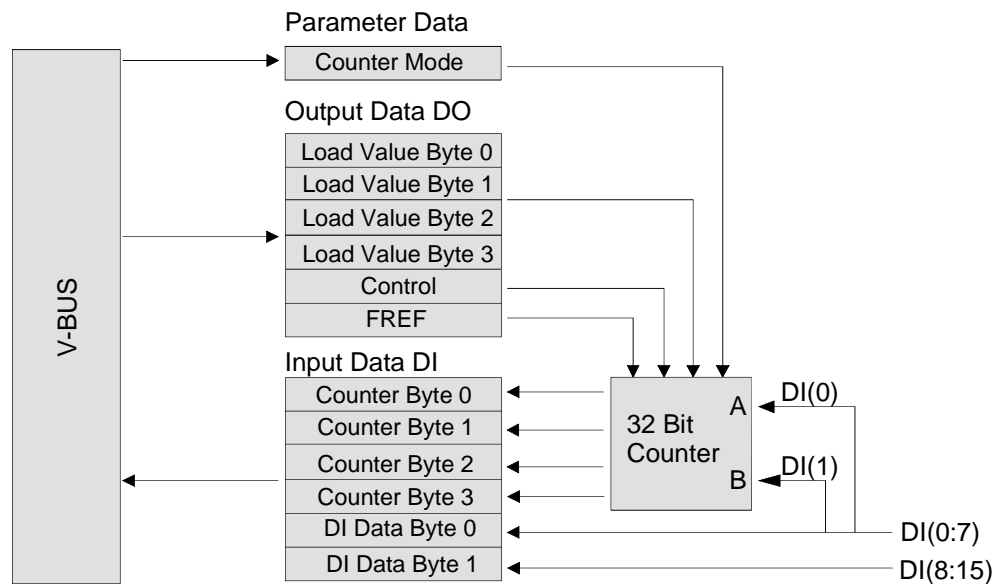
**Schematic diagram**



**Overview Module Functions**

The Counter Module is a 16Bit digital input module for System 200V combined with a one-channel 32Bit counter.

Inputs DI [0] and DI [1] are used as 'normal' process inputs and as counter inputs (signal A and signal B) simultaneously.



By writing DO data to the module, you may preset a counter value as well as a reference frequency. The activation of this values is via control byte.

By means of 1Byte parameter data, you may set the counter mode. There are 5 counter modes supported. By read access at the according bytes of the input data, the counter state is shown.

The counting is started res. stopped via the control byte (SW gate).

**Count Range /  
Limit values**

The counter module can count up and down. The count value is 32Bit wide and is to be interpreted as of type unsigned integer. Therefore the count limits are given as:

Lower Count Limit	Upper Count Limit
0	+ 4.294.967.295 $(2^{32} - 1)$

**Load Value**

It is possible to specify a load value for the counter. After loading the counter starts counting up res. down from this new value to the upper res. lower limit value. After receiving a new counting pulse, the counter jumps to the lower (counting up) res. upper limit (counting down) and starts the counting again.

In the operation mode "Frequency Measurement" the load value is used to define the time window of the measurement.

The load mechanism is controlled via the bit LOAD in the control byte.

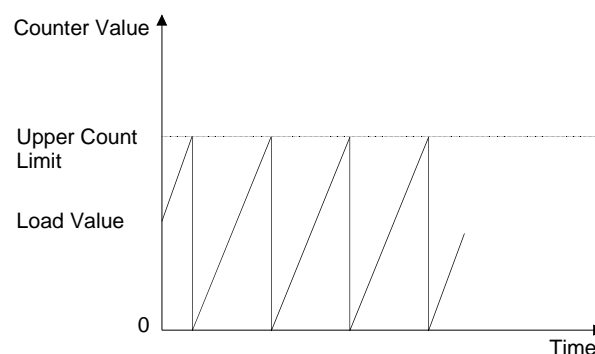
**Continuous  
Counting**

In all counter modes, a continuous counter function is used as described in the following paragraphs and as shown in figure.

If the counter reaches the upper count limit when counting up and a further count pulse is received, the counter jumps to the lower count limit and starts to add the count pulses again, meaning it counts continuously.

If the counter reaches the lower count limit when counting down and a further count pulse is received, the counter jumps to the upper count limit and continues to count down from there.

The count range in all modes is 0 to +4.294.967.295 and cannot be changed. The counter starts to count at 0 when a complete restart (Power-On Reset or VBUS-Reset) is executed on the module or the counter is cleared by setting bit CLEAR in the control byte.



**Counter activation via Software Gate**

Many applications require that the count can be started or stopped at a defined time depending on other events. This starting and stopping of the count process is done via a software gate function. If the gate is opened, count pulses can reach the counter and the count is started. If the gate is closed, count pulses can no longer reach the counter and the count is stopped.

The software gate is controlled via the bits START and STOP in the Control Byte. Setting the bit START will open the software gate whereas setting the bit STOP will close the software gate.

**Module access**

For input and output data, the module occupies each 6Byte in the address area. For setting the counter mode there are 1Byte parameter data at disposal.

Loading the counter res. presetting of a reference frequency is via a control byte by typing the wanted value into the output address area and setting the Bit 2 of the control byte to activate the counter.

You may see the counter value and the state of the inputs in the input address area. Also during count operation you may call all input channels.

**Input data  
DI data bytes**

The module has 6Byte input data that can be accessed by direct reading. Input bytes 0 to 3 are assigned to the 32Bit counter value whereas bytes 4 and 5 are assigned to the 16Bit digital inputs.

Byte	Bit 7 ... 0
0	Counter Value Byte 0
1	Counter Value Byte 1
2	Counter Value Byte 2
3	Counter Value Byte 3
4	DI Data Byte 0 (E.7 ... E.0)
5	DI Data Byte 1 (E.15 ... E.8)

**Output data**  
**DO data bytes**

The module has 6Byte output data.

Byte 0 to 3 are assigned to a load value according to the selected counter mode. Byte 4 is used as control byte for the counter.

Byte 5 selects a reference frequency for the counter modes "Frequency Measurement" and "Period Measurement".

Byte	Bit 7 ... 0
0	Load Value Byte 0
1	Load Value Byte 1
2	Load Value Byte 2
3	Load Value Byte 3
4	Control Byte
5	Reference Frequency

*Control Byte:*

Bit	Function
0	'1' = START counter (the software gate is open)
1	'1' = STOP counter (the software gate is closed)
2	'1' = LOAD counter
3	'1' = CLEAR counter
4	
...	reserved
7	

*Reference Frequency Selection:*

Value	Reference Frequency
00h	16 MHz
01h	8 MHz
02h	4 MHz
03h	1 MHz
04h	100 kHz
05h	10 kHz
06h	1 kHz
07h	100 Hz
others	not allowed

**Parameter Data** The module has 1Byte parameter data for selecting the counter mode.

Byte	Bit 7 ... 0
0	Counter Mode

*Counter Mode Selection:*

Value	Counter Mode
00h	Quadruple Pulse Evaluation
01h	Pulse and Direction Evaluation
02h	Clock Up / Clock Down Evaluation
03h	Frequency Measurement
04h	Period Measurement
others	not allowed

### Counter Modes

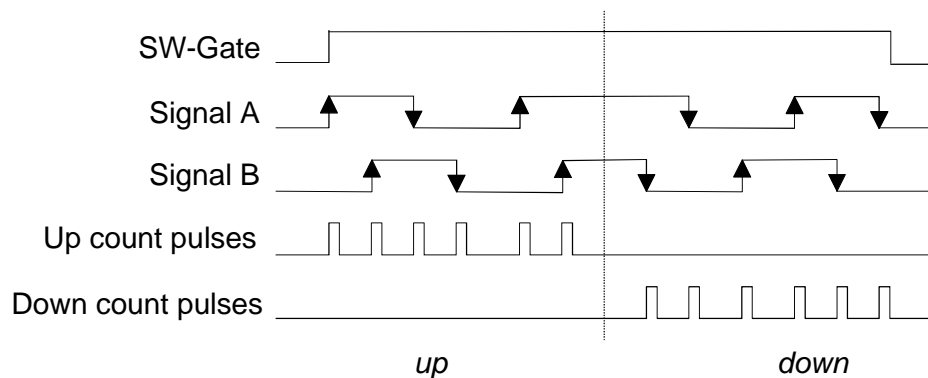
#### Quadruple Pulse Evaluation (Mode 00h)

Quadruple evaluation means that the rising and falling edges of A and B are evaluated; whether up or down count pulses are generated depends on which channel hurries first.

In this counting mode E.0 and E.1 have the following assignment and function:

E.0 as channel A: If channel A hurries in front, the counter counts up.

E.1 as channel B: If channel B hurries in front, the counter counts down.



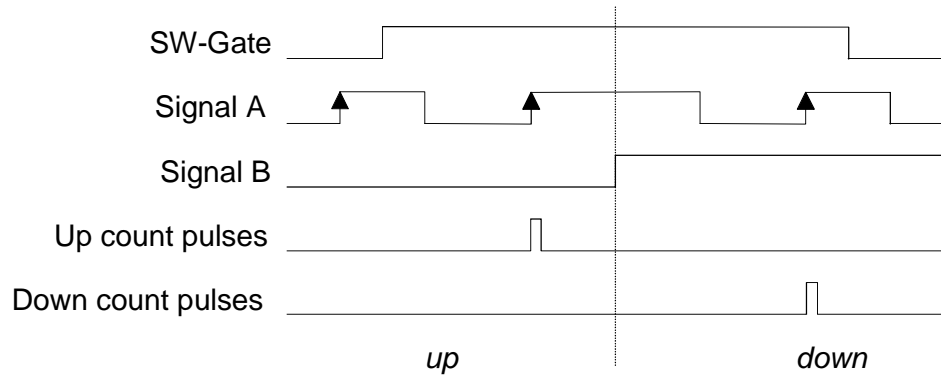
**Pulse and Direction Evaluation Mode (01h)**

Every rising edge of A is evaluated. Channel B defines the counter direction.

In this counting mode E.0 and E.1 have the following assignment and function:

E.0 as channel A: Clock pulse for the counter at rising edge.

E.1 as channel B: Defines the counter direction (0 = up, 1 = down)



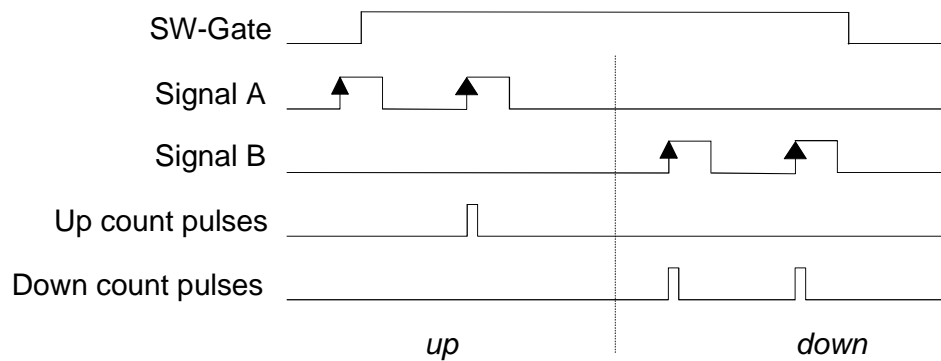
**Clock Up / Clock Down Evaluation (Mode 02h)**

The rising edges of channel A and B are evaluated. The counter is incremented with every rising edge of A and decremented with every rising edge of B.

In this counting mode E.0 and E.1 have the following assignment and function:

E.0 as channel A: Clock up pulse for the counter at rising edge.

E.1 as channel B: Clock down pulse for the counter at rising edge.



**Frequency Measurement (Mode 03h)**

In frequency measurement mode, the module counts the number of rising edges of channel A received within a specified time window. Channel B is not used in this mode.

The time window  $T_w$  is specified indirectly by selecting a reference frequency with DO byte 5 and defining a load value in DO bytes 0 to 3:

$$T_w = \frac{1}{\text{Reference Frequency}} * \text{Load Value}$$

By setting the Bit 2 of the control byte, the time window is transferred. When the counter is enabled (software gate is open), the reference counter is started with the first rising edge of channel A and is incremented with every rising edge of the reference clock.

When the reference counter reaches the load value (time  $T_w$  has expired), the current counter value is copied to DI byte 0 to 3 and can be read.

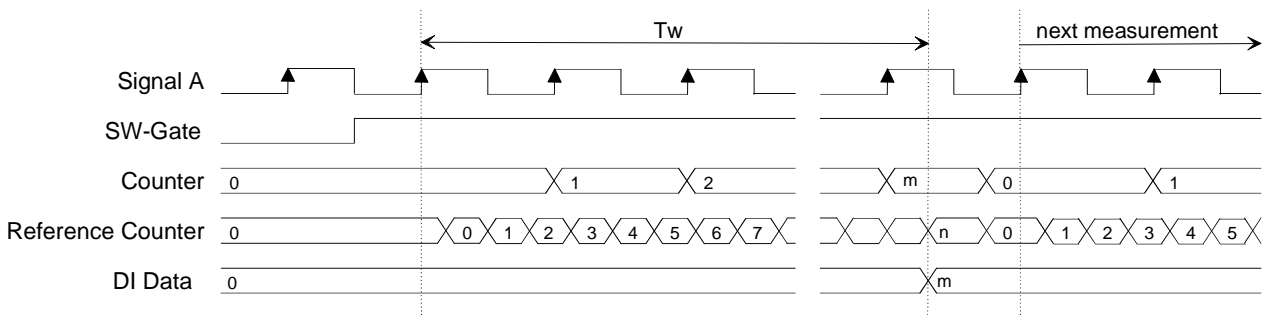
Then the counter and the reference counter is cleared automatically and the next frequency measurement is started with the next rising edge of channel A. If there aren't at least two rising edges of channel A within the time window  $T_w$ , the counter value will be read as 0 for this measurement.

Frequency measurement is started and ended by using the software gate, that is as long as the software gate is open, the frequency of channel A is measured.

The counter can be cleared at any time by CLEAR='1' in the Control Byte while the load value stays valid until a new value is loaded or a Reset is detected.

The recent frequency can be computed by using the following formula:

$$\text{Frequency} = \text{Reference Frequency} * \frac{\text{Counter Value}}{\text{Load Value}}$$



**Example:** Reference Frequency : 1 MHz  
 Load Value (n) : 1.000.000  
 Counter Value (m) : 10.000

$$\text{Frequency} = 1 \text{ MHz} * \frac{10.000}{1.000.000} = 10 \text{ kHz}$$

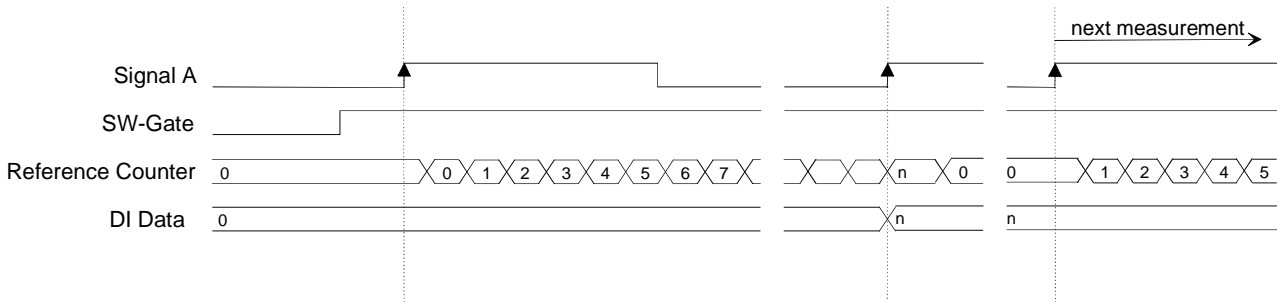
**Period Measurement (Mode 04h)**

With very small frequencies, it is convenient to measure the period instead of the frequency. In the operating mode "Period Measurement", the time between two rising edges of channel A is measured by counting the number of rising edges of the selected reference clock occurring between two rising edges of channel A. Channel B is not used in this mode.

Period measurement is started and ended by using the software gate, that is: as long as the software gate is open the period of channel A is measured continuously. The counter can be cleared at any time by CLEAR='1' in the Control Byte. The period measurement will then start again with the next rising edge of channel A.

The recent signal period can be computed by using the following formula:

$$Period = \frac{1}{Reference\ Frequency} * Counter\ Value$$



**Example:** Reference Frequency : 1 MHz  
Counter Value (n) : 10.000

$$Period = \frac{1}{1\ MHz} * 10.000 = 10\ ms$$



**Note!**

The counter value stays valid until the next measurement is completed or the counter is cleared.

If the next measurement is never completed (e.g. because the second rising edge of channel A never occurs), you will always see the 'old' counter value and not the current value of the Reference Counter.



**Technical data**

Electrical data	VIPA 221-1BH20
Number of inputs	16
Counter	1 (2 inputs A, B)
Rated input voltage	DC 24V (18 ... 28.8V)
Signal voltage "0"	0 ... 5V
Signal voltage "1"	15 ... 28.8V
Input filter time delay	3ms
Input filter counter	100µs
Max. frequency	100kHz
Input current	typ. 7mA
Power supply	5V via backplane bus
Current consumption via backplane bus	100mA
Isolation	500Vrms (field voltage - backplane bus)
Status indicator	via LEDs located on the front
Programming specifications	
Input data	6 Byte
Output data	6 Byte
Parameter data	1 Byte
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4 x 76 x 76
Weight	50g

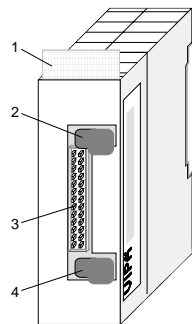
## DI 16xDC 24V NPN

**Order data** DI 16xDC 24V NPN VIPA 221-1BH50

**Description** The digital input accepts binary control signals from the process and provides an electrically isolated interface to the central bus system. The input becomes active when it is connected to ground.

- Properties**
- 16 inputs, isolated from the backplane bus
  - Active low input (signal level "1" when input is at ground)
  - DC 24V nominal input voltage
  - Suitable for standard switches and proximity switches

**Construction**



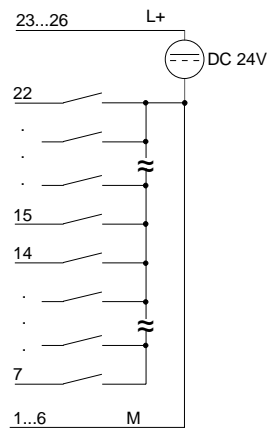
- [1] Label for module description
- [2] Clip
- [3] Recessed connector for the interface input
- [4] Clip

**Pin assignment**

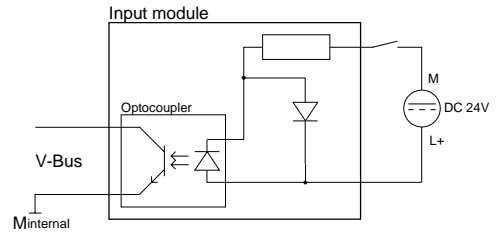
Connector	Pin	Assignment
	23...26	Supply voltage +DC 24V
	22	Input E.0
	21	Input E.1
	.	.
	.	.
	.	.
	8	Input E.14
	7	Input E.15
	1...6	Supply voltage Ground

**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 221-1BH50
Number of inputs	16
Nominal input voltage	DC 24V (18 ... 28.8V)
Signal voltage "0"	15 ... 28.8V
Signal voltage "1"	0 ... 5V
Input filter time delay	3ms
Input current	typ. 7mA
Power supply	5V via backplane bus
Current consumption via backplane bus	20mA
Isolation	500Vrms (field voltage - backplane bus)
Status indicator	-
Programming specifications	
Input data	2Byte
Output data	-
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	50g

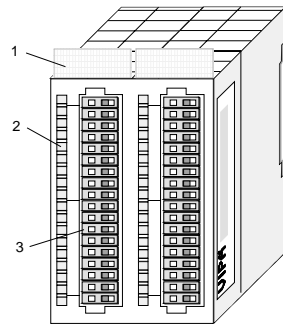
# DI 32xDC 24V

**Order data** DI 32xDC 24V VIPA 221-2BL10

**Description** The digital input accepts binary control signals from the process and provides an electrically isolated interface to the central bus system. It has 32 channels that indicate the respective status by means of LEDs.

- Properties**
- 32 inputs, isolated from the backplane bus
  - DC 24V nominal input voltage
  - Suitable for standard switches and proximity switches
  - Status indicator for each channel by means of an LED

**Construction**

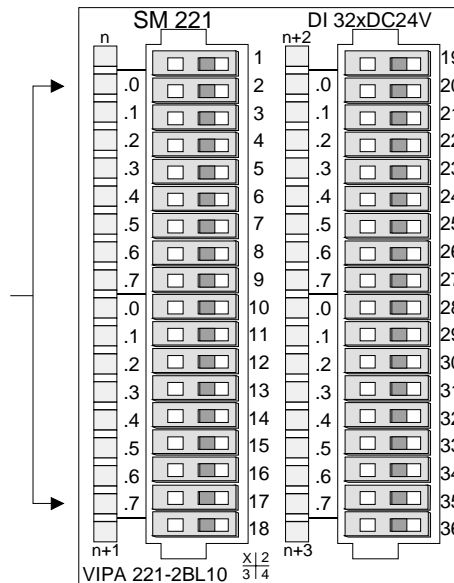


- [1] Label for module description
- [2] LED status indicator
- [3] Edge connector

**Status indicator pin assignment**

LED	Description	Pin	Assignment
-----	-------------	-----	------------

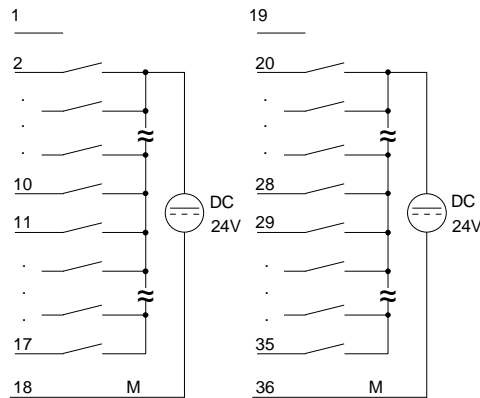
.0 ... .7	LEDs (green) E.0 to E.7 (per byte) A "1" signal level is recognized as of app. 15V and the respective LED is turned on		
-----------	--	--	--



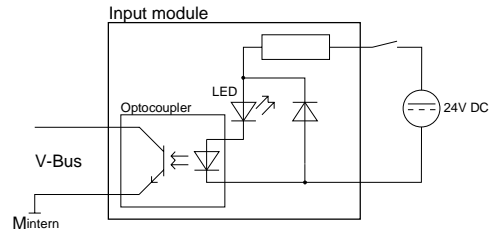
1	Not connected
2...17	Input E.0...E.15
.	.
.	.
.	.
18	Ground
19	Not connected
.	.
.	.
.	.
20 ... 35	Input E.16...E.31
36	Ground

**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 221-2BL10
Number of inputs	32
Nominal input voltage	DC 24V (18 ... 28.8V)
Signal voltage "0"	0 ... 5V
Signal voltage "1"	15 ... 28.8V
Input filter time delay	3ms
Input current	typ. 7mA
Power supply	5V via backplane bus
Current consumption via backplane bus	50mA
Isolation	in 2 groups of 16 inputs each 500Vrms (field voltage - backplane bus)
Status indicator	via LEDs located on the front
Programming specifications	
Input data	4Byte
Output data	-
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	50.8x76x76
Weight	50g



## Chapter 14 Digital output modules

### Overview

This chapter contains a description of the construction and the operation of the VIPA digital output modules.

Below follows a description of:

- A system overview of the digital output modules
- Properties
- Construction
- Interfacing and schematic diagrams
- Technical data

### Content

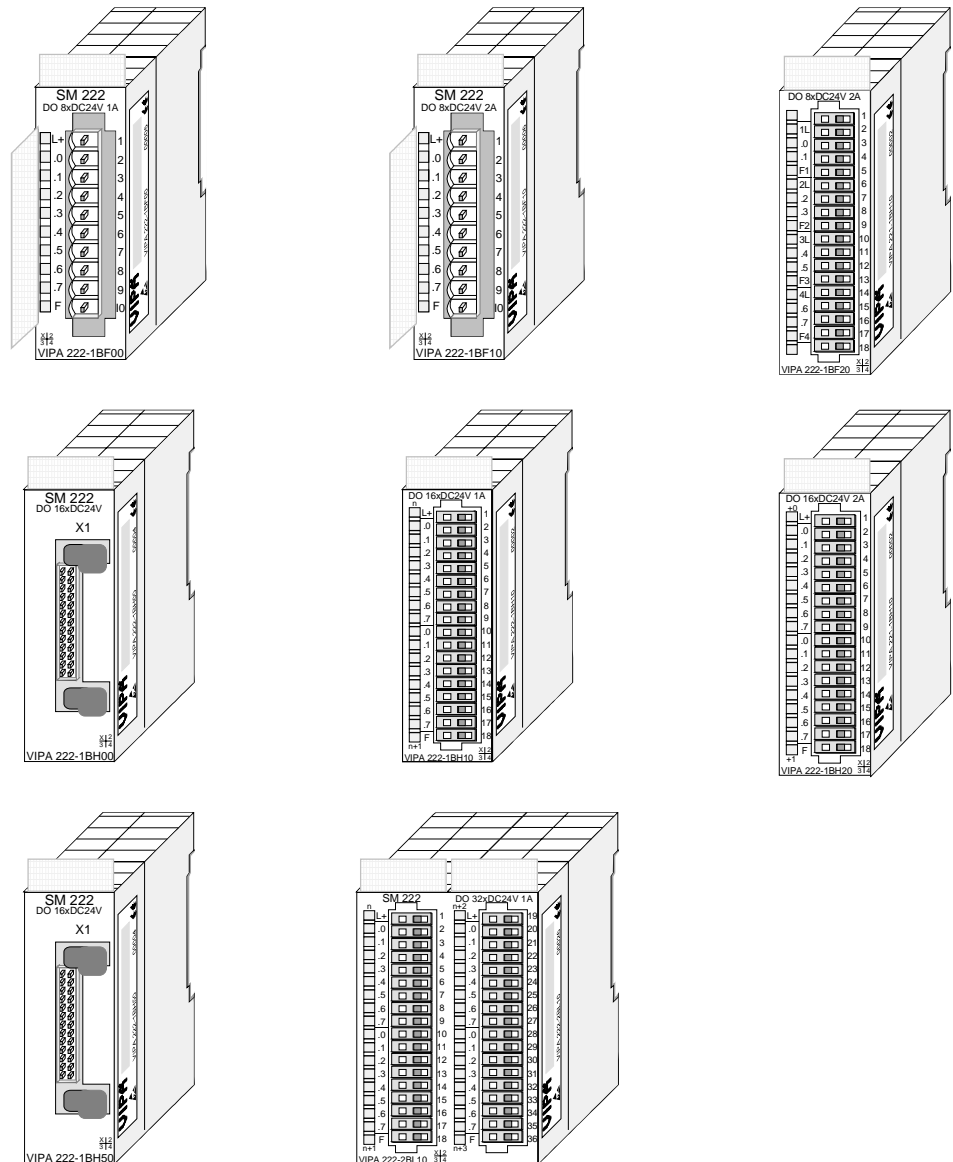
Topic	Page
<b>Chapter 14 Digital output modules</b> .....	<b>14-1</b>
System overview.....	14-2
DO 8xDC 24V 1A.....	14-4
DO 8xDC 24V 2A.....	14-6
DO 8xDC 24V 2A separated 4 á 2.....	14-8
DO 16xDC 24V 0.5A with UB4x.....	14-10
DO 16xDC 24V 1A.....	14-12
DO 16xDC 24V 2A.....	14-14
DO 16xDC 24V 0.5A NPN.....	14-16
DO 32xDC 24V 1A.....	14-18
DO 8xRelay COM.....	14-20
DO 4xRelay.....	14-22
DO 4xRelay bistable.....	14-24
DO 8xSolid State COM.....	14-26
DO 4xSolid State.....	14-28

## System overview

### Output modules SM 222

Here follows a summary of the digital output modules that are currently available from VIPA:

#### DC 24V output modules

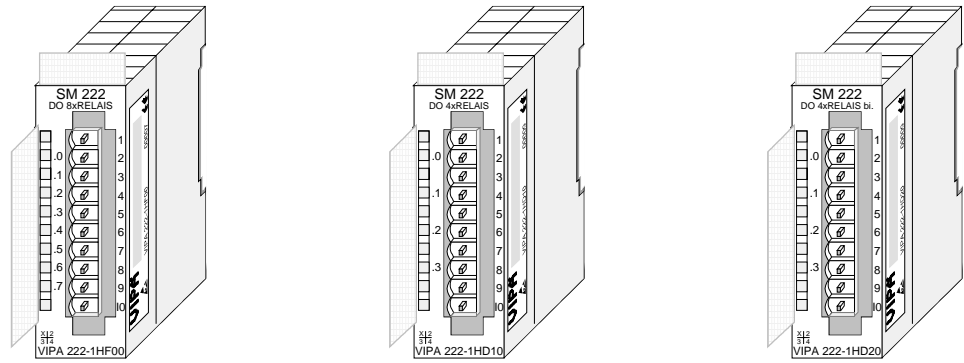


#### Order data DC 24V output modules

Type	Order number	Page
DO 8xDC 24V 1A	VIPA 222-1BF00	14-4
DO 8xDC 24V 2A	VIPA 222-1BF10	14-6
DO 8xDC 24V 2A floating 4 á 2	VIPA 222-1BF20	14-8
DO 16xDC 24V 0.5A with UB4x	VIPA 222-1BH00	14-10
DO 16xDC 24V 1A	VIPA 222-1BH10	14-12
DO 16xDC 24V 2A	VIPA 222-1BH20	14-14
DO 16xDC 24V 0,5A NPN	VIPA 222-1BH50	14-16
DO 32xDC 24V 1A	VIPA 222-2BL10	14-18



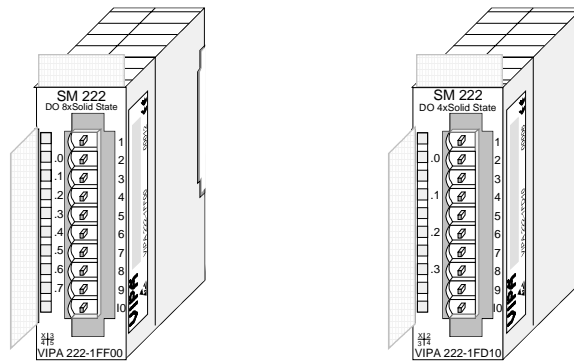
**Relay output modules**



**Order data relay output modules**

Type	Order number	Page
DO 8xRelay COM	VIPA 222-1HF00	14-20
DO 4xRelay	VIPA 222-1HD10	14-22
DO 4xRelay bistable	VIPA 222-1HD20	14-24

**Solid-state output modules**



**Order data solid-state output modules**

Type	Order number	Page
DO 8xSolid State COM	VIPA 222-1FF00	14-26
DO 4xSolid State	VIPA 222-1FD10	14-28

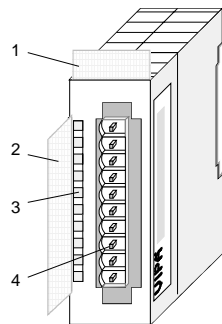
## DO 8xDC 24V 1A

**Order data** DO 8xDC 24V 1A VIPA 222-1BF00

**Description** The digital output module accepts binary control signals from the central bus system and transfers them to the process level via outputs. The module requires a supply of DC 24V via the front-facing connector. It provides 8 channels and the status of each channel is displayed by means of an LED.

- Properties**
- 8 outputs, isolated from the backplane bus
  - DC 24V supply voltage
  - 1A output current
  - Suitable for magnetic valves and DC contactors
  - LEDs for supply voltage and error message
  - Active channel indication by means of an LED

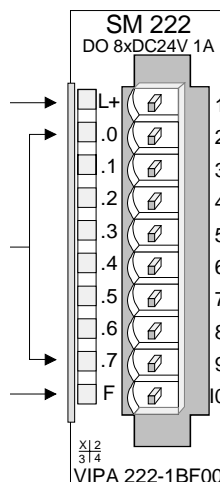
**Construction**



- [1] Label for module description
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

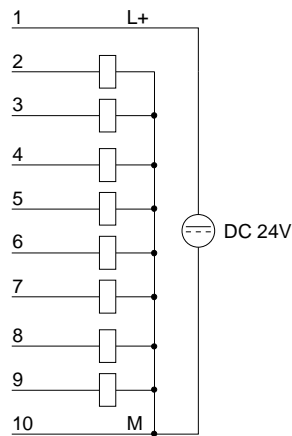
**Status indicator pin assignment**

LED	Description	Pin	Assignment
L+	LED (yellow) Supply voltage available	1	DC 24V supply voltage
.0... .7	LEDs (green) A.0 to A.7 when an output is active the respective LED is turned on	2-9	Output A.0 to A.7
F	LED (red) Overload, overheat or short circuit error	10	Supply ground

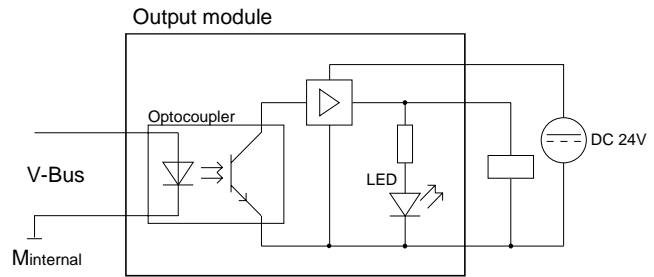


**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 222-1BF00
Number of outputs	8
Nominal load voltage	DC 24V (18...35V) from ext. power supply
No-load current consumption at L+ (all A.x=off)	10mA
Output current per channel	1A protected against sustained short circuits
Total current of all 8 channels	8A
Current consumption via backplane bus	50mA
Voltage supply	5V via backplane bus
Isolation	500Vrms (field voltage - backplane bus)
Status indicator	via LEDs located on the front
<b>Programming specifications</b>	
Input data	-
Output data	1 Byte
Parameter data	-
Diagnostic data	-
<b>Dimensions and weight</b>	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	50g

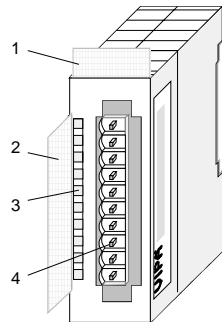
# DO 8xDC 24V 2A

**Order data** DO 8xDC 24V 2A VIPA 222-1BF10

**Description** The digital output module accepts binary control signals from the central bus system and transfers them to the process level via outputs. The module requires a DC 24V supply via the connector located on the front. It provides 8 channels and the status of each channel is displayed by means of an LED. The maximum load current per output is 2A.

- Properties**
- 8 outputs, isolated from the backplane bus
  - DC 24V supply voltage
  - Output current 2A
  - Suitable for magnetic valves and DC contactors
  - LEDs for supply voltage and error message
  - Active channel indication by means of an LED

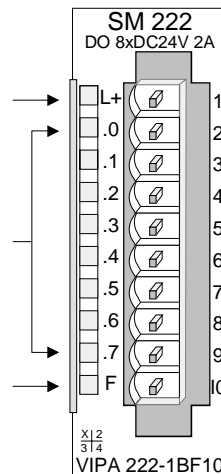
**Construction**



- [1] Label for module description
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

**Status indicator pin assignment**

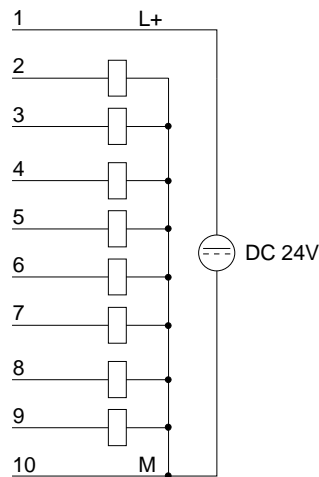
LED	Description
L+	LED (yellow) Supply voltage available
.0... .7	LEDs (green) A.0 to A.7 when an output becomes active the respective LED is turned on
F	LED (red) Overload, overheat, short circuit error



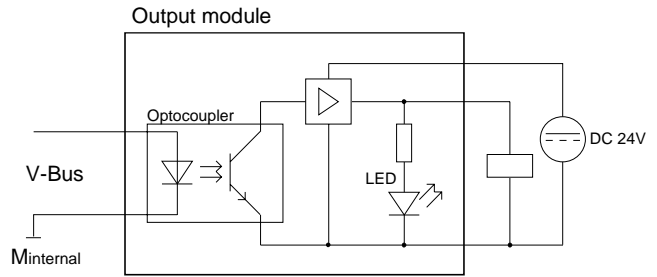
Pin	Assignment
1	DC 24V supply voltage
2	Output A.0
3	Output A.1
4	Output A.2
5	Output A.3
6	Output A.4
7	Output A.5
8	Output A.6
9	Output A.7
10	Supply ground

**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 222-1BF10
Number of outputs	8
Nominal load voltage	DC 24V (18...35V) from ext. power supply
No-load current consumption at L+ (all A.x=off)	10mA
Output current per channel	2A protected against sustained short circuits
Total current of all 8 channels	max. 12A
Diversity factor	$I_D=50\%$ (8A)
Current consumption via backplane bus	50mA
Voltage supply	5V via backplane bus
Isolation	500Vrms (field voltage - backplane bus)
Status indicator	via LEDs located on the front
<b>Programming specifications</b>	
Input data	-
Output data	1Byte
Parameter data	-
Diagnostic data	-
<b>Dimensions and weight</b>	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	50g

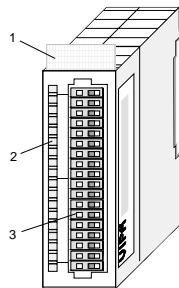
## DO 8xDC 24V 2A separated 4 á 2

**Order data** DO 8xDC 24V 2A VIPA 222-1BF20

**Description** The digital output module accepts binary control signals from the central bus system and transfers them to the process level via outputs. The module requires a DC 24V supply via the connector located on the front. It provides 8 channels and the status of each channel is displayed by means of an LED. The maximum load current per output is 2A.

- Properties**
- 8 outputs, isolated from the backplane bus
  - Potential separation in 4 groups á 2 outputs
  - DC 24V supply voltage
  - Output current 2A
  - Suitable for magnetic valves and DC contactors
  - LEDs for supply voltage and error message
  - Active channel indication by means of an LED

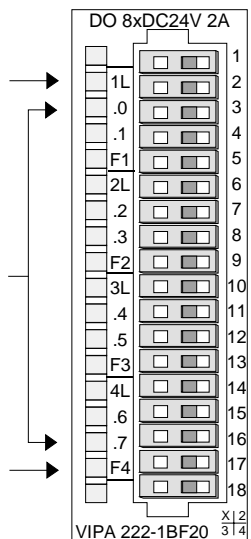
**Construction**



- [1] Label for module description
- [2] LED status indicator
- [3] Edge connector

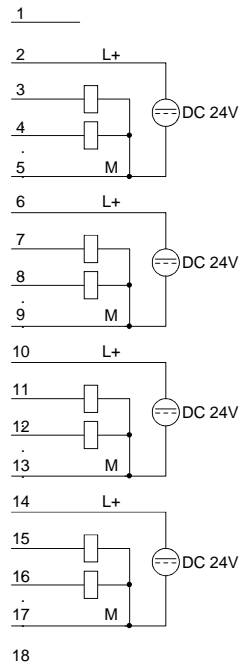
**Status indicator pin assignment**

LED	Description	Pin	Assignment
1L...4L	LED (yellow) Supply voltage available	1	not used
	LEDs (green)	2	Supply voltage 1L+
.0... .7	A.0 to A.7 (green) when an output becomes active the respective LED is turned on	3	Output A.0
		4	Output A.1
F1...F4	LED (red) Overload, overheat, short circuit error	5	Ground 1M
		6	Supply voltage 2L+
		7	Output A.2
		8	Output A.3
		9	Ground 2M
		...	...
		14	Supply voltage 4L+
		15	Output A.6
		16	Output A.7
		17	Ground 4M
		18	not used

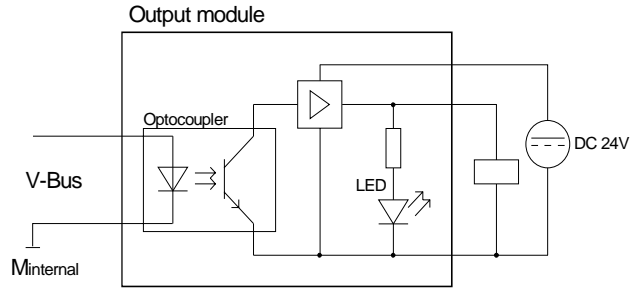


**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 222-1BF20
Number of outputs	8
Nominal load voltage	DC 24V (18...35V) from ext. power supply
No-load current consumption at L+ (all A.x=off)	10mA
Output current per channel	2A protected against sustained short circuits
Current consumption via backplane bus	50mA
Voltage supply	5V via backplane bus
Isolation	500Vrms (field voltage - backplane bus)
Status indicator	via LEDs located on the front
Programming specifications	
Input data	-
Output data	1Byte
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	50g

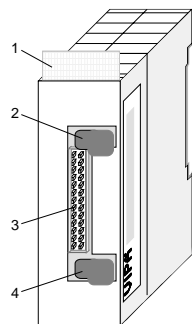
## DO 16xDC 24V 0.5A with UB4x

**Order data** DO 16xDC 24V 0.5A VIPA 222-1BH00

**Description** The digital output module accepts binary control signals from the central bus system and transfers them to the process level via outputs. The module requires 24V via the connector on the front. It has 16 channels and the status of each channel is displayed by means of an LED. This module requires a converter (DEA-UB4x). The module must be connected to the converter module by means of a flattened round cable (DEA-KB91C).

- Properties**
- 16 outputs, isolated from the backplane bus
  - DC 24V supply voltage
  - Output current 0.5A
  - Suitable for magnetic valves and DC contactors
  - LEDs for supply voltage and error message
  - Active channel indication by means of a LED located on converter module UB4x

**Construction**



- [1] Label for module description
- [2] Clip
- [3] Recessed connector for the interface to a conversion module UB4x via the flattened round cable
- [4] Clip

**Status indicator on UB4x**

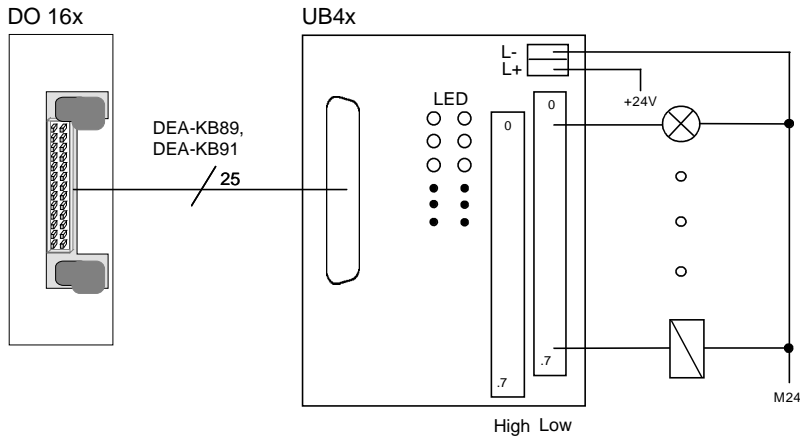
LED	Description
0... .15	LEDs (yellow) A.0 to A.7 High A.0 to A.7 Low when an output is active the respective LED is turned on
L+ L-	LED (green) Supply voltage available

**Pin assignment module**

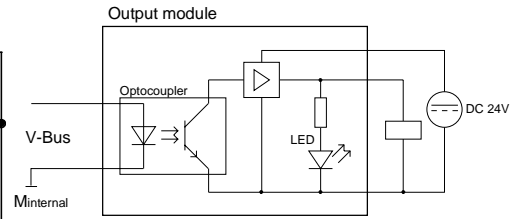
Connector	Pin	Assignment
	23...26	DC 24V supply voltage
	22	Output A.0
	21	Output A.1
	.	.
	.	.
	.	.
	8	Output A.14
	7	Output A.15
	1...6	Supply ground



**Interfacing of UB4x**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 222-1BH00
Number of outputs	16
Nominal load voltage	DC 24V (18 ... 35V) via ext. power supply
No-load current consumption at L+ (all A.x=off)	10mA
Output current per channel	0.5A protected against sustained short circuits
Current consumption via backplane bus	100mA
Voltage supply	5V via backplane bus
Isolation	500Vrms (field voltage - backplane bus)
Status indicator	via LEDs located on the UB4x
Programming specifications	
Input data	-
Output data	2Byte
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	50g

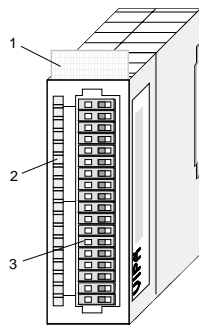
# DO 16xDC 24V 1A

**Order data** DO 16xDC 24V 1A VIPA 222-1BH10

**Description** The digital output module accepts binary control signals from the central bus system and transfers them to the process level via outputs. The module requires 24V via the connector on the front. It has 16 channels and the status of each channel is displayed by means of an LED.

- Properties**
- 16 outputs, isolated from the backplane bus
  - DC 24V supply voltage
  - 1A output current rating
  - Suitable for magnetic valves and DC contactors
  - LEDs for supply voltage and error message
  - Active channel indication by means of an LED

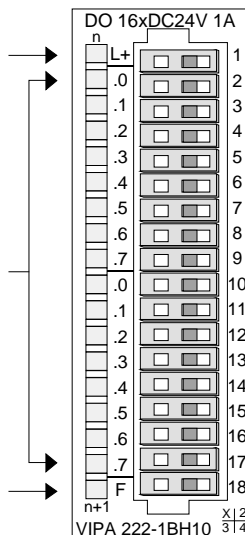
**Construction**



- [1] Label for module description
- [2] LED status indicator
- [3] Edge connector

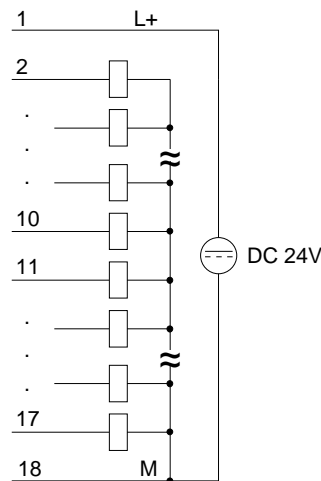
**Status indicator pin assignment**

LED	Description	Pin	Assignment
L+	LED (yellow) Supply voltage available	1	DC 24V supply voltage
A.0 ... A.7	LEDs (green) A.0 to A.7 (per Byte) when an output is active the respective LED is turned on	2	Output A.0
		3	Output A.1
		.	.
		.	.
		.	.
F	LED (red) Overload, overheat or short circuit error	16	Output A.14
		17	Output A.15
		18	Supply ground

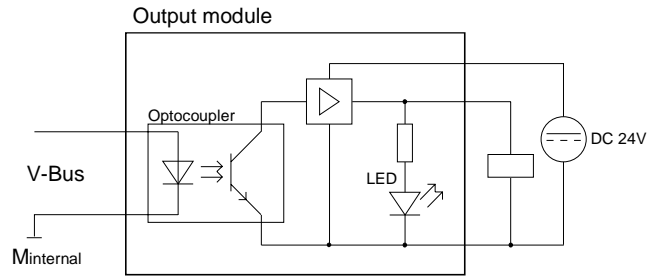


**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 222-1BH10
Number of outputs	16
Nominal load voltage	DC 24V (18 ... 35V) via ext. power supply
No-load current consumption at L+ (all A.x=off)	10mA
Output current per channel	1A protected against sustained short circuits
max. total current	10A
Current consumption via backplane bus	80mA
Voltage supply	5V via backplane bus
Isolation	500Vrms (field voltage - backplane bus)
Status indicator	via LEDs located on the front
Programming specifications	
Input data	-
Output data	2Byte
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	50g

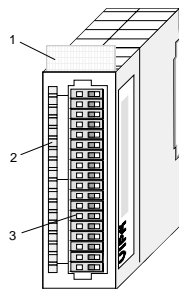
# DO 16xDC 24V 2A

**Order data** DO 16xDC 24V 2A VIPA 222-1BH20

**Description** The digital output module accepts binary control signals from the central bus system and transfers them to the process level via outputs. The module requires 24V via the connector on the front. It has 16 channels and the status of each channel is displayed by means of an LED.

- Properties**
- 16 outputs, isolated from the backplane bus
  - DC 24V supply voltage
  - 2A output current rating
  - Suitable for magnetic valves and DC contactors
  - LEDs for supply voltage and error message
  - Active channel indication by means of an LED

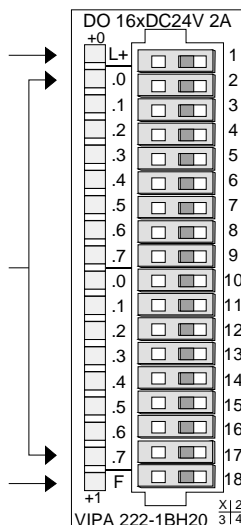
**Construction**



- [1] Label for module description
- [2] LED status indicator
- [3] Edge connector

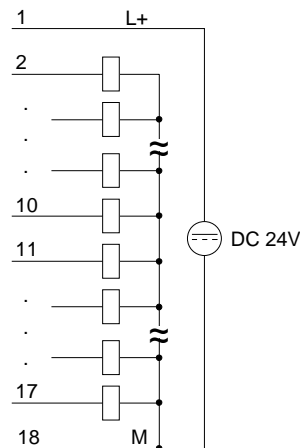
**Status indicator pin assignment**

LED	Description	Pin	Assignment
L+	LED (yellow) Supply voltage available	1	DC 24V supply voltage
A.0 ... A.7	LEDs (green) A.0 to A.7 (per Byte) when an output is active the respective LED is turned on	2	Output A.0
		3	Output A.1
		.	.
		.	.
		.	.
F	LED (red) Overload, overheat or short circuit error	16	Output A.14
		17	Output A.15
		18	Supply ground

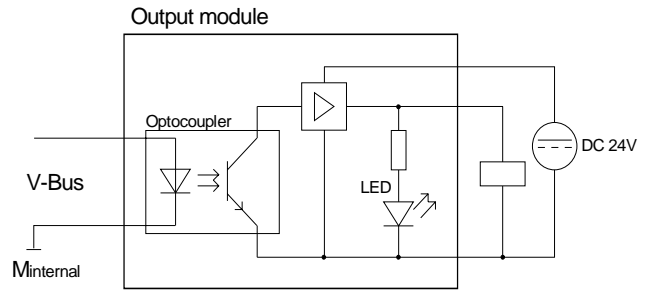


**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 222-1BH20
Number of outputs	16
Nominal load voltage	DC 24V (18 ... 35V) via ext. power supply
No-load current consumption at L+ (all A.x=off)	10mA
Output current per channel	2A protected against sustained short circuits
max. total current	10A
Current consumption via backplane bus	100mA
Voltage supply	5V via backplane bus
Isolation	500Vrms (field voltage - backplane bus)
Status indicator	via LEDs located on the front
Programming specifications	
Input data	-
Output data	2Byte
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	50g

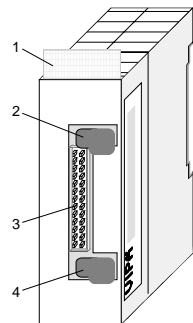
## DO 16xDC 24V 0.5A NPN

**Order data** DO 16xDC 24V 0.5A NPN VIPA 222-1BH50

**Description** The digital output module accepts binary control signals from the central bus system and controls the connected loads at the process level via Misfit outputs. It provides 16 channels that operate as Low-Side switches and that are interconnected via the load voltage. Low-Side switches are suitable for the control of grounds. When a short circuit occurs between the switched line and ground the result is that the load is activated until the short circuit has been removed. Short circuits do not place an additional load on the supply voltage.

- Properties**
- 16 Low-Side outputs
  - Maximum external load voltage DC 32V
  - Output current per channel 0.5A
  - Suitable for small motors, lamps, magnetic valves and contactors

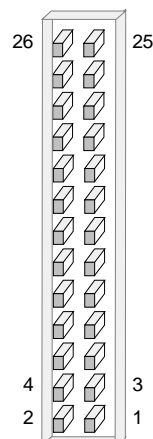
### Construction



- [1] Label for module description
- [2] Clip
- [3] Recessed connector for the interface to a outputconnection
- [4] Clip

### Pin assignment

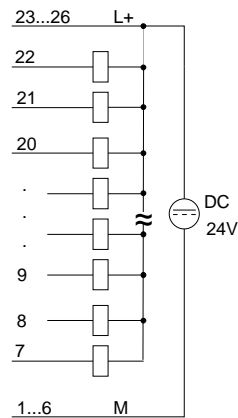
#### Connector



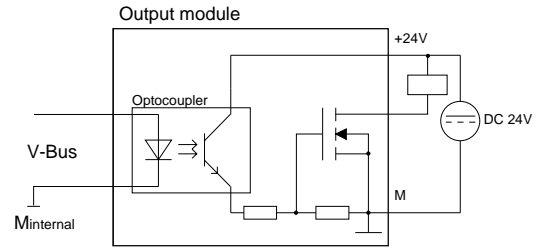
Pin	Assignment
23...26	DC 24V supply voltage
22	Output A.0
21	Output A.1
.	.
.	.
.	.
8	Output A.14
7	Output A.15
1...6	Supply ground

**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Attention!**

This module is not deployable with UB4x from VIPA without technical intervention. For deploying the module with a converter module from VIPA, please call the VIPA Hotline

**Technical data**

Electrical data	VIPA 222-1BH50
Number of outputs	16 via Low-Side
Nominal load voltage	max. DC 24V
max. Output current per channel	0.5A
Current consumption via backplane bus	100mA
Voltage supply	5V via backplane bus
Isolation	500Vrms (field voltage - backplane bus)
Switching rate	20kHz max.
Status indicator	-
Programming specifications	
Input data	-
Output data	2Byte
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	80g

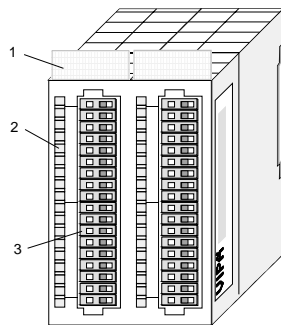
# DO 32xDC 24V 1A

**Order data** DO 32xDC 24V 1A VIPA 222-2BL10

**Description** The digital output module accepts binary control signals from the central bus system and transfers them to the process level via outputs. The module requires 24V via the connector on the front. It provides 32 channels and the status of each channel is displayed by means of LEDs.

- Properties**
- 32 outputs, isolated from the backplane bus
  - DC 24V supply voltage
  - Output current per channel 1A
  - Suitable for magnetic valves and DC contactors
  - LEDs for supply voltage and error message
  - Active channel indication by means of an LED

**Construction**



- [1] Label for module description
- [2] LED status indicator
- [3] Edge connector

**Status indicator pin assignment**

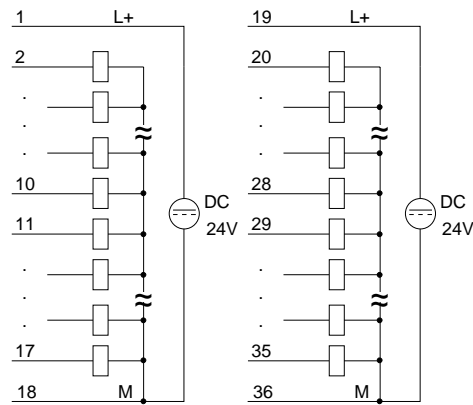
LED	Description	Pin	Assignment
L+	LED (yellow) Supply voltage available	1	DC 24V supply voltage
.0 ... .7	LEDs (green) A.0 to A.7 (per Byte) when an output is active the respective LED is turned on	2	Output A.0
		3	Output A.1
		...	...
		17	Output A.15
		18	Supply ground
		19	DC 24V supply voltage
		20	Output A.16
		...	...
F	LED (red) Overload, overheat or short circuit error	34	Output A.30
		35	Output A.31
		36	Supply ground



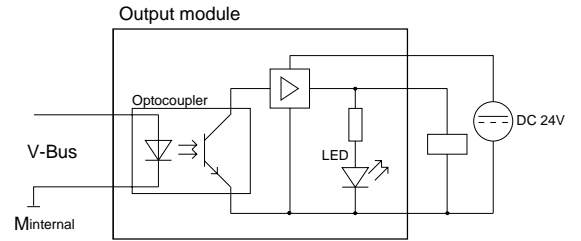


**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 222-2BL10
Number of outputs	32
Nominal load voltage	DC 24V (18 ... 35V) from ext. power supply
No-load current consumption at L+ (all A.x=off)	15mA
max. Output current per channel	1A protected against sustained short circuits
max. Contact load	10A
Current consumption via backplane bus	140mA
Voltage supply	5V via backplane bus in 2 groups of 16 outputs each
Isolation	500Vrms (field voltage - backplane bus)
Status indicator	via LEDs located on the front
Programming specifications	
Input data	-
Output data	4Byte
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	50.8x76x76
Weight	50g

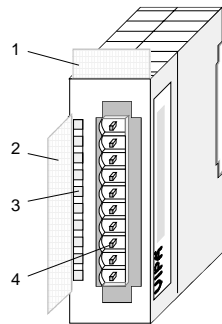
## DO 8xRelay COM

**Order data** DO 8xRelay COM VIPA 222-1HF00

**Description** The digital output module accepts binary control signals from the central bus system and controls the connected loads at the process level via relay outputs. The module derives power from the backplane bus. The load voltage must be connected to terminal 1. When the total current exceeds 8A you have to balance the load current between terminals 1 and 10. The module has 8 channels and the status of each channel is displayed by means of an LED.

- Properties**
- 8 relay outputs
  - Power supply via backplane bus
  - External load voltage AC 230V / DC 30V
  - Output current per channel 5A (AC 230V / DC 30V)
  - Suitable for motors, lamps, magnetic valves and DC contactors
  - Active channel indication by means of LED

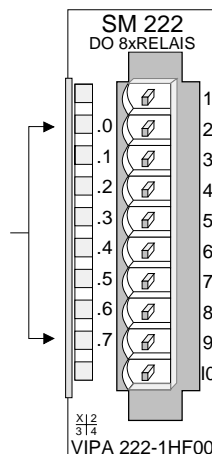
**Construction**



- [1] Label for module description
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

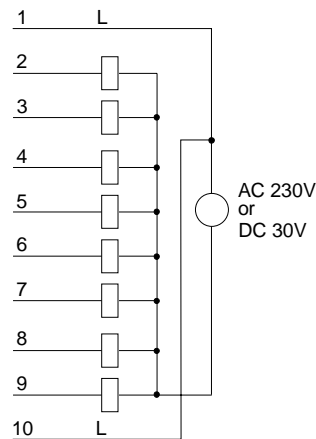
**Status indicator pin assignment**

LED	Description	Pin	Assignment
.0... .7	LEDs (green) A.0 to A.7 when an output is active the respective LED is turned on	1	Supply voltage L
		2	Relay output. A.0
		3	Relay output. A.1
		4	Relay output. A.2
		5	Relay output. A.3
		6	Relay output. A.4
		7	Relay output. A.5
		8	Relay output. A.6
		9	Relay output. A.7
		10	Supply voltage L

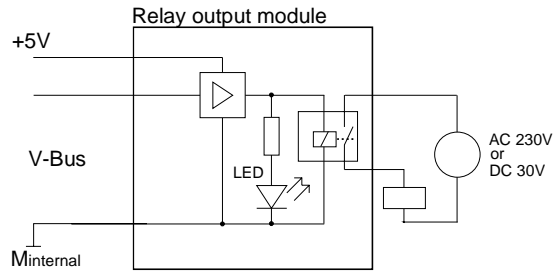


**Wiring and schematic diagram**

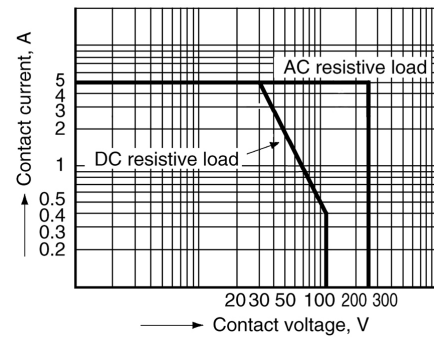
**Wiring diagram**



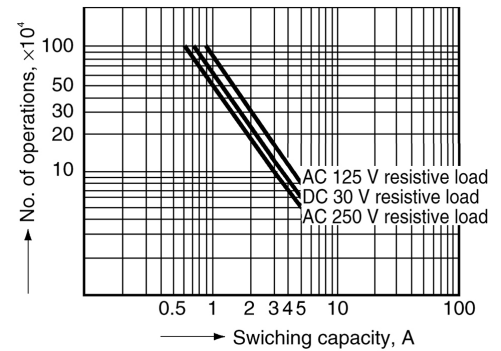
**Schematic diagram**



**Maximum load**



**Service life**



**Technical data**

Electrical data	VIPA 222-1HF00
Number of outputs	8 via relay
Nominal load voltage	max. AC 230V or DC 30V
No-load current consumption at L+ (all A.x=off)	-
Total current	with 1 L: max. 8A with 2 L: max. 16A
max. output current per channel	AC 230V: 5A / DC 30V: 5A
Current consumption v. backp. bus	270mA
Voltage supply	5V via backplane bus
Isolation	500Vrms (field voltage-backpl. bus)
Switching rate	max. 100Hz
Status indicator	via LEDs located on the front
Programming specifications	
Input data	-
Output data	1Byte
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	80g

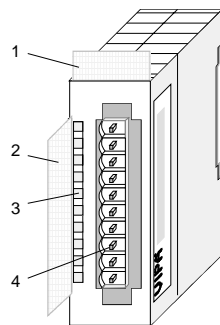
# DO 4xRelay

**Order data** DO 4xRelay VIPA 222-1HD10

**Description** The digital output module accepts binary control signals from the central bus system and controls the connected loads at the process level via relay outputs. The module derives power from the backplane bus. The module has 4 isolated channels that operate as switches and the status of each channel is displayed by means of a LED. Power required by active loads must be supplied externally.

- Properties**
- 4 galvanically isolated relay outputs
  - Power supply via backplane bus
  - External load voltage AC 230V / DC 30V (may be mixed)
  - Max. output current per channel 5A (AC 230V / DC 30V )
  - Suitable for motors, lamps, magnetic valves and DC contactors
  - Active channel indication by means of an LED

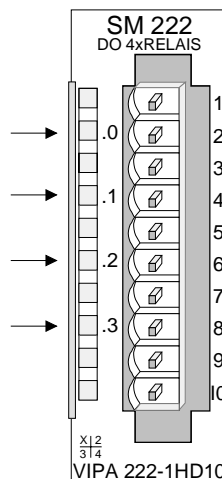
**Construction**



- [1] Label for module description
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

**Status indicator pin assignment**

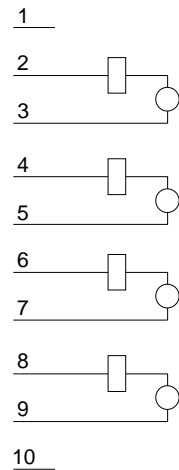
LED	Description
.0... .3	LEDs (green) A.0 to A.3 when an output is active the respective LED is turned on



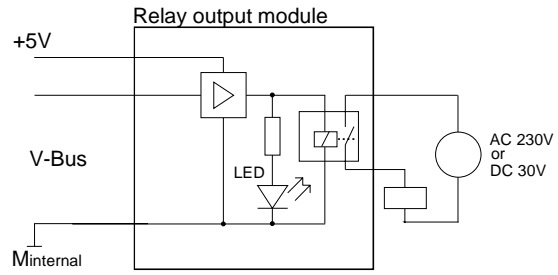
Pin	Assignment
1	not connected
2+3	Relay output A.0
4+5	Relay output A.1
6+7	Relay output A.2
8+9	Relay output A.3
10	not connected

**Wiring and schematic diagram**

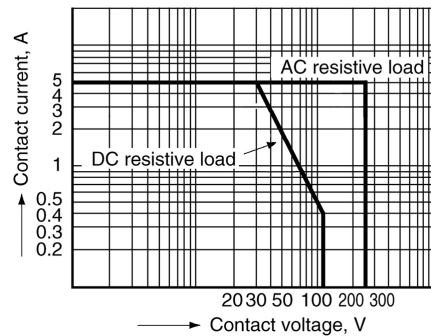
**Wiring diagram**



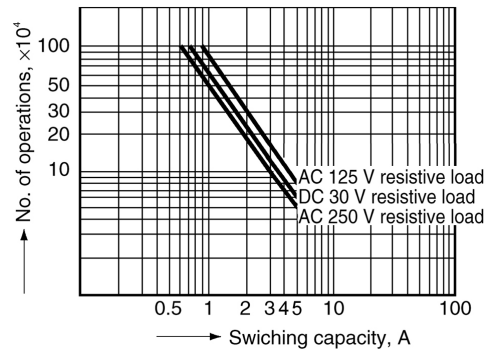
**Schematic diagram**



**Maximum load**



**Service life**



**Technical data**

Electrical data	VIPA 222-1HD10
Number of outputs	4 via relay
Nominal load voltage	AC 230V or max. DC 30V
max. Output current	AC 230V: 5A / DC 30V: 5A
Current consumption via backplane bus	150mA
Voltage supply	5V via backplane bus
Isolation	500Vrms (field voltage - backplane bus)
Switching rate	max. 100Hz
Status indicator	via LEDs located on the front
Programming specifications	
Input data	-
Output data	1Byte (Bit 0 ... Bit 3)
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	80g

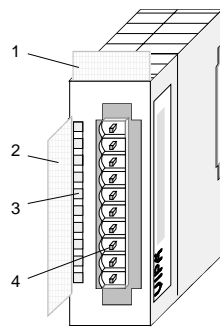
## DO 4xRelay bistable

**Order data** DO 4xRelay bistable VIPA 222-1HD20

**Description** The digital output module accepts binary control signals from the central bus system and controls the connected loads at the process level via bistable relay outputs. The module derives power from the backplane bus. The module has 4 channels that operate as switches. The status of the respective switch is retained if the power from the controlling system fails.

- Properties**
- 4 galvanically isolated relay outputs
  - Power supply via backplane bus
  - External load voltage AC 230V / DC 30V (may be mixed)
  - Max. Output current per channel 16A (AC 230V / DC 30V)
  - Suitable for motors, lamps, magnetic valves and DC contactors

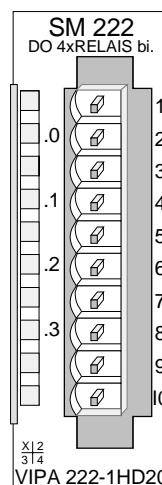
**Construction**



- [1] Label for module description
- [2] Label for the bit address with description
- [3] LEDs (not used)
- [4] Edge connector

**Output byte /  
Pin assignment**

Bit	Description
Bit 0	set A.0
Bit 1	set A.1
Bit 2	set A.2
Bit 3	set A.3
Bit 4	reset A.0
Bit 5	reset A.1
Bit 6	reset A.2
Bit 7	reset A.3

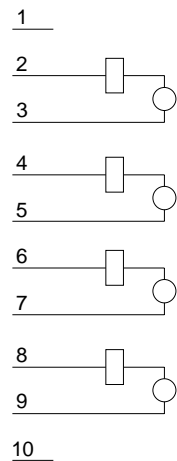


Pin	Assignment
1	not connected
2+3	Relay output. A.0
4+5	Relay output. A.1
6+7	Relay output. A.2
8+9	Relay output. A.3
10	not connected

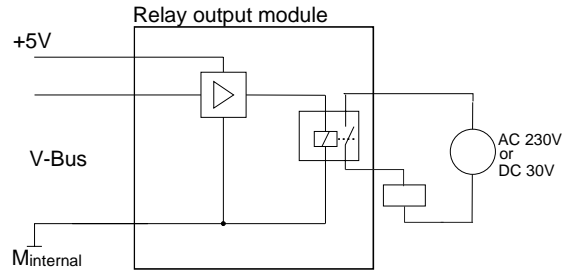
Setting the Bits 0...3 activates the concerning channel.  
Setting Bits 4..7 causes a reset of the concerning channel after min. 50ms.

**Wiring and schematic diagram**

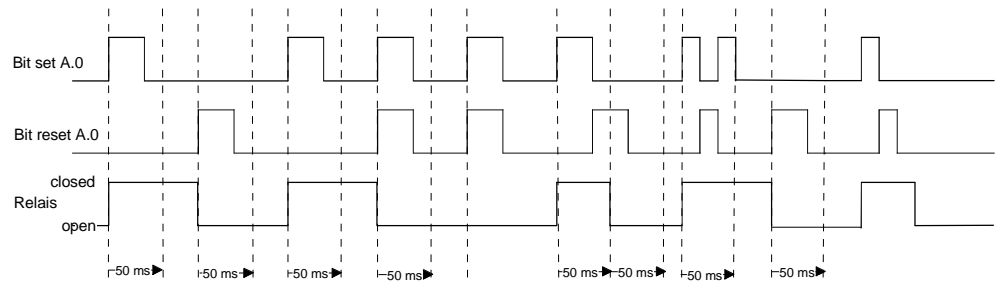
**Wiring diagram**



**Schematic diagram**



**Signaling diagram**



**Note!**

Please remember that a relay output that has been set may only be reset after at least 50ms when the set-signal has been removed.

**Technical data**

Electrical data	VIPA 222-1HD20
Number of outputs	4 via relay
Nominal load voltage	AC 230V or DC 30V
max. Output current per channel	AC 230V: 16A / DC 30V: 16A
Current consumption via backplane bus	40mA
Voltage supply	5V via backplane bus
Isolation	500Vrms (field voltage-backpl. bus)
Switching rate	max. 100Hz
Status indicator	-
Programming specifications	
Input data	-
Output data	1Byte
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	80g

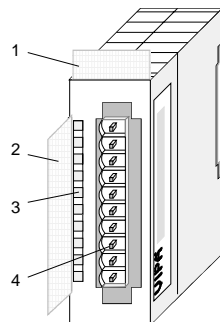
## DO 8xSolid State COM

**Order data** DO 8xSolid State COM VIPA 222-1FF00

**Description** The solid-state output module accepts binary control signals from the central bus system and controls the connected loads at the process level via solid-state relay outputs. The module derives power from the backplane bus. The module has 8 channels that are interconnected via the load voltage that act as switches and display the status by means of LEDs. Solid-state relays change state when the load voltage passes through zero (AC).

- Properties**
- 8 solid-state outputs with active channel indication by means of a LED
  - Extended service life due to the fact that the load voltage (provided this is AC) is switched when it passes through zero
  - External load voltage AC 230V or DC 400V
  - Max. output current per channel 0.5A (AC 230V / DC 400V)
  - Suitable for small motors, lamps, magnetic valves and contactors

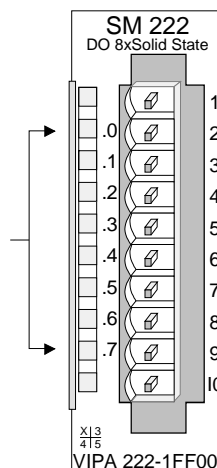
**Construction**



- [1] Label for module description
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

**Status indicator pin assignment**

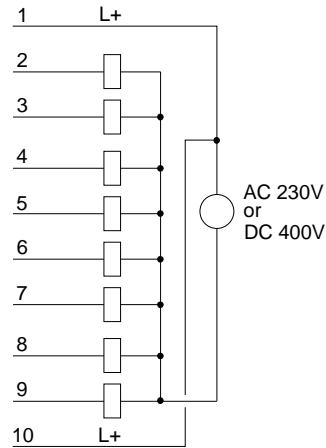
LED	Description	Pin	Assignment
.0 ... .7	LEDs (green) A.0 to A.7 when an output is active the respective LED is turned on	1	Supply voltage
		2	Output A.0
		3	Output A.1
		4	Output A.2
		5	Output A.3
		6	Output A.4
		7	Output A.5
		8	Output A.6
		9	Output A.7
		10	Supply voltage



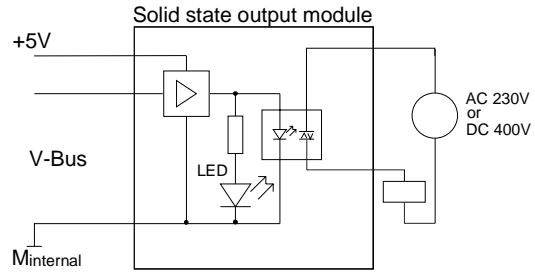


**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 222-1FF00
Number of outputs	8 via solid-state
Nominal load voltage	AC 230V or DC 400V
max. Output current per channel	AC 230V: 0.5A / DC 400V: 0.5A
Contact resistance	typ. 2.1Ω , max. 3.2Ω
Current consumption via backplane bus	140mA
Voltage supply	5V via backplane bus
Isolation	500Vrms (field voltage - backplane bus)
Switching rate	max. 100Hz
Status indicator	via LEDs located on the front
Programming specifications	
Input data	-
Output data	1Byte (Bit 0 ... Bit 7)
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	80g

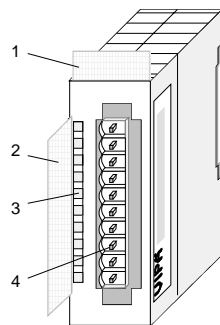
## DO 4xSolid State

**Order data** DO 4xSolid State VIPA 222-1FD10

**Description** The solid-state output module accepts binary control signals from the central bus system and controls the connected loads at the process level via solid-state relay outputs. The module derives power from the backplane bus. The module has 4 separate channels that operate as switches and display the status by means of LEDs. Active loads must be supplied with external power.

- Properties**
- 4 galvanically isolated solid-state outputs
  - Power supply via backplane bus
  - External load voltage AC 230V or DC 400V
  - Max. output current per channel 0.5A (AC 230V / DC 400V )
  - Suitable for motors, lamps, magnetic valves and contactors
  - Active channel indication by means of an LED

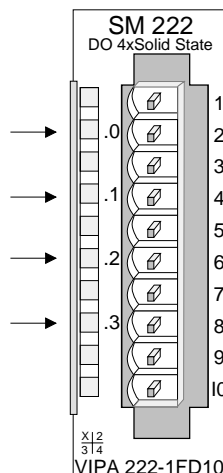
**Construction**



- [1] Label for module description
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

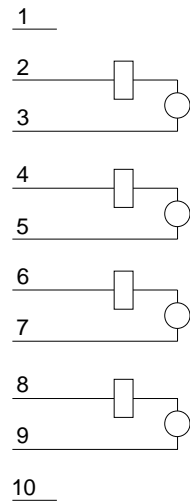
**Status indicator pin assignment**

LED	Description	Pin	Assignment
.0... .3	LEDs (green) A.0 to A.3 when an output is active the respective LED is turned on	1	not connected
		2+3	Output A.0
		4+5	Output A.1
		6+7	Output A.2
		8+9	Output A.3
		10	not connected

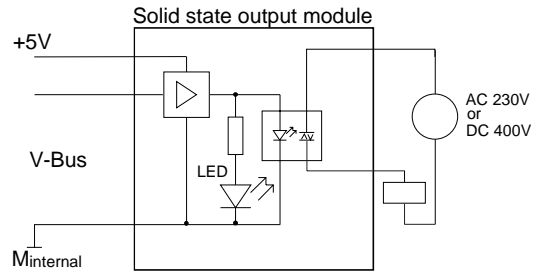


**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 222-1FD10
Number of outputs	4 via solid state
Nominal load voltage	AC 230V or DC 400V
max. output current per channel	AC 230V: 0.5A / DC 400V: 0.5A
Current consumption via backplane bus	100mA
Voltage supply	5V via backplane bus
Isolation	500Vrms (field voltage - backplane bus)
Switching rate	max. 100Hz
Status indicator	via LEDs located on the front
Programming specifications	
Input data	-
Output data	1Byte (Bit 0 ... Bit 3)
Parameter data	-
Diagnostic data	-
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	80g



## Chapter 15 Digital input/output modules

### Overview

This chapter contains a description of the construction and the operation of the VIPA digital input/output modules.

Below follows a description of:

- A system overview of the digital input/output modules
- Properties
- Construction
- Interfacing and schematic diagram
- Technical data

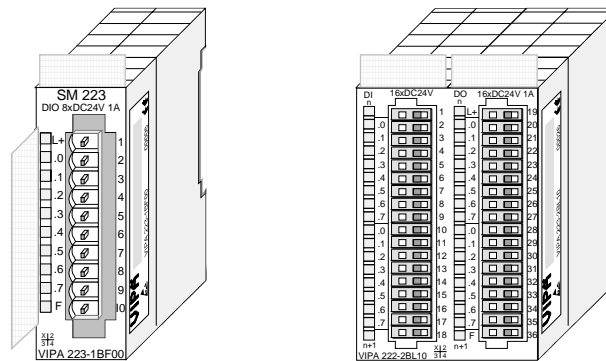
### Contents

Topic	Page
<b>Chapter 15 Digital input/output modules .....</b>	<b>15-1</b>
System overview .....	15-2
Security hints for DIO modules .....	15-2
DIO 8xDC 24V 1A.....	15-3
DI 16xDC 24V, DO 16xDC 24V 1A .....	15-5

## System overview

### Input/output modules SM 223

Here follows a summary of the digital input/output modules that are currently available from VIPA:



### Order data input/output modules

Type	Order number	Page
DIO 8xDC 24V 1A	VIPA 223-1BF00	15-3
DI 16xDC 24V, DO 16xDC 24V 1A	VIPA 223-2BL10	15-5

## Security hints for DIO modules



### Attention!

Please regard that the voltage applied to an output channel must be  $\leq$  the voltage supply applied to L+.

Due to the parallel connection of in- and output channel per group, a set output channel may be supplied via an applied input signal.

Thus, a set output remains active even at power-off of the voltage supply with the applied input signal.

Non-observance may cause module demolition.

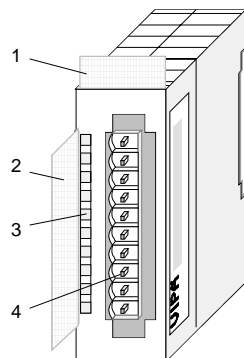
# DIO 8xDC 24V 1A

**Order data**                      DIO 8xDC 24V 1A                      VIPA 223-1BF00

**Description**                      This module is a combination module. It has 8 channels that may be used as input or as output channel. The status of the channels is displayed by means of LEDs. Every channel is provided with a diagnostic function, i.e. when an output is active the respective input is set to "1". When a short circuit occurs at the load, the input is held at "0" and the error is detectable by analyzing the input.

- Properties**
- 8 channels, isolated from the backplane bus (as input or output)
  - Diagnostic function
  - Nominal input voltage DC 24V / supply voltage DC 24V
  - Output current 1A
  - LED error display for overload, overheat or short circuit
  - Active channels displayed by means of LED

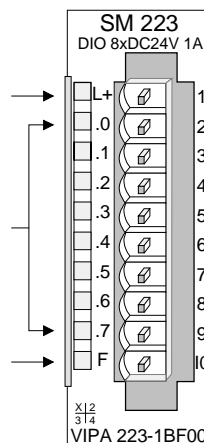
**Construction**



- [1] Label for the module description
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

**Status indicator pin assignment**

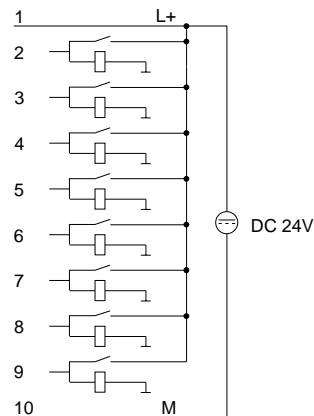
LED	Description
L+	LED (yellow) Supply voltage available
.0 ... .7	LEDs (green) when the input signal is "1" or the output is active the respective LED is turned on
F	LED (red) Overload, overheat or short circuit error



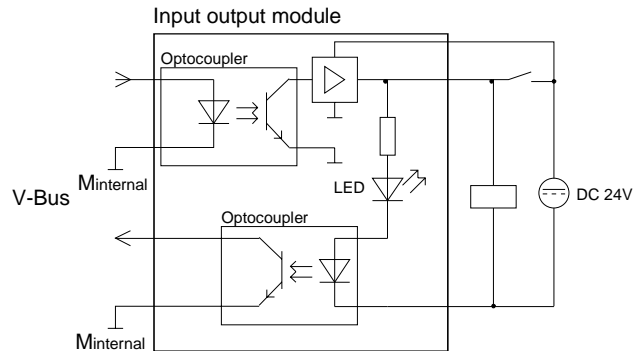
Pin	Assignment
1	+DC 24V supply voltage
2	Channel K.0
3	Channel K.1
4	Channel K.2
5	Channel K.3
6	Channel K.4
7	Channel K.5
8	Channel K.6
9	Channel K.7
10	Supply ground

**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 223-1BF00
Number of channels	8
Rated load voltage	DC 24V (18...35V) via ext. power supply
No-load current consumption at L+ (all A.x=off)	50mA
Output current per channel	1A protected against short circuits
Total output current	12A
Nominal input voltage	DC 24V (18 ... 28,8V)
Signal voltage "0"	0 ... 5V
Signal voltage "1"	15 ... 28,8V
Input filter time delay	3ms
Input current	typ. 7mA
Voltage supply	5V via backplane bus
Current consumption via backplane bus	65mA
Data width in the process image	1Byte PAA, 1Byte PAE
Status indicator	via LEDs located on the front
<b>Programming specifications</b>	
Input data	1Byte
Output data	1Byte
Parameter data	-
Diagnostic data	-
<b>Dimensions and weight</b>	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	50g



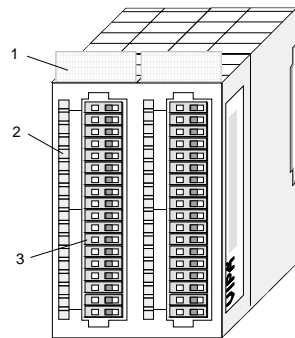
# DI 16xDC 24V, DO 16xDC 24V 1A

**Order data** DI 16xDC 24V, DO 16xDC 24V 1A VIPA 223-1BL10

**Description** The module has 32 channels that are isolated from the backplane bus. 16 channels operate as inputs and 16 as outputs. The status of the channels is displayed by means of LEDs.

- Properties**
- 32 channels, of these 16 input and 16 output channels
  - Nominal input voltage DC 24V
  - Supply voltage DC 24V(external) for outputs
  - Output current 1A per channel
  - LED error display for overload, overheat or short circuit
  - Active channels displayed by means of an LED

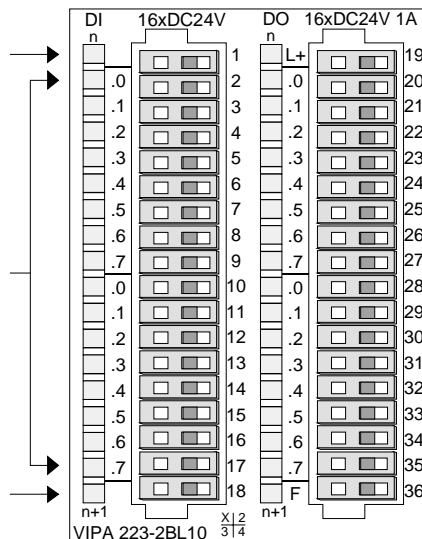
**Construction**



- [1] Label for the module description
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

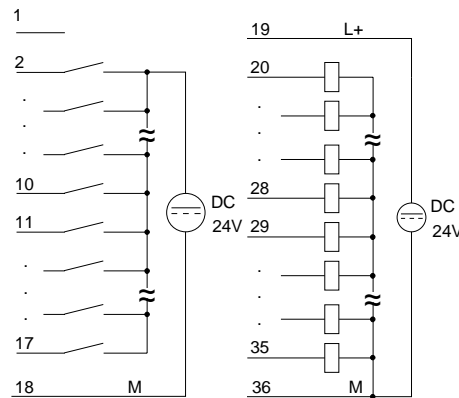
**Status indicator pin assignment**

LED	Description	Pin	Assignment
L+	LED (yellow) Supply voltage available	1	not connected
.0 ... .7	LED (green) E.0 ... E.7 (per Byte) A.0 ... A.7 (per Byte) when the signal (input) is "1" or the output is active, the respective LED is turned on	2	Input E.0
F	LED (red) Overload, overheat or short circuit error	17	Input E.15
		18	Ground for inputs
		19	Supply voltage +24V
		20	Output A.0
		35	Output A.15
		36	Supply voltage ground outputs

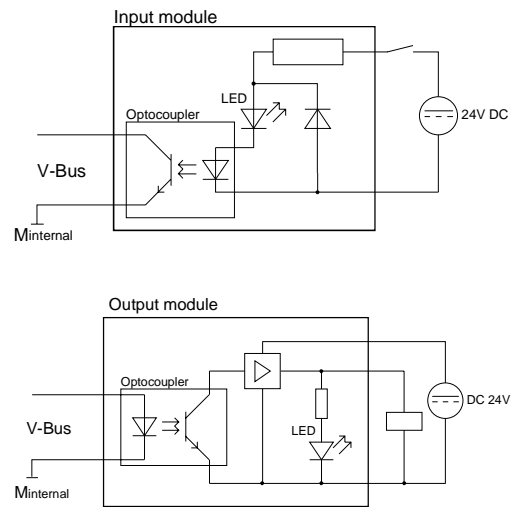


**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Technical data**

Electrical data	VIPA 223-2BL10
Number of channels	32
Rated load voltage	DC 24V (18...35V) via ext. power source
No-load current consumption at L+ (all A.x=off)	10mA
Output current per channel	1A protected against short circuits
max. contact load per connector	10A
Nominal input voltage	DC 24V (18 ... 28,8V)
Signal voltage "0"	0 ... 5V
Signal voltage "1"	15 ... 28,8V
Input filter time delay	3ms
Input current	typ. 7mA
Voltage supply	5V via backplane bus
Current consumption via backplane bus	100mA
Data width in the process image	2Byte PAA, 2Byte PAE
Status indicator	via LEDs located on the front
<b>Programming specifications</b>	
Input data	2Byte
Output data	2Byte
Parameter data	-
Diagnostic data	-
<b>Dimensions and weight</b>	
Dimensions (WxHxD) in mm	50.8x76x76
Weight	100g

## Chapter 16 Analog input modules

### Overview

This chapter contains a description of the construction and the operation of the VIPA analog input modules.

Below follows a description of:

- A system overview of the analog input modules
- Properties
- Constructions
- Interfacing and schematic diagram
- Technical data

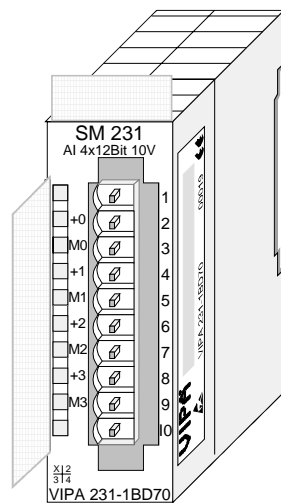
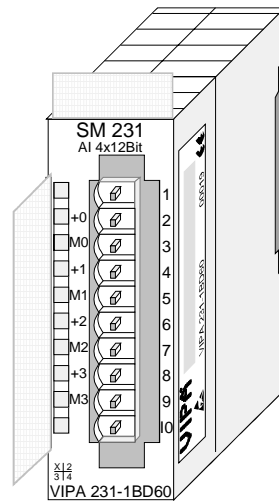
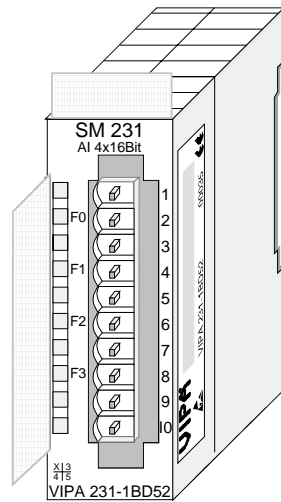
### Contents

Topic	Page
<b>Chapter 16 Analog input modules .....</b>	<b>16-1</b>
System overview .....	16-2
General .....	16-4
AI 4x16Bit, multiinput .....	16-5
AI 4x12Bit, 4 ... 20mA, isolated .....	16-16
AI 4x12Bit, $\pm 10V$ , isolated .....	16-19
AI 4x16Bit f .....	16-22
AI 8x16Bit .....	16-32

## System overview

### Input modules SM 231

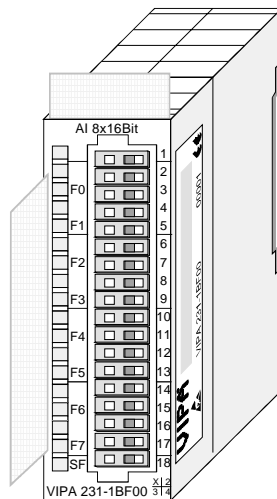
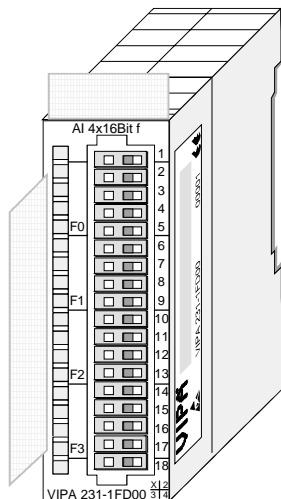
Here follows a summary of the analog input modules that are currently available from VIPA:



### Order data input modules

Type	Order number	Page
AI4x16Bit, Multiinput	VIPA 231-1BD52	16-5
AI4x12Bit, 4 ... 20mA, isolated	VIPA 231-1BD60	16-16
AI4x12Bit, $\pm 10V$ , isolated	VIPA 231-1BD70	16-19

**Input modules  
SM 231**



**Order data  
input modules**

Type	Order number	Page
A14x16Bit f	VIPA 231-1FD00	16-22
A18x16Bit	VIPA 231-1BF00	16-32

## General

### Cabling for analog signals

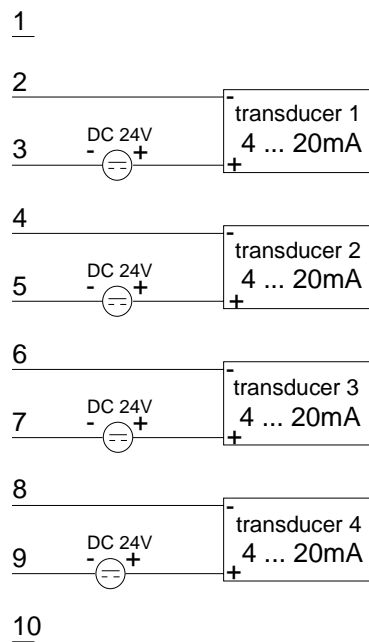
You should only use screened twisted-pair cable when you are connecting analog signals. These cables reduce the effect of electrical interference. The screen of the analog signal cable should be grounded at both ends. When there are potential differences between the cable ends, there may flow a current will to equalize the potential difference. This current could interfere with the analog signals. Under these circumstances it is advisable to ground the screen of the signal cable at one end only.

### Connecting test probes

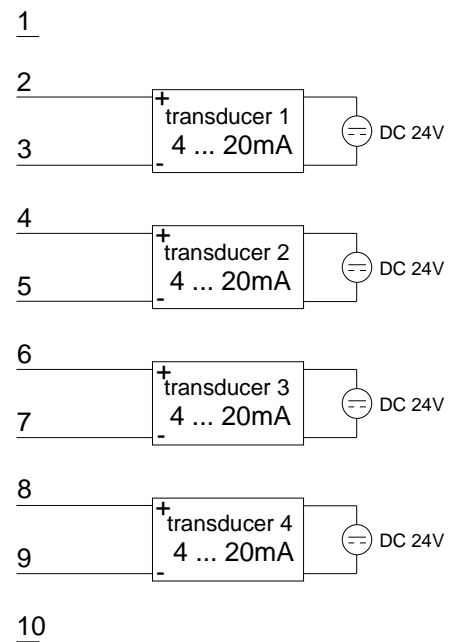
Our analog input modules provide a large number of input configurations for 2- and 4wire transducers.

Please remember that transducers require an external power source. You have to connect an external power supply in line with any 2wire transducer. The following diagram explains the connection of 2- and 4wire transducers:

#### 2wire interfacing



#### 4wire interfacing



#### Note!

Please ensure that you connect transducers with the correct polarity! Unused inputs should be short circuited by placing a link between the positive pole and the common ground for the channel.

### Parameterization and diagnosis during runtime

By using the SFCs 55, 56 and 57 you may change the parameters of the analog modules during runtime via the CPU 21x.

For diagnosis evaluation during runtime, you may use the SFCs 51 and 59. They allow you to request detailed diagnosis information and to react to it.

# AI 4x16Bit, multiinput

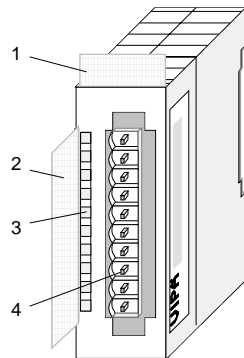
**Order data** AI 4x16Bit multiinput VIPA 231-1BD52

**Description** The module has 4 inputs that you may configure individually. The module requires a total of 8 input data bytes in the process image (2Byte per channel).

Isolation between the channels on the module and the backplane bus is provided by means of DC/DC converters and optocouplers.

- Properties**
- the different channels are individually configurable and may be turned off
  - the common signal inputs of the channels are not isolated from each other and the permitted potential difference is up to 5V
  - LED for cable break and over current in sensor circuits
  - diagnostic function

**Construction**



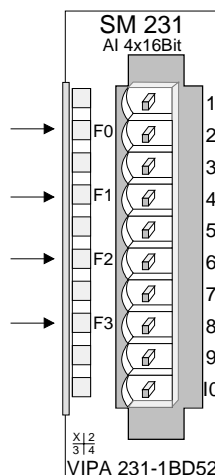
- [1] Label for module description
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

**Status indicators pin assignment**

**LED Description**

F0 ... F3 LED (red):  
 turned on when an open circuit exists on the 4...20mA sensor circuits  
  
 blinks when the current > 40mA current sensor circuits

**Pin Assignment**



- 1 For four-wire systems channel 0
- 2 + channel 0
- 3 Channel 0 common
- 4 + channel 1
- 5 Channel 1 common
- 6 + channel 2
- 7 Channel 2 common
- 8 + channel 3
- 9 Channel 3 common
- 10 For 4wire systems channel 2

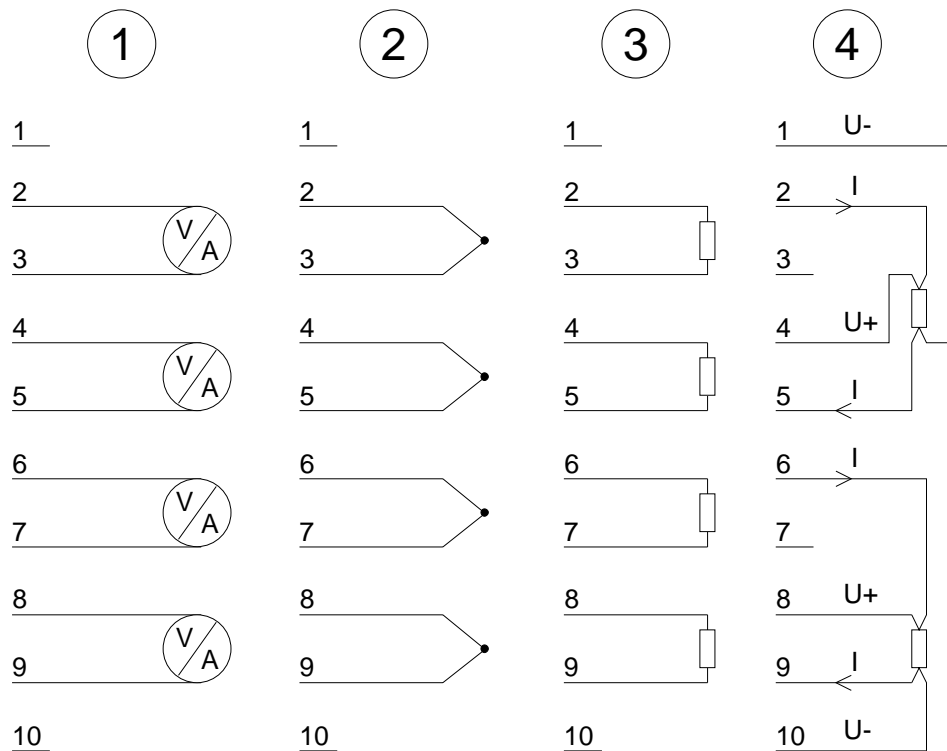
**Wiring diagrams**

The following illustration shows the connection options for the different measuring ranges. The assignment to the measuring ranges is to find in the column "Conn." of the table "Function no. assignment" on the next pages.



**Note!**

Please note that the module 231-1BD52 was developed from the VIPA 231-1BD50. The measuring function no longer starts at 00h but it is offset by one to 01h. The measurement function no. 00h does not affect permanently stored configuration data.



**Attention!**

Temporarily not used inputs have to be connected with the concerning ground at activated channel. When deactivating unused channels by means of FFh, this is not required.



**Note!**

Every channel is individual parameterizable. For the parameterization 10Byte parameterization data are available. They are stored permanently and remain in memory during power-off.



**Function no. assignment**

The assignment of a function no. to a certain channel happens during parameterization. The function no. 00h does not influence the function no. stored in the permanent parameterization data.

Assigning FFh deactivates the according channel.

No.	Function	Measurement range / representation	Tolerance	Conn.
00h	Does not affect permanently stored configuration data			
01h	Pt100 in two-wire mode	-200 .. +850°C / in units of 1/10°C, two's complement	<sup>1)2)3)</sup> ±1°C	(3)
02h	Pt1000 in two-wire mode	-200 .. +500°C / in units of 1/10°C, two's complement	<sup>1)2)3)</sup> ±1°C	(3)
03h	NI100 in two-wire mode	-50 .. +250°C / in units of 1/10°C, two's complement	<sup>1)2)3)</sup> ±1°C	(3)
04h	NI1000 in two-wire mode	-50 .. +250°C / in units of 1/10°C, two's complement	<sup>1)2)3)</sup> ±1°C	(3)
05h	Resistance measurement 60Ohm two-wire	- / 60Ω= final value (32767)	<sup>1)2)3)</sup> ±0.2% of final value	(3)
06h	Resistance measurement 600Ohm two-wire	- / 600Ω = final value (32767)	<sup>1)2)3)</sup> ±0.1% of final value	(3)
07h	Resistance measurement 3000Ohm two-wire	- / 3000Ω = final value (32767)	<sup>1)2)3)</sup> ±0.1% of final value	(3)
08h	Resistance measurement 6000Ohm two-wire	- / 6000Ω = final value (32767)	<sup>1)2)3)</sup> ±0.1% of final value	(3)
09h	Pt100 via four-wire connection	-200 .. +850°C / in units of 1/10°C, two's complement	<sup>1)2)</sup> ±0.5°C	(4)
0Ah	Pt1000 via four-wire connection	-200 .. +500°C / in units of 1/10°C, two's complement	<sup>1)2)</sup> ±0.5°C	(4)
0Bh	NI100 via four-wire connection	-50 .. +250°C / in units of 1/10°C, two's complement	<sup>1)2)</sup> ±0.5°C	(4)
0Ch	NI1000 via four-wire connection	-50 .. +250°C / in units of 1/10°C, two's complement	<sup>1)2)</sup> ±0.5°C	(4)
0Dh	Resistance measurement 60Ohm four-wire	- / 60Ω= final value (32767)	<sup>1)2)</sup> ±0.1% of final value	(4)
0Eh	Resistance measurement 600Ohm four-wire	- / 600Ω= final value (32767)	<sup>1)2)</sup> ±0.05% of final value	(4)
0Fh	Resistance measurement 3000Ohm four-wire	- / 3000Ω = final value (32767)	<sup>1)2)</sup> ±0.05% of final value	(4)
10h	Thermo element type J , externally compensated	-210 °C .. 850 °C / in units of 1/10°C, two's complement	<sup>1)2)4)</sup> ±1°C	(2)

*continue ...*

continue ...

11h	Thermo element type K, externally compensated	-270 °C .. 1200 °C / in units of 1/10°C, two's complement	<sup>1)</sup> <sup>2)</sup> <sup>4)</sup> ±1.5°C	(2)
12h	Thermo element type N, externally compensated	-200 °C .. 1300 °C / in units of 1/10°C, two's complement	<sup>1)</sup> <sup>2)</sup> <sup>4)</sup> ±1.5°C	(2)
13h	Thermo element type R, externally compensated	-50 °C .. 1760 °C / in units of 1/10°C, two's complement	<sup>1)</sup> <sup>2)</sup> <sup>4)</sup> ±4°C	(2)
14h	Thermo element type T, externally compensated	-270 °C .. 400 °C / in units of 1/10°C, two's complement	<sup>1)</sup> <sup>2)</sup> <sup>4)</sup> ±1.5°C	(2)
15h	Thermo element type S, externally compensated	-50 °C .. 1760 °C / in units of 1/10°C, two's complement	<sup>1)</sup> <sup>2)</sup> <sup>4)</sup> ±5°C	(2)
18h	Thermo element type J, internally compensated	-210 °C .. 850 °C / in units of 1/10°C, two's complement	<sup>1)</sup> <sup>2)</sup> <sup>5)</sup> ±1.5°C	(2)
19h	Thermo element type K, internally compensated	-270 °C .. 1200 °C / in units of 1/10°C, two's complement	<sup>1)</sup> <sup>2)</sup> <sup>5)</sup> ±2°C	(2)
1Ah	Thermo element type N, internally compensated	-200 °C .. 1300 °C / in units of 1/10°C, two's complement	<sup>1)</sup> <sup>2)</sup> <sup>5)</sup> ±2°C	(2)
1Bh	Thermo element type R, internally compensated	-50 °C .. 1760 °C / in units of 1/10°C, two's complement	<sup>1)</sup> <sup>2)</sup> <sup>5)</sup> ±5°C	(2)
1Ch	Thermo element type T, internally compensated	-270 °C .. 400 °C / in units of 1/10°C, two's complement	<sup>1)</sup> <sup>2)</sup> <sup>5)</sup> ±2°C	(2)
1Dh	Thermo element type S, internally compensated	-50 °C .. 1760 °C / in units of 1/10°C, two's complement	<sup>1)</sup> <sup>2)</sup> <sup>4)</sup> ±5°C	(2)
27h	Voltage 0...50mV Siemens S7-format	0...50mV / 59.25mV = max. range before over range (32767) 0...50mV = nominal value (0...27648) two's complement	<sup>1)</sup> ±0.1% of final value	(1)
28h	Voltage ±10V Siemens S7-format	±11.85V / 11.85V= max. value before over range (32767) -10...10V= nominal range (-27648...27648) -11.85V= min. value before under range (-32767) two's complement	<sup>1)</sup> ±0.05% of final value	(1)
29h	Voltage ±4V Siemens S7-format	±4.74V / 4.74V = max. value before over range (32767) -4...4V = rated range (-27648...27648) -4.74V = min. value before under range (-32767) two's complement	<sup>1)</sup> ±0.05% of final value	(1)
2Ah	Voltage ±400mV Siemens S7-format	±0.474V / 474mV = max. value before over range (32767) -400...400mV = rated range (-27648...27648) -474mV = min. value before under range (-32767) two's complement	<sup>1)</sup> ±0.1% of final value	(1)
2Bh	Voltage ±10V Siemens S5-format	±11.85V / 12.5V = max. value before over range (20480) -10...10V = rated range (-16384...16384) -12.5V = min. value before under range (-20480) value and sign	<sup>1)</sup> ±0.2% of final value	(1)

continue ...

continue ...

2Ch	Current $\pm 20\text{mA}$ Siemens S7-format	$\pm 23.70\text{mA}$ / 23.70mA = max. value before over range (32767) -20...20mA = rated value (-27648...27648) -23.70mA = min. value before under range (-32767) two's complement	<sup>1)</sup> $\pm 0.05\%$ of final value	(1)
2Dh	Current 4...20mA Siemens S7-format	1.185 .. +22.96mA / 22.96mA = max. value before over range (32767) 4...20mA = rated range (0...27648) 0mA = min. value before under range (-5530) two's complement	<sup>1)</sup> $\pm 0.05\%$ of final value	(1)
2Eh	Current 4...20mA Siemens S5-format	1.185 .. +22.96mA / 22.96mA = max. value before over range (20480) 20mA = rated range (0...16384) 0mA = min. value before under range (-4096) value and sign	<sup>1)</sup> $\pm 0.2\%$ of final value	(1)
2Fh	Current $\pm 20\text{mA}$ Siemens S5-format	$\pm 23.70\text{mA}$ / 23.70mA = max. value before over range (19456) -20...20mA = rated value (-16384...16384) -23.70mA = min. value before under range (-19456) value and sign	<sup>1)</sup> $\pm 0.05\%$ of final value	(1)
32h	Resistance measurement 6000Ohm four-wire	- / 6000 $\Omega$ = final value (32767)	<sup>1)2)</sup> $\pm 0.05\%$ of final value	(4)
33h	Resistance measurement 6000Ohm four-wire	- / 6000 $\Omega$ = final value (6000)	<sup>1)2)</sup> $\pm 0.05\%$ of final value	(4)
35h	Resistance measurement 60Ohm two-wire	- / 60 $\Omega$ = final value (6000)	<sup>1)2)3)</sup> $\pm 0.2\%$ of final value	(3)
36h	Resistance measurement 600Ohm two-wire	- / 600 $\Omega$ = final value (6000)	<sup>1)2)3)</sup> $\pm 0.1\%$ of final value	(3)
37h	Resistance measurement 3000Ohm two-wire	- / 3000 $\Omega$ = final value (30000)	<sup>1)2)3)</sup> $\pm 0.1\%$ of final value	(3)
38h	Resistance measurement 6000Ohm two-wire	- / 6000 $\Omega$ = final value (6000)	<sup>1)2)3)</sup> $\pm 0.1\%$ of final value	(3)
<sup>6)</sup> 3Ah	Current $\pm 20\text{mA}$ S5-Format from Siemens	$\pm 23.70\text{mA}$ / 23.70mA = max. value before over range (19456) -20...20mA = nominal range (-16384...16384) -23.70mA = min. value before under range (-19456) two's complement	<sup>1)</sup> $\pm 0.05\%$ of final value	(1)
<sup>6)</sup> 3Bh	Voltage $\pm 10\text{V}$ S5-Format from Siemens	$\pm 11.85\text{V}$ / 12.5V = max. value before over range (20480) -10...10V = nominal range (-16384...16384) -12.5V = min. value before under range (-20480) two's complement	<sup>1)</sup> $\pm 0.2\%$ of final value	(1)
3Dh	Resistance measurement 60Ohm four-wire	- / 60 $\Omega$ = final value (6000)	<sup>1)2)</sup> $\pm 0.1\%$ of final value	(4)

continue ...

continue...

3Eh	Resistance measurement 600Ohm four-wire	- / 600Ω= final value (6000)	<sup>1)</sup> <sup>2)</sup> ±0.05% of final value	(4)
3Fh	Resistance measurement 3000Ohm four-wire	- / 3000Ω = final value (30000)	<sup>1)</sup> <sup>2)</sup> ±0.05% of final value	(4)
57h	Voltage 0...50mV	0...50mV / 59.25mV = max. value before over range (5925) 0...50mV = rated range (0...5000) two's complement	<sup>1)</sup> ±0.1% of final value	(1)
58h	Voltage ±10V	±11.85V / 11.85V= max. value before over range (11850) -10...10V= rated range (-10000...10000) -11.85V= min. value before under range (-11850) two's complement	<sup>1)</sup> ±0.05% of final value	(1)
59h	Voltage ±4V	±4.74V / 4.74V = max. value before over range (47400) -4...4V = rated range (-40000...40000) -4.74V = min. value before under range (-47400) two's complement	<sup>1)</sup> ±0.05% of final value	(1)
5Ah	Voltage ±400mV	±0.474V / 474mV = max. value before over range (47400) -400...400mV = rated range (-40000...40000) -474mV = min. value before under range (-47400) two's complement	<sup>1)</sup> ±0.1% of final value	(1)
5Ch	Current ±20mA	±23.70mA / 23.70mA = max. value before over range (23700) -20...20mA = rated value (-20000...20000) -23.70mA = min. value before under range (-23700) two's complement	<sup>1)</sup> ±0.05% of final value	(1)
5Dh	Current 4...20mA	1.185 .. +22.96mA / 22.96mA = max. value before over range (18960) 4...20mA = rated range (0...16000) 0mA = min. value before under range (-4000) two's complement	<sup>1)</sup> ±0.05% of final value	(1)
FFh	Channel not active (turned off)			

<sup>1)</sup> measured at an environmental temperature of 25°C, velocity of 15 conversions/s

<sup>2)</sup> excluding errors caused by transducer inaccuracies

<sup>3)</sup> excluding errors caused by contact resistance and line resistance

<sup>4)</sup> the compensation of the neutralization must be implemented externally

<sup>5)</sup> the compensation for the neutralization is implemented internally by including the temperature of the front plug. The thermal conductors have to be connected directly to the front plug, and where necessary these must be extended by means of thermo element extension cables

<sup>6)</sup> starting from hardware release 11

**Note!**

The module is preset to the range "±10V voltage".

**Numeric notation in S5 from Siemens**

In S5 format, the input data are stored in one word. The word consists of the binary value and the information bits.

*Numeric notation:*

Byte	Bit 7 ... Bit 0
0	Bit 0: overflow bit 0: value within measuring range 1: measuring range overrun Bit 1: error bit (set at internal error) Bit 2: activity bit (always 0) Bit 3 ... 7: binary measured value
1	Bit 0 ... 6: binary measured value Bit 7: sign 0 positive 1 negative

**+/- 10V (two's complement)**

Voltage	Decimal	Hex
-10V	-16384	C000
-5V	-8192	E000
0V	0	0000
+5V	+8192	2000
+10V	+16384	4000

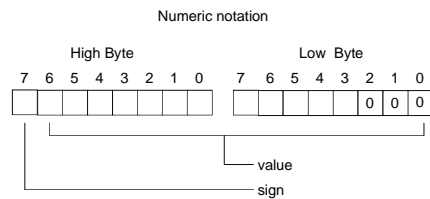
Formulas for the calculation:

$$Value = 16384 \cdot \frac{U}{10}, \quad U = Value \cdot \frac{10}{16384}$$

U: voltage, Value: Decimal value

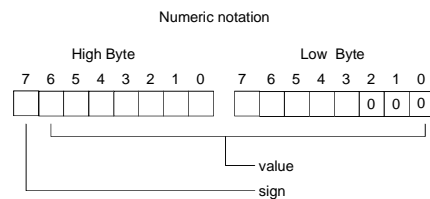
**+/- 10V (value and sign)**

Voltage	Decimal	Hex
-10V	-16384	C000
-5V	-8192	A000
0V	0	0000
+5V	+8192	2000
+10V	+16384	4000



**4...20mA (value and sign)**

Current	Decimal	Hex
+4mA	0	0000
+12mA	+8192	2000
+20mA	+16384	4000



**+/- 20mA (two's complement)**

Current	Decimal	Hex
-20mA	-16384	C000
-10mA	-8192	E000
0mA	0	0000
+10mA	+8192	2000
+20mA	+16384	4000

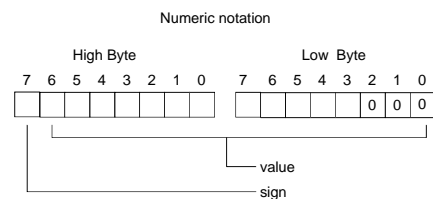
Formula for the calculation:

$$Value = 16384 \cdot \frac{I}{20}, \quad I = Value \cdot \frac{20}{16384}$$

I: Current, Value: Decimal value

**+/- 20mA (value and sign)**

Current	Decimal	Hex
-20mA	-16384	C000
-10mA	-8192	A000
0mA	0	0000
+10mA	+8192	2000
+20mA	+16384	4000



### Numeric notation in S7 from Siemens

Analog values are represented as a two's complement value.

*Numeric notation:*

Byte	Bit 7 ... Bit 0
0	Bit 0 ... 7: binary measured value
1	Bit 0 ... 6: binary measured value Bit 7: sign 0 positive 1 negative

#### +/- 10V

Voltage	Decimal	Hex
-10V	-27648	9400
-5V	-13824	CA00
0V	0	0
+5V	13824	3600
+10V	+27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{U}{10}, \quad U = Value \cdot \frac{10}{27648}$$

U: voltage, Value: decimal value

#### 0...10V

Voltage	Decimal	Hex
0V	0	0000
5V	8192	2000
10V	16384	4000

Formulas for the calculation:

$$Value = 16384 \cdot \frac{U}{10}, \quad U = Value \cdot \frac{10}{16384}$$

U: voltage, Value: decimal value

#### 1...5V

Voltage	Decimal	Hex
+1V	0	0
+3V	+13824	3600
+5V	+27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{U-1}{4}, \quad U = Value \cdot \frac{4}{27648} + 1$$

U: voltage, Value: decimal value

#### +/-4V

Voltage	Decimal	Hex
-4V	-27648	9400
0V	0	0
4V	27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{U}{4}, \quad U = Value \cdot \frac{4}{27648}$$

U: voltage, Value: decimal value

#### +/-400mV

Voltage	Decimal	Hex
-400mV	-27648	9400
0V	0	0
400mV	27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{U}{400}, \quad U = Value \cdot \frac{400}{27648}$$

U: voltage, Value: decimal value

#### 4...20mA

Current	Decimal	Hex
+4mA	0	0
+12mA	+13824	3600
+20mA	+27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{I-4}{16}, \quad I = Value \cdot \frac{16}{27648} + 4$$

I: current, Value: decimal value

#### +/- 20mA

Current	Decimal	Hex
-20mA	-27648	9400
-10mA	-13824	CA00
0mA	0	0
+10mA	+13824	3600
+20mA	+27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{I}{20}, \quad I = Value \cdot \frac{20}{27648}$$

I: current, Value: decimal value

**Measurement data acquisition**

During a measurement the data is stored in the data input area. The table above shows the allocation of the data to a measured value as well as the respective tolerance.

The following figure shows the structure of the data input area:

*Data input area:*

Byte	Bit 7 ... Bit 0
0	High-Byte channel 0
1	Low-Byte channel 0
2	High-Byte channel 1
3	Low-Byte channel 1
4	High-Byte channel 2
5	Low-Byte channel 2
6	High-Byte channel 3
7	Low-Byte channel 3

**Note!**

Only channels 0 and 2 are used in four-wire systems.

**Parameter data**

You may configure every channel individually. 10Byte are available for the configuration data. Configuration parameters are stored in permanent memory and they will be retained even if power is turned off.

The following table show the structure of the parameter area:

*Parameter area:*

Byte	Bit 7 ... Bit 0	Default
0	Diagnostic alarm byte: Bit 0 ... 5: reserved Bit 6: 0: diagnostic alarm inhibited 1: diagnostic alarm enabled Bit 7: reserved	00h
1	reserved	00h
2	Function no. channel 0 (see table)	2Dh
3	Function no. channel 1 (see table)	2Dh
4	Function no. channel 2 (see table)	2Dh
5	Function no. channel 3 (see table)	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

**Parameters***Diagnostic alarm*

The diagnostic alarm is enabled by means of Bit 6 of Byte 0. In this case an error a 4Byte diagnostic message will be issued to the master system.

*Function no.*

Here you have to enter the function number of your measurement function for every channel. The allocation of the function number to a measurement function is available from the table above.

*Option byte*

Here you may specify the conversion rate. In addition selection and envelope functions have been implemented.

**Note!**

Please note that the resolution is reduced when conversion rate is increased due to the shorter integration time.

The format of the data transfer remains the same. The only difference is that the lower set of bits (LSBs) lose significance for the analog value.

*Structure of the option byte:*

Byte	Bit 7 ... Bit 0	Resolution	Default
6 ... 9	Option byte: Bit 0 ... 3: rate* 0000 15 conversions/s 0001 30 conversions/s 0010 60 conversions/s 0011 123 conversions/s 0100 168 conversions/s 0101 202 conversions/s 0110 3.7 conversions/s 0111 7.5 conversions/s Bit 4 ... 5: Selection function 00 deactivated 01 use 2 of 3 values 10 use 4 of 6 values Bit 6 ... 7: Envelope function 00 deactivated 01 envelope $\pm 8$ 10 envelope $\pm 16$	16 16 15 14 12 10 16 16	00h

\*) These specifications apply to 1channel operation. For multi-channel operations, the conversion rate per channel can be calculated by dividing the specified conversion rate by the number of active channels.



**Diagnostic data**

As soon as you activated the alarm release in Byte 0 of the parameter area, 4 diagnostic Bytes with fixed content are transferred to the superordinated system in case of an error. Please note that analog modules only use the first two bytes for diagnostic purposes. The remaining two byte are not used. The structure of the diagnostic bytes is as follows:

*Diagnostic data:*

Byte	Bit 7 ... Bit 0	Default
0	Bit 0: Module malfunction Bit 1: constant 0 Bit 2: external error Bit 3: channel error present Bit 4 ... 7: reserved	-
1	Bit 0 ... 3 class of module 0101 analog module Bit 4: channel information available	-
2 ... 3	not assigned	-

**Technical data**

Electrical data	VIPA 231-1BD52
Number of inputs	4 differential inputs
Input resistance	> 2M $\Omega$ (voltage range) < 50 $\Omega$ (current range)
measuring range	
- Thermo element	Typ J, K, N, R, S, T
- Resistance thermometer	Pt100, Pt1000, NI100, NI1000
- Resistance measuring	60 $\Omega$ , 600 $\Omega$ , 3k $\Omega$
- Voltage measuring	0...50mV, 0...10V, $\pm$ 4mV, $\pm$ 4V, $\pm$ 10V
- Current measuring	4...20mA, $\pm$ 20mA
Power supply	5V via backplane bus
Current consumption	240mA via backplane bus
Isolation	500Vrms (field voltage - backplane bus)
Status indicators	via LEDs on the front
Programming specifications	
Input data	8Byte (1 word per channel)
Output data	-
Parameter data	10Byte
Diagnostic data	4Byte
Process alarm data	-
Dimensions and weight	
Dimensions (WxHxD)	25.4x76x76mm
Weight	100g

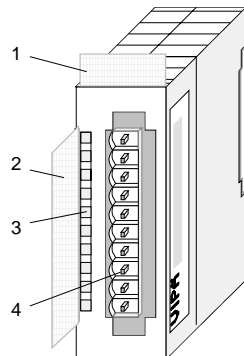
## AI 4x12Bit, 4 ... 20mA, isolated

**Order data** AI 4x12Bit, 4...20mA, isolated VIPA 231-1BD60

**Description** The module has 4 inputs that are permanently configured to measure current signals (4 ... 20mA). This module requires a total of 8Byte of the process image for the input data (2Byte per channel).  
The measured values are returned in S5 format from Siemens. DC/DC converters and isolation amplifiers are employed to provide electrical isolation for the channels of the module with respect to the backplane bus and between the different channels.

- Properties**
- 4 inputs, channels isolated from the backplane bus and from each other (galvanic isolation of the channels by means of isolation amplifiers)
  - Permanently configured for current measurements
  - No parameterization required
  - Suitable for transducers with 4 ... 20mA outputs
  - LEDs to indicate wire break

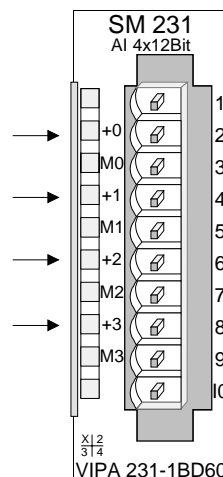
**Construction**



- [1] Label for the name of the module
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

**Status indicator pin assignment**

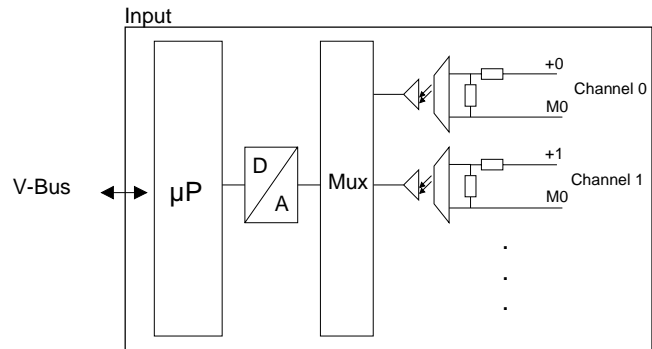
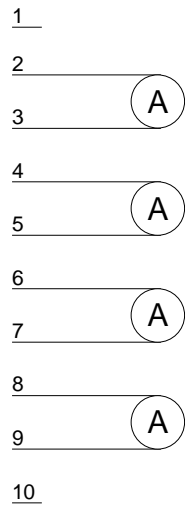
LED	Description
+0 ... +3	LED (red) wire break detection These LEDs is turned on when the transducer is disconnected.



Pin	Assignment
1	
2	pos. connection Ch. 0
3	Channel 0 common
4	pos. connection Ch.1
5	Channel 1 common
6	pos. connection Ch.2
7	Channel 2 common
8	pos. connection Ch.3
9	Channel 3 common
10	

**Wiring and schematic diagram**

**Wiring diagram      Schematic diagram**



**Wire break recognition**

The wire break recognition is always active. In case of a wire break res. when no encoder is connected, the LED of the according channel is turned on. The module has no diagnostic ability.

**Numeric notation**

Input data in Siemens S5 format is stored in a word. The word contains the binary value and information bits:

*Numeric notation:*

Byte	Bit 7 ... Bit 0
0	Bit 0: overflow bit 0: value within measuring range 1: measuring range exceeded Bit 1: error bit (set at internal error) Bit 2: activity bit (always 0) Bit 3 ... 7: <b>binary measured value</b> (see table below)
1	Bit 0 ... 6: <b>binary measured value</b> (see table below) Bit 7: sign 0 positive 1 negative

The following table shows the allocation of binary values to the respective measured values.

**Numeric notation in Siemens S5 format**

Measured value in mA	Units	Binary measured value	T	E	Ü	Range
24.0	2560	0 1 0 1 0 0 0 0 0 0 0 0 0 0	0	0	0	overdrive region
20.016	2049	0 1 0 0 0 0 0 0 0 0 0 0 0 1	0	0	0	
20.0	2048	0 1 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	nominal range
19.98	2047	0 0 1 1 1 1 1 1 1 1 1 1 1 1	0	0	0	
12.0	1024	0 0 1 0 0 0 0 0 0 0 0 0 0 0	0	0	0	
8.0	512	0 0 0 1 0 0 0 0 0 0 0 0 0 0	0	0	0	
6.0	256	0 0 0 0 1 0 0 0 0 0 0 0 0 0	0	0	0	
5.0	128	0 0 0 0 0 1 0 0 0 0 0 0 0 0	0	0	0	
4.016	2	0 0 0 0 0 0 0 0 0 0 0 0 0 1 0	0	0	0	
4.008	1	0 0 0 0 0 0 0 0 0 0 0 0 0 0 1	0	0	0	
4	0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	
3.984	-2	1 1 1 1 1 1 1 1 1 1 1 1 1 1 0	0	0	0	Underdrive region
3.0	-128	1 1 1 1 1 1 0 0 0 0 0 0 0 0	0	0	0	
2.0	-256	1 1 1 1 1 0 0 0 0 0 0 0 0 0	0	0	0	
1.0	-384	1 1 1 1 0 1 0 0 0 0 0 0 0 0	0	0	0	
0.0	-512	1 1 1 1 0 0 0 0 0 0 0 0 0 0	0	0	0	

**Technical data**

Electrical data	VIPA 231-1BD60
Number of inputs	4 individually isolated
Current measuring range	4 ... 20mA
Input filter time delay	3ms
Input resistance	20Ω
Power supply	5V via backplane bus
Current consumption	280mA via backplane bus
Isolation	yes, every channel separately, isolation tested at 500Vrms
Status indicators	via LEDs on the front
Programming specifications	
Input data	8Byte (1 word per channel)
Output data	-
Parameter data	-
Diagnostic data	-
Process alarm data	-
Dimensions and weight	
Dimensions (WxHxD)	25.4x76x76mm
Weight	120g

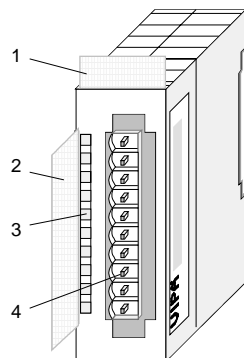
## AI 4x12Bit, ±10V, isolated

**Order data** AI 4x12Bit, ±10V, isolated VIPA 231-1BD70

**Description** The module has 4 inputs that are permanently configured to measure voltage signals (±10V). This module requires a total of 8Byte of the process image for the input data (2Byte per channel). The measured values are returned in S5 format from Siemens. DC/DC converters and isolation amplifiers are employed to provide electrical isolation for the channels of the module with respect to the backplane bus and between the different channels.

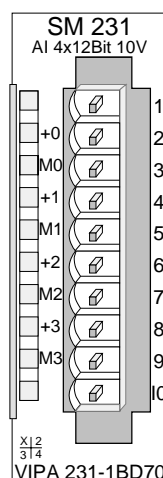
- Properties**
- 4 inputs, channels isolated from the backplane bus and from each other (Galvanic isolation of the channels by means of isolation amplifiers)
  - Permanently configured for voltage measurements
  - No parameterization required
  - Suitable for transducers with ±10V outputs

**Construction**



- [1] Label for the name of the module
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

**Pin assignment**

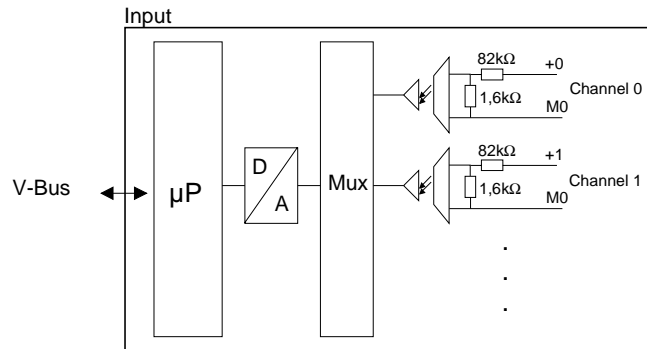
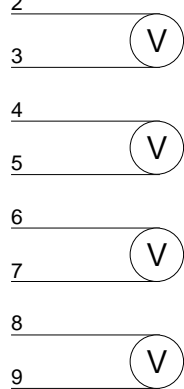


Pin	Assignment
1	
2	pos. connection Channel 0
3	Channel 0 common
4	pos. connection Channel 1
5	Channel 1 common
6	pos. connection Channel 2
7	Channel 2 common
8	pos. connection Channel 3
9	Channel 3 common
10	

**Wiring and schematic diagram**

**Wiring diagram      Schematic diagram**

1  
2  
3  
4  
5  
6  
7  
8  
9  
10



**Numeric notation**

Input data in Siemens S5 format is stored in a word. The word contains the binary value and information bits:

*Numeric notation:*

Byte	Bit 7 ... Bit 0
0	Bit 0: overflow bit 0: value within measuring range 1: measuring range exceeded Bit 1: error bit (set at internal error) Bit 2: activity bit (always 0) Bit 3 ... 7: <b>binary measured value</b> (see table below)
1	Bit 0 ... 6: <b>binary measured value</b> (see table below) Bit 7: sign 0 positive 1 negative

The following table shows the allocation of binary values to the respective measured values.

**Numeric notation  
in Siemens  
S5 format**

Measured value in V	Units	Binary measured value	T	E	Ü	Range
12,5	2560	0 1 0 1 0 0 0 0 0 0 0 0 0 0	0	0	0	overdrive region
10,005	2049	0 1 0 0 0 0 0 0 0 0 0 0 0 1	0	0	0	
10,0	2048	0 1 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	nominal range
5	1024	0 0 1 0 0 0 0 0 0 0 0 0 0 0	0	0	0	
2,5	512	0 0 0 1 0 0 0 0 0 0 0 0 0 0	0	0	0	
1,25	256	0 0 0 0 1 0 0 0 0 0 0 0 0 0	0	0	0	
0,625	128	0 0 0 0 0 1 0 0 0 0 0 0 0 0	0	0	0	
0,005	1	0 0 0 0 0 0 0 0 0 0 0 0 0 1	0	0	0	
0	0	0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	
-0,005	-1	1 1 1 1 1 1 1 1 1 1 1 1 1 1	0	0	0	
-0,625	-128	1 1 1 1 1 1 0 0 0 0 0 0 0 0	0	0	0	
-1,25	-256	1 1 1 1 1 0 0 0 0 0 0 0 0 0	0	0	0	
-2,5	-512	1 1 1 1 0 0 0 0 0 0 0 0 0 0	0	0	0	
-5	-1024	1 1 1 0 0 0 0 0 0 0 0 0 0 0	0	0	0	
-10,0	-2048	1 1 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	
-10,005	-2049	1 0 1 1 1 1 1 1 1 1 1 1 1 1	0	0	0	Underdrive region
-12	-2560	1 0 1 1 0 0 0 0 0 0 0 0 0 0	0	0	0	

**Technical data**

Electrical data	VIPA 231-1BD70
Number of inputs	4 individually isolated
Voltage measuring range	±10V
Input filter time delay	3ms
Input resistance	83.5kΩ
Power supply	5V via backplane bus
Current consumption	300mA via backplane bus
Isolation	yes, every channel separately, isolation tested at 500Vrms
Status indicators	via LEDs on the front
Programming specifications	
Input data	8Byte (1 word per channel)
Output data	-
Parameter data	-
Diagnostic data	-
Process alarm data	-
Dimensions and weight	
Dimensions (WxHxD)	25.4x76x76mm
Weight	120g

## AI 4x16Bit f

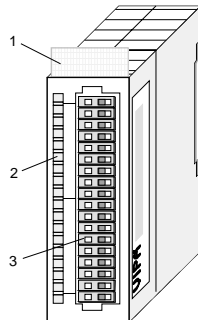
**Order data** AI 4x16Bit f VIPA 231-1FD00

**Description** The module has 4 fast (f=fast) inputs that you may configure individually. The module requires a total of 8 input data bytes in the process image (2Byte per channel).

Isolation between the channels on the module and the backplane bus is provided by means of DC/DC converters and optocouplers.

- Properties**
- Using all 4 channels, the cycle time is < 1ms
  - the different channels are individually configurable and can be turned off
  - LED for signaling wire break in current loop operation
  - Diagnostic function
  - Resolution 16Bit

**Construction**



- [1] Label for the name of the module
- [2] LED status indicator
- [3] Edge connector

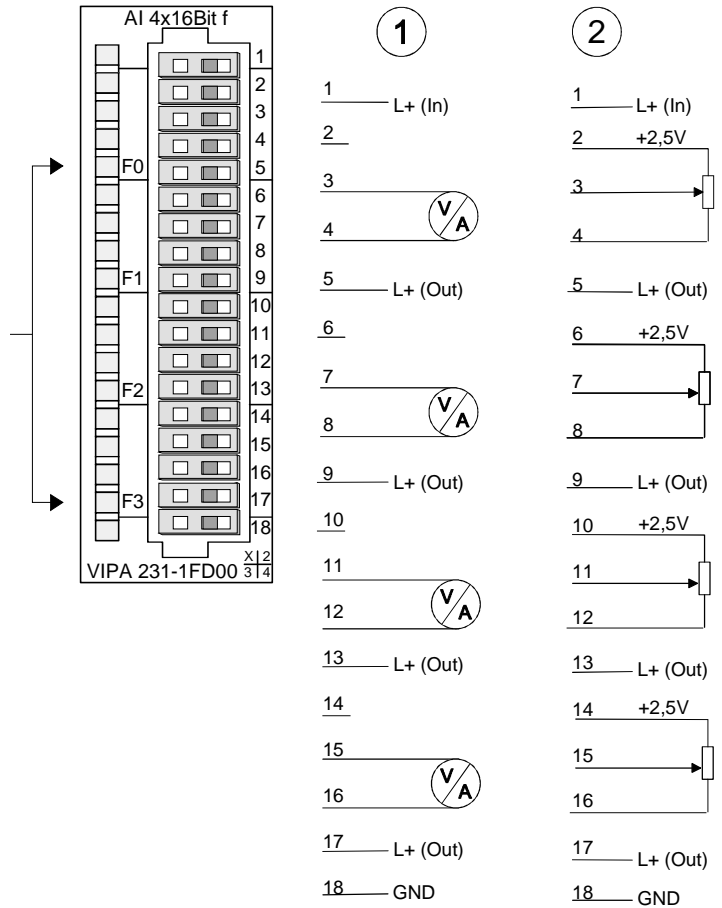


**Status indicator pin assignment**

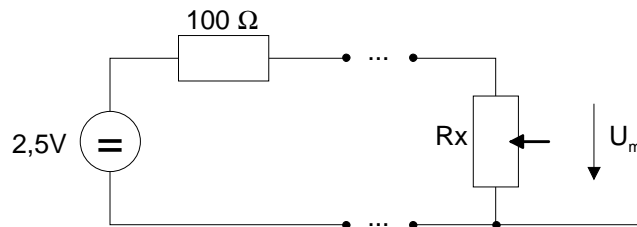
**LED Description**

F0 ... F3 LED (red):  
is on if the measured current value exceeds the range 4...20mA (cable break or overload).

**Wiring diagram**



The internal resistance  $R_i$  of the 2,5V voltage source is 100Ω. For a connection of a voltage divider the following equivalent circuit is valid:



**Note!**

Unused inputs on activated channels have to be connected to the respective ground. This is not necessary when the unused channels are turned off by means of FFh.

**Function no. allocation**      The assignment of a function no. to a certain channel happens during parameterization. The function no. 00h does not influence the function no. stored in the permanent parameterization data.  
Assigning FFh deactivates the according channel.

No.	Function	Measurement range / representation	Tolerance	Conn.
00h	Does not affect permanently stored configuration data			
28h	Voltage $\pm 10V$ S7-Format from Siemens	-10V ... -9.9V (-27648 ... -27371) <sup>1)</sup> -9.9 ... 9.9 (-27370 ... 27370) -10...10V= rated range (-27648...27648) 9.9 ... 10V (27371 ... 27648) <sup>1)</sup>	0,2% of final value	(1), (2)
29h	Voltage $\pm 4V$ S7-Format from Siemens	$\pm 4.70V$ / 4.70V = max. value before over range (32511) -4...4V = rated range (-27648...27648) -4.70V = min. value before under range (-32512) two's complement	0,2% of final value	(1), (2)
2Ah	Voltage $\pm 400mV$ S7-Format from Siemens	$\pm 0.47V$ / 470mV = max. value before over range (32511) -400...400mV = rated range(-27648...27648) -470mV = min. value before under range (-32512) two's complement	0,4% of final value	(1)
2Ch	Current $\pm 20mA$ S7-Format from Siemens	$\pm 23.51mA$ / 23.51mA = max. value before over range (32511) -20...20mA = rated range (-27648...27648) -23.51mA = min. value before under range (-32512) two's complement	0,2% of final value	(1)
2Dh	Current 4...20mA S7-Format from Siemens	1.185 .. +22.81mA / 22.81mA = max. value before over range (32511) 4...20mA = rated range (0...27648) 1.18mA = min. value before under range (-4864) two's complement	0,5% of final value	(1)
58h	Voltage $\pm 10V$	-10 ... -9.9V (-10000 ... -9901) <sup>1)</sup> -9,9 ... 9.9V (-9900 ... 9900) -10...10V= rated range(-10000...10000) 9.9 ... 10V (9901 ... 10000) <sup>1)</sup>	0,2% of final value	(1), (2)
59h	Voltage $\pm 4V$	$\pm 4.95V$ / 4,95V = max. value before over range (4950) -4...4V = rated range (-4000...4000) -4.95V = min. value before under range (-4950) two's complement	0,2% of final value	(1), (2)
5Ah	Voltage $\pm 400mV$	$\pm 0.495V$ / 495mV = max. value before over range (4950) -400...400mV = rated range (-4000...4000) -495mV = min. value before under range (-4950) two's complement	0,4% of final value	(1)
5Ch	Current $\pm 20mA$	$\pm 25mA$ / 25mA = max. value before over range (25000) -20...20mA = rated range (-20000...20000) -25mA = min. value before under range(-25000) two's complement	0,2% of final value	(1)
5Dh	Current 4...20mA	0.8 .. +24.00mA / 24.00mA = Ende Übersteuerungsbereich (20000) 4...20mA = rated range (0...16000) 0,8mA = min. value before under range (-3200) two's complement	0,5% of final value	(1)
FFh	Channel not active (turned off)			

<sup>1)</sup> depends on calibration factor and is not guaranteed.

**Note!**

The module is preset to the range "±10V voltage" in S7 format from Siemens.

**Numeric notation  
in S7 from  
Siemens**

Analog values are represented as a two's complement value.

*Numeric notation:*

Byte	Bit 7 ... Bit 0
0	Bit 0 ... 7: binary measured value
1	Bit 0 ... 6: binary measured value Bit 7: sign 0 positive 1 negative

## +/- 10V

Voltage	Decimal	Hex
-10V	-27648	9400
-5V	-13824	CA00
0V	0	0
+5V	13824	3600
+10V	+27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{U}{10}, \quad U = Value \cdot \frac{10}{27648}$$

U: voltage, Value: decimal value

## +/-4V

Voltage	Decimal	Hex
-4V	-27648	9400
0V	0	0
4V	27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{U}{4}, \quad U = Value \cdot \frac{4}{27648}$$

U: voltage, Value: decimal value

## +/-400mV

Voltage	Decimal	Hex
-400mV	-27648	9400
0V	0	0
400mV	27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{U}{400}, \quad U = Value \cdot \frac{400}{27648}$$

U: voltage, Value: decimal value

## 4....20mA

Current	Decimal	Hex
+4mA	0	0
+12mA	+13824	3600
+20mA	+27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{I - 4}{16}, \quad I = Value \cdot \frac{16}{27648} + 4$$

I: current, Value: decimal value

## +/- 20mA

Current	Decimal	Hex
-20mA	-27648	9400
-10mA	-13824	CA00
0mA	0	0
+10mA	+13824	3600
+20mA	+27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{I}{20}, \quad I = Value \cdot \frac{20}{27648}$$

I: current, Value: decimal value

**Measurement data acquisition**

During a measurement the data is stored in the data input area. The table above shows the allocation of the data to a measured value as well as the respective tolerance.

The following figures show the structure of the data input area:

*Data input area:*

Byte	Bit 7 ... Bit 0
0	High-Byte channel 0
1	Low-Byte channel 0
2	High-Byte channel 1
3	Low-Byte channel 1
4	High-Byte channel 2
5	Low-Byte channel 2
6	High-Byte channel 3
7	Low-Byte channel 3

**Parameter data**

You may configure every channel individually. 32Byte are available for the configuration data. Configuration parameters are stored in permanent memory and they will be retained even if power is turned off.

The following table show the structure of the parameter area:

*Parameter area:*

Byte	Bit 7 ... Bit 0	Default
0	Diagnostic alarm byte: Bit 0 ... 5: reserved Bit 6: 0: diagnostic alarm inhibited 1: diagnostic alarm enabled Bit 7: reserved	00h
1	Limit value monitoring: Bit 0: limit value monitoring channel 0 Bit 1: limit value monitoring channel 1 Bit 2: limit value monitoring channel 2 Bit 3: limit value monitoring channel 3 Bit 4 ... 7: reserved	00h
2	Function no. channel 0 (see table)	28h
3	Function no. channel 1 (see table)	28h
4	Function no. channel 2 (see table)	28h
5	Function no. channel 3 (see table)	28h
6-9	reserved	00h

*Continue ...*

... Continue parameter area

Byte	Bit 7 ... Bit 0	Default
10	Bit 0 ... 2: mean value 000: disabled 001: mean value over 2 values 010: mean value over 4 values 011: mean value over 8 values 100: mean value over 16 values 101, 011, 111: disabled Bit 3 ... 7: reserved	00h
11-15	reserved	00h
16	channel 0, upper limit, High-Byte	7Fh
17	channel 0, upper limit, Low-Byte	FFh
18	channel 0, lower limit, High-Byte	80h
19	channel 0, lower limit, Low-Byte	00h
20	channel 1, upper limit, High-Byte	7Fh
21	channel 1, upper limit, Low-Byte	FFh
22	channel 1, lower limit, High-Byte	80h
23	channel 1, lower limit, Low-Byte	00h
24	channel 2, upper limit, High-Byte	7Fh
25	channel 2, upper limit, Low-Byte	FFh
26	channel 2, lower limit, High-Byte	80h
27	channel 2, lower limit, Low-Byte	00h
28	channel 3, upper limit, High-Byte	7Fh
29	channel 3, upper limit, Low-Byte	FFh
30	channel 3, lower limit, High-Byte	80h
31	channel 3, lower limit, Low-Byte	00h

**Diagnostic data**

The diagnostic data have a size of 12Byte and are stored in the record sets 0 and 1 of the system data area.

As soon as you activated the alarm release in Byte 0 of the parameter area, in case of an error *record set 0* is transferred to the superordinated system.

*Record set 0* has a fixed content and a length of 4Byte. The contents of *record set 0* may be monitored in plain text via the diagnosis window of the CPU.

For extended diagnostic purposes during runtime, you may evaluate the *record set 1* with a size of 12Byte via the SFCs 51 and 59.

**Evaluate diagnosis**

At a diagnostic task the CPU interrupts the user application and branches into OB82. With according programming, you may request in this OB with the SFCs 51 and 59 detailed diagnostic information and react on it.

After execution of the OB82, the processing of the user application is continued. The diagnostic data remains consistent until leaving the OB82.

**Record set 0**

*Byte 0 to 3:*

*Record set 0 (Byte 0 to 3):*

Byte	Bit 7 ... Bit 0	Default
0	Bit 0: error in module Bit 1: reserved Bit 2: external error Bit 3: channel error Bit 4 ... 6: reserved Bit 7: wrong parameter in module	00h
1	Bit 0 ... 3: module class 0101 analog module Bit 4: channel information present Bit 5 ... 7: reserved	15h
2	not used	00h
3	Bit 0 ... 5: reserved Bit 6: missing (lost) process alarm (see process alarm) Bit 7: reserved	00h

**Record set 1***Byte 0 to 11:*

Record set 1 contains the 4Byte of record set 0 and 8Byte module specific diagnostic data.

The diagnostic bytes have the following assignment:

*Record set 1 (Byte 0 to 11):*

Byte	Bit 7 ... Bit 0	Default
0 ... 3	content of record set 0 (see page above)	-
4	Bit 0 ... 6: channel type 70h: digital input 71h: analog input 72h: digital output 73h: analog output Bit 7: reserved	71h
5	Bit 0 ... 7: number of diagnostic output bits per channel	04h
6	Bit 0 ... 7: number of similar channels of a module	04h
7	Bit 0: channel error channel 0 Bit 1: channel error channel 1 Bit 2: channel error channel 2 Bit 3: channel error channel 3 Bit 4 ... 7: reserved	00h
8	Bit 0: reserved Bit 1: parameterization error channel 0 Bit 2 ... 4: reserved Bit 5: parameterization error channel 1 Bit 6, 7: reserved	00h
9	Bit 0: reserved Bit 1: parameterization error channel 2 Bit 2 ... 4: reserved Bit 5: parameterization error channel 3 Bit 6, 7: reserved	00h
10 ... 11	reserved	00h

**Process alarm**

The upper and the lower limit value is parameterizable for every channel. Please regard during parameterization that you have to enable the limit value monitoring in parameter byte 1.

If the signal is beyond the defined operation range, a process alarm is initialized. In the CPU, the process alarm block (OB40) is called.

The 4Byte of process alarm additional information are used as follows:

*Process alarm additional information*

Byte	Bit 7 ... Bit 0	Default
0	Bit 0: upper limit exceeded channel 0 Bit 1: upper limit exceeded channel 1 Bit 2: upper limit exceeded channel 2 Bit 3: upper limit exceeded channel 3 Bit 4 ... 7: reserved	00h
1	Bit 0: lower limit underrun channel 0 Bit 1: lower limit underrun channel 1 Bit 2: lower limit underrun channel 2 Bit 3: lower limit underrun channel 3 Bit 4 ... 7: reserved	00h
2	reserved	00h
3	reserved	00h

**Note!**

When a process alarm has not yet been acknowledged by the CPU and a new process alarm of the same type occurs at this channel, a diagnostic alarm is initialized, containing the information "Process alarm missing/lost" (diagnostic data Byte 3).



**Technical data**

Electrical data	VIPA 231-1FD00
Number of inputs	4 differential inputs
Cycle time (all channels)	< 1ms
measuring range	
- Voltage measuring	$\pm 400\text{mV}$ , $\pm 4\text{V}$ , $\pm 10\text{V}$
- Current measuring	4...20mA, $\pm 20\text{mA}$
Input resistance	> $2\text{M}\Omega$ (voltage range) < $57\Omega$ (current range)
Power supply	5V via backplane bus
Current consumption	300mA via backplane bus
Isolation	500Vrms (field voltage - backplane bus)
Status indicators	via LEDs on the front
Programming specifications	
Input data	8Byte (1 word per channel)
Output data	-
Parameter data	32Byte
Diagnostic data	12Byte
Process alarm data	4Byte
Dimensions and weight	
Dimensions (WxHxD)	25.4x76x76mm
Weight	100g

## AI 8x16Bit

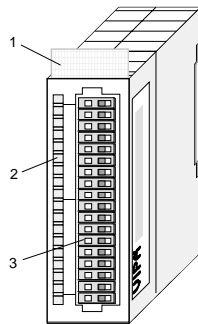
**Order data** AI 8x16Bit VIPA 231-1BF00

**Description** The analog input module transfers analog signals from the process into digital signals for the internal processing.  
As transducer you may connect thermo couplers type J, K, T and resistance thermometer Pt100.  
The modules has 8 inputs that you may configure in groups of two channels individually.

**Properties**

- 8 analog inputs
- wire break detection
- resolution 15Bit + sign

### Construction



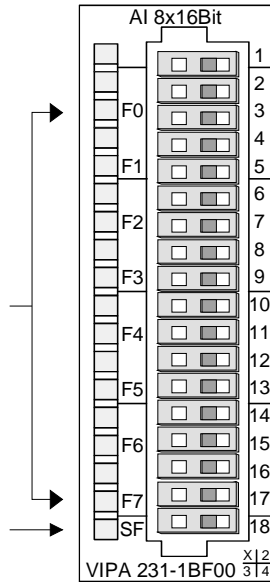
- [1] Label for the name of the module
- [2] LED status indicator
- [3] Edge connector

**Status indicator  
pin assignment**

**LED Description**

F0...F7 LED (red):  
error for each channel

SF LED (red):  
sum error



**Pin Assignment**

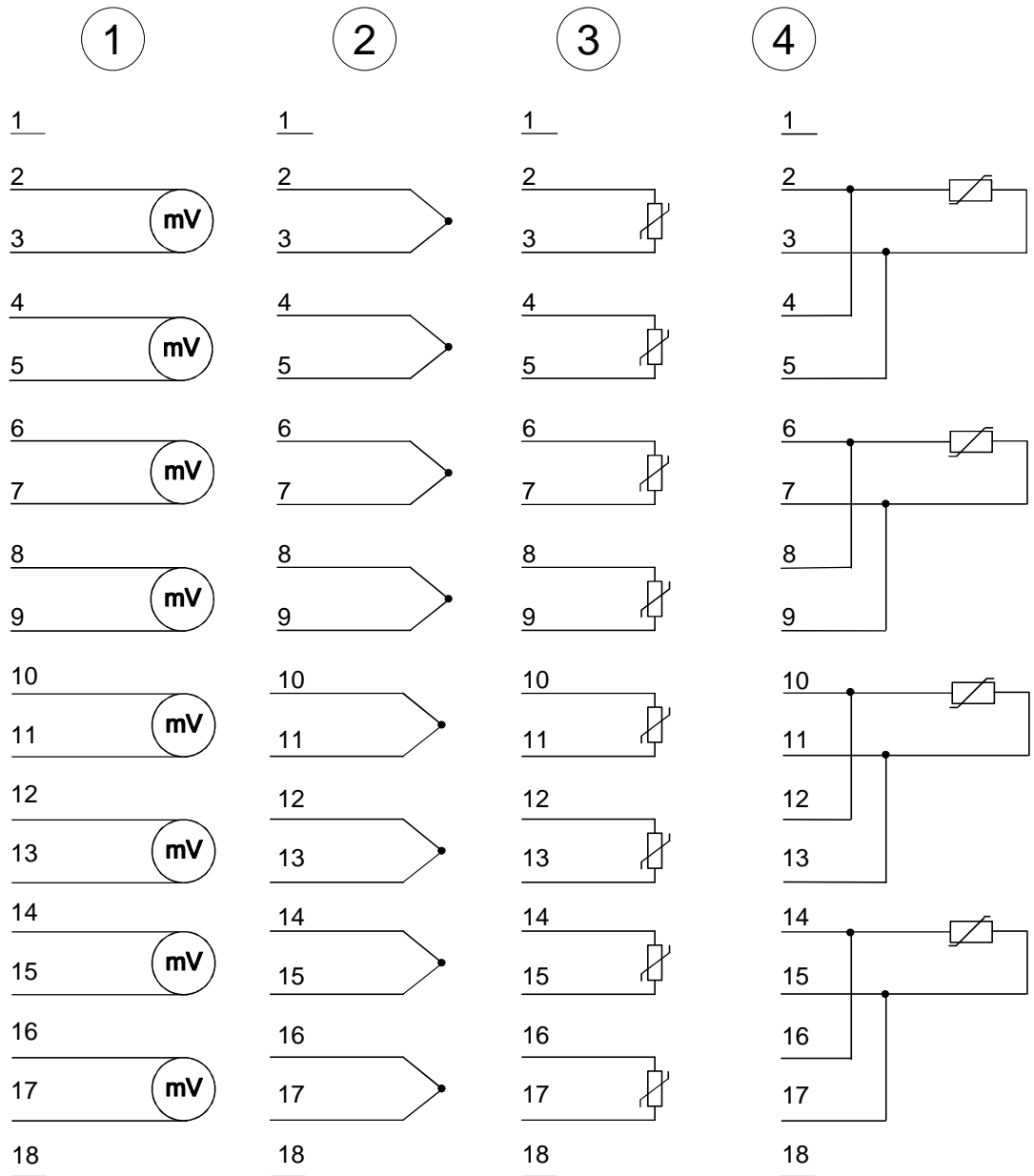
- 1 not connected
- 2 pos. connection K.0
- 3 Channel 0 common
- 4 pos. connection K.1
- 5 Channel 1 common
- 6 pos. connection K.2
- 7 Channel 2 common
- 8 pos. connection K.3
- 9 Channel 3 common
- 10 pos. connection K.4
- 11 Channel 4 common
- 12 pos. connection K.5
- 13 Channel 5 common
- 14 pos. connection K.6
- 15 Channel 6 common
- 16 pos. connection K.7
- 17 Channel 7 common
- 18 not connected



**Note!**

Unused inputs on activated channels have to be connected to the respective ground. This is not necessary when the unused channels are turned off by means of FFh.

Connection diagram



**Function no. assignment** The assignment of a function no. to a certain channel happens during parameterization. The function no. 00h does not influence the function no. stored in the permanent parameterization data. Assigning FFh deactivates the according channel.

No.	Function	Measurement range / representation	Tolerance ref. to nominal range	Conn.
00h	Does not affect permanently stored configuration data			
01h	RTD Pt100 in two-wire mode	-200 .. +850°C / in units of 1/10°C, two's complement	<sup>1)2)3)</sup> ±0.1%	(3)
61h	RTD Pt100 in two-wire mode	-328 .. 1562°F in units of 1/10°F, two's complement	<sup>1)2)3)</sup> ±0.1%	(3)
09h	RTD Pt100 via four-wire connection	-200 .. +850°C / in units of 1/10°C, two's complement	<sup>1)2)</sup> ±0.1%	(4)
69h	RTD Pt100 via four-wire connection	-328 .. 1562°F in units of 1/10°F, two's complement	<sup>1)2)</sup> ±0.1%	(4)
10h	Thermo element type J, externally compensated	0 °C .. 1000 °C / in units of 1/10°C, two's complement	<sup>1)2)4)</sup> ±0.1%	(2)
40h	Thermo element type J, externally compensated	32 .. 1832°F in units of 1/10°F, two's complement	<sup>1)2)4)</sup> ±0.1%	(2)
11h	Thermo element type K, externally compensated	0 °C .. 1300 °C / in units of 1/10°C, two's complement	<sup>1)2)4)</sup> ±0.1%	(2)
41h	Thermo element type K, externally compensated	32 .. 2372°F in units of 1/10°F, two's complement	<sup>1)2)4)</sup> ±0.1%	(2)
14h	Thermo element type T, externally compensated	-200 °C .. +400 °C / in units of 1/10°C, two's complement	<sup>1)2)4)</sup> -200...-60.1 ±0.5% -60...400 ±0,2%	(2)
44h	Thermo element type T, externally compensated	-328 .. 752°F in units of 1/10°F, two's complement	<sup>1)2)4)</sup> -328...-76,1 ±0.5% -76...752 ±0,2%	(2)
18h	Thermo element type J, internally compensated	0 °C .. 1000 °C / in units of 1/10°C, two's complement	<sup>1)2)5)</sup> ±1,0%	(2)
48h	Thermo element type J, internally compensated	32-1832°F in units of 1/10°F, two's complement	<sup>1)2)5)</sup> ±1,0%	(2)
19h	Thermo element type K, internally compensated	0 °C .. 1300 °C / in units of 1/10°C, two's complement	<sup>1)2)5)</sup> ±1,0%	(2)
49h	Thermo element type K, internally compensated	32-2372°F in units of 1/10°F, two's complement	<sup>1)2)5)</sup> ±1,0%	(2)
1Ch	Thermo element type T, internally compensated	-200 °C .. +400 °C / in units of 1/10°C, two's complement	<sup>1)2)5)</sup> ±2,0%	(2)
4Ch	Thermo element type T, internally compensated	-328 .. 752°F in units of 1/10°F, two's complement	<sup>1)2)5)</sup> ±2,0%	(2)
26h	Voltage 0...60mV	0...60mV = nominal range (0-27648)	<sup>1)</sup> ±0.1%	(1)
56h	Voltage 0...60mV	0...60mV = nominal range (0-6000) in units of 1/100mV	<sup>1)</sup> ±0.1%	(1)
FFh	Channel not active (off)			

<sup>1)</sup> measured at an ambient temperature of 25°C, velocity of 15 conversions/s

<sup>2)</sup> excluding errors caused by transducer inaccuracies

<sup>3)</sup> excluding errors caused by contact resistance and line resistance

<sup>4)</sup> the compensation of the neutralization has to be implemented externally

<sup>5)</sup> the compensation for the neutralization is implemented internally by including the temperature of the front plug. The thermal conductors have to be connected directly to the front plug, and where necessary these have to be extended by means of thermo element extension cables.

**Measurement data acquisition**

During a measurement, the data is stored in the data input area. The table above shows the allocation of the data to a measured value as well as the respective tolerance.

The following figures show the structure of the data input area:

*Data input area:*

Byte	Bit 7 ... Bit 0
0	High-Byte channel 0
1	Low-Byte channel 0
2	High-Byte channel 1
3	Low-Byte channel 1
4	High-Byte channel 2
5	Low-Byte channel 2
6	High-Byte channel 3
7	Low-Byte channel 3
8	High-Byte channel 4
9	Low-Byte channel 4
10	High-Byte channel 5
11	Low-Byte channel 5
12	High-Byte channel 6
13	Low-Byte channel 6
14	High-Byte channel 7
15	Low-Byte channel 7

**Note!**

Only channels 0, 2, 4 and 6 are used in four-wire systems.

**Parameter data**

You may configure the channels in groups of two individually. 10Byte are available for the configuration data. Configuration parameters are stored in permanent memory and they will be retained even if power is turned off.

The following table show the structure of the parameter area:

*Parameter area:*

Byte	Bit 7 ... Bit 0	Default
0	Diagnostic alarm byte: Bit 0: 0: wire break recognition channel 0/1 off 1: wire break recognition channel 0/1 on Bit 1: 0: wire break recognition channel 2/3 off 1: wire break recognition channel 2/3 on Bit 2: 0: wire break recognition channel 4/5 off 1: wire break recognition channel 4/5 on Bit 3: 0: wire break recognition channel 6/7 off 1: wire break recognition channel 6/7 on Bit 4, 5: reserved Bit 6: 0: diagnostic alarm inhibited 1: diagnostic alarm enabled Bit 7: reserved	0Fh
1	reserved	00h
2	Function no. channel 0/1 (see table)	26h
3	Function no. channel 2/3 (see table)	26h
4	Function no. channel 4/5 (see table)	26h
5	Function no. channel 6/7 (see table)	26h
6	Option Byte channel 0/1	00h
7	Option Byte channel 2/3	00h
8	Option Byte channel 4/5	00h
9	Option Byte channel 6/7	00h

**Parameter***Diagnostic alarm*

The diagnostic alarm is enabled by means of Bit 6 of Byte 0. In this case an error a 4Byte diagnostic message will be issued to the master system.

*Function no.*

Here you have to enter the function number of your measurement function for 2 channels. The allocation of the function number to a measurement function is available from the table above.

*Option-Byte*

Here you may specify for 2 channels the conversion rate.

**Note!**

Please note that the resolution is reduced when conversion rate is increased due to the shorter integration time.

The format of the data transfer remains the same. The only difference is that the lower set of bits (LSBs) lose significance for the analog value.

*Structure of the option byte:*

Byte	Bit 7 ... Bit 0	Resolution	Default
6 ... 9	Option byte: Bit 0 ... 3: rate* 0000 15 conversions/s 0001 30.1 conversions/s 0010 60 conversions/s 0011 123.2 conversions/s 0100 168.9 conversions/s 0101 202.3 conversions/s 0110 3.76 conversions/s 0111 7.51 conversions/s Bit 4 ... 7: reserved	16 16 15 14 12 10 16 16	00h

\*) These specifications apply to 1channel operation. For multi-channel operations the conversion rate per channel can be calculated by dividing the specified conversion rate by the number of active channels.



**Diagnostic data**

The diagnostic data have a size of 12Byte and are stored in the record sets 0 and 1 of the system data area.

As soon as you activated the alarm release in Byte 0 of the parameter area, in case of an error *record set 0* is transferred to the superordinated system.

*Record set 0* has a fixed content and a length of 4Byte. The contents of *record set 0* may be monitored in plain text via the diagnosis window of the CPU.

For extended diagnostic purposes during runtime, you may evaluate the *record set 1* with a size of 12Byte via the SFCs 51 and 59.

**Evaluate diagnosis**

At a diagnostic task the CPU interrupts the user application and branches into OB82. With according programming, you may request in this OB with the SFCs 51 and 59 detailed diagnostic information and react on it.

After execution of the OB82, the processing of the user application is continued. The diagnostic data remains consistent until leaving the OB82.

**Record set 0**

*Byte 0 to 3:*

*Record set 0 (Byte 0 to 3):*

Byte	Bit 7 ... Bit 0	Default
0	Bit 0: error in module Bit 1: reserved Bit 2: external error Bit 3: channel error Bit 4 ... 6: reserved Bit 7: wrong parameter in module	00h
1	Bit 0 ... 3: module class 0101 analog module Bit 4: channel information present Bit 5 ... 7: reserved	15h
2	not used	00h
3	Bit 0 ... 5: reserved Bit 6: missing (lost) process alarm (see process alarm) Bit 7: reserved	00h

**Record set 1**

Byte 0 to 11:

Record set 1 contains the 4Byte of record set 0 and 8Byte module specific diagnostic data.

The diagnostic bytes have the following assignment:

*Record set 1 (Byte 0 to 11):*

Byte	Bit 7 ... Bit 0	Default
0 ... 3	content of record set 0 (see page above)	-
4	Bit 0 ... 6: channel type 70h: digital input 71h: analog input 72h: digital output 73h: analog output Bit 7: reserved	71h
5	Bit 0 ... 7: number of diagnostic output bits per channel	04h
6	Bit 0 ... 7: number of similar channels of a module	04h
7	Bit 0: Channel error channel 0 Bit 1: Channel error channel 1 Bit 2: Channel error channel 2 Bit 3: Channel error channel 3 Bit 4: Channel error channel 4 Bit 5: Channel error channel 5 Bit 6: Channel error channel 6 Bit 7: Channel error channel 7	00h
8	Bit 0: Wire break channel 0 Bit 1: Parameterization error channel 0 Bit 2: Measuring range underflow channel 0 Bit 3: Measuring range overflow channel 0 Bit 4: Wire break channel 1 Bit 5: Parameterization error channel 1 Bit 6: Measuring range underflow channel 1 Bit 7: Measuring range overflow channel 1	00h
9	Bit 0: Wire break channel 2 Bit 1: Parameterization error channel 2 Bit 2: Measuring range underflow channel 2 Bit 3: Measuring range overflow channel 2 Bit 4: Wire break channel 3 Bit 5: Parameterization error channel 3 Bit 6: Measuring range underflow channel 3 Bit 7: Measuring range overflow channel 3	00h

*Continue ...*

... Continue

Byte	Bit 7 ... Bit 0	Default
10	Bit 0: Wire break channel 4 Bit 1: Parameterization error channel 4 Bit 2: Measuring range underflow channel 4 Bit 3: Measuring range overflow channel 4 Bit 4: Wire break channel 5 Bit 5: Parameterization error channel 5 Bit 6: Measuring range underflow channel 5 Bit 7: Measuring range overflow channel 5	00h
11	Bit 0: Wire break channel 6 Bit 1: Parameterization error channel 6 Bit 2: Measuring range underflow channel 6 Bit 3: Measuring range overflow channel 6 Bit 4: Wire break channel 7 Bit 5: Parameterization error channel 7 Bit 6: Measuring range underflow channel 7 Bit 7: Measuring range overflow channel 7	00h

**Technical data**

Electrical data	VIPA 231-1BF00
Number of inputs	8
Input resistance	> 2M $\Omega$
measuring range	
- Thermo element	Type J, K, T
- Resistance thermometer	Pt100
- Voltage measuring	0...60mV
Power supply	5V via backplane bus
Current consumption	280mA via backplane bus
Isolation	500Vrms (field voltage - backplane bus)
Dissipation power	typ. 1.3W
Status indicators	via LEDs on the front
Programming specifications	
Input data	16Byte (1 word per channel)
Output data	-
Parameter data	10Byte
Diagnostic data	12Byte
Process alarm data	-
Dimensions and weight	
Dimensions (WxHxD)	25.4x76x76mm
Weight	120g



## Chapter 17 Analog output modules

### Overview

This chapter contains a description of the construction and the operation of the VIPA analog output modules.

Below follows a description of:

- A system overview of the analog output modules
- Properties
- Constructions
- Interfacing and schematic diagram
- Technical data

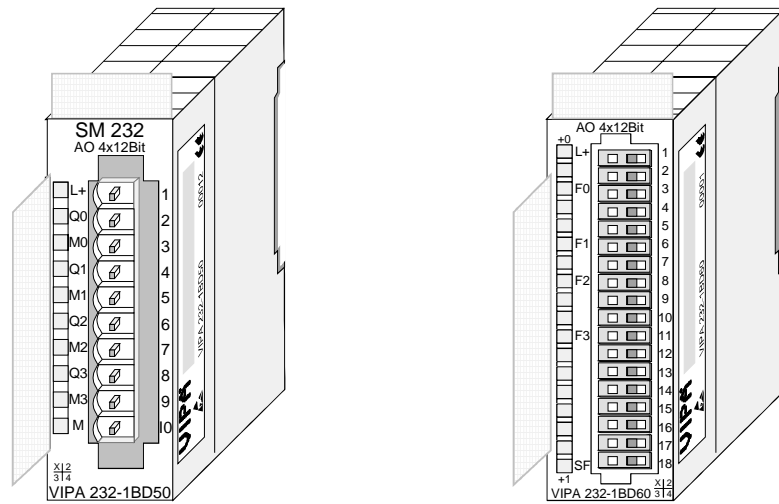
### Contents

Topic	Page
<b>Chapter 17 Analog output modules</b> .....	<b>17-1</b>
System overview.....	17-2
General.....	17-3
AO 4x12Bit, multioutput.....	17-4
AO 4x12Bit f, multioutput.....	17-12

## System overview

### Output modules SM 232

Here follows a summary of the analog output modules that are currently available from VIPA:



### Order data output modules

Type	Order number	Seite
AO 4x12Bit, multi-output	VIPA 232-1BD50	17-4
AO 4x12Bit f, multi-output	VIPA 232-1BD60	17-12

## General

### Cabling for analog signals

You should only use screened twisted-pair cable when you are connecting analog signals. These cables reduce the effect of electrical interference. The screen of the analog signal cable should be grounded at both ends. In situations with different electrical potentials, it is possible that a current will flow to equalize the potential difference. This current could interfere with the analog signals. Under these circumstances it is advisable to ground the screen of the signal cable at one end only.

### Connecting loads and actuators

Due to the fact that actuators also require a source of external power they may be connected to actuators by means of 2 wires or 4 wires. Where control signals are supplied to 2wire actuators a power supply has to be connected in series with the control cable. 4wire actuators are connected to an external power source.



#### Note!

Please ensure that you connect actuators to the correct polarity!  
Unused output terminals are not connected!

### Parameterization and diagnosis during runtime

By using the SFCs 55, 56 and 57 you may change the parameters of the analog modules during runtime via the CPU 21x.

For diagnosis evaluation during runtime, you may use the SFCs 51 and 59. They allow you to request detailed diagnosis information and to react to it.

## AO 4x12Bit, multioutput

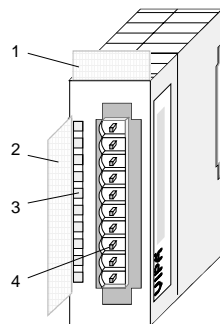
**Order data** AO 4x12Bit Multi-Output VIPA 232-1BD50

**Description** This module provides 4 outputs that can be configured individually. The module occupies a total of 8Byte of output data (2Byte per channel) in the process image. These values have to be defined as left justified two's complement entries.

Galvanic isolation between the channels on the module and the backplane bus is provided by means of DC/DC converters and optocouplers. The module requires an external supply of DC 24V.

- Properties**
- 4 outputs with common ground
  - Outputs with individually configurable functions
  - Suitable for connection to actuators requiring  $\pm 10V$ , 1 ... 5V, 0 ... 10V,  $\pm 20mA$ , 4 ... 20mA or 0 ... 20mA inputs
  - Diagnostic LED and diagnostic function

**Construction**



- [1] Label for the name of the module
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

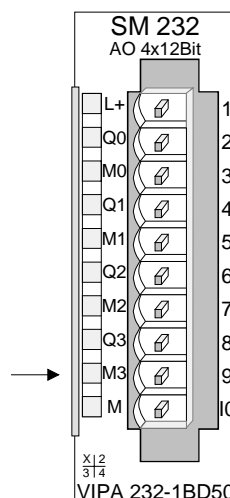
**Status indicator pin assignment**

**LED Description**

- M3 Diagnostic LED (red) turned on by:
- a short circuit is detected at the control voltage output
  - an open circuit is detected on the current output line
  - the CPU is in STOP mode
  - the bus coupler does not receive supply voltage

**Pin Assignment**

- 1 DC 24V supply voltage
- 2 + Channel 0
- 3 Channel 0 common
- 4 + Channel 1
- 5 Channel 1 common
- 6 + Channel 2
- 7 Channel 2 common
- 8 + Channel 3
- 9 Channel 3 common
- 10 Supply voltage common



**Note!**

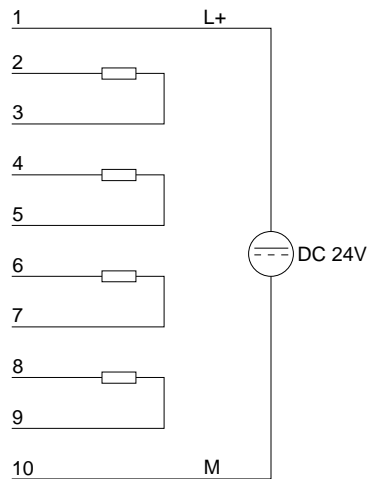
Please note that the diagnostic LEDs of the **entire** module are denoted M3!



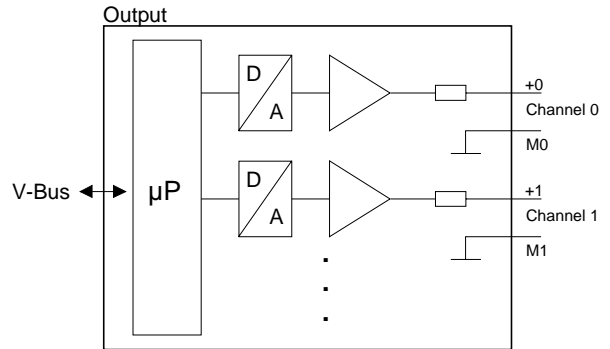


**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Function no. allocation**

No.	Function	Output range	Tolerance
00h	no output		
01h	Voltage $\pm 10V$ Siemens S5-format	$\pm 12.5V$ 12.5V = max. value before over range (20480) -10...10V = rated range (-16384...16384) -12.5V = min. value before under range (-20480)	<sup>1)</sup> $\pm 0.2\%$ of final value
02h	Voltage 1...5V Siemens S5-format	0...6V 6V = max. value before over range (20480) 1...5V = rated range (0...16384) 0V = min. value before under range (-4096)	<sup>1)</sup> $\pm 0.2\%$ of final value
05h	Voltage 0...10V Siemens S5-format	0...12.5V 12.5V = max. value before over range (20480) 0...10V = rated range (0...16384) no under range available	<sup>1)</sup> $\pm 0.2\%$ of final value
09h	Voltage $\pm 10V$ Siemens S7-format (two's complement)	$\pm 11.85V$ 11.85V = max. value before over range (32767) -10V...10V = rated range (-27648...27648) -11.85 = min. value before under range (-32767)	<sup>1)</sup> $\pm 0.2\%$ of final value
0Ah	Voltage 1...5V Siemens S7-format (two's complement)	0...5.75V 5.75V = max. value before over range (32767) 1...5V = rated range (0...27648) 0V = min. value before under range (-6912)	<sup>1)</sup> $\pm 0.2\%$ of final value
0Dh	Voltage 0...10V Siemens S7-format (two's complement)	0...11.85V 11.85V = max. value before over range (32767) 0...10V = rated range (0...27648) no under range available	<sup>1)</sup> $\pm 0.2\%$ of final value

continue ...

... continue

No.	Function	Output range	Tolerance
03h	Current $\pm 20\text{mA}$ Siemens S5-format	$\pm 25\text{mA}$ 25mA = max. value before over range (20480) -20...20mA = rated range (-16384...16384) -25mA = min. value before under range (-20480)	<sup>1)</sup> $\pm 0.2\%$ of final value
04h	Current 4...20mA Siemens S5-format	0...25mA 25mA = max. value before over range (20480) 4...20mA = rated range (0...16384) 0mA = min. value before under range (-4096)	<sup>1)</sup> $\pm 0.2\%$ of final value
06h	Current 0...20mA Siemens S5-format	0...25mA 25mA = max. value before over range (20480) 0...20mA = rated range (0...16384) no under range available	<sup>1)</sup> $\pm 0.2\%$ of final value
0Bh	Current $\pm 20\text{mA}$ Siemens S7-format (two's complement)	$\pm 23.70\text{mA}$ 23.70mA = max. value before over range (32767) -20...20mA = rated range (-27648...27648) -23.70mA = min. value before under range (-32767)	<sup>1)</sup> $\pm 0.2\%$ of final value
0Ch	Current 4...20mA Siemens S7-format (two's complement)	0...22.96mA 22.96mA = max. value before over range (32767) 4...20mA = rated range (0...27648) 0mA = min. value before under range (-5530)	<sup>1)</sup> $\pm 0.2\%$ of final value
0Eh	Current 0...20mA Siemens S7-format (two's complement)	0...23.70mA 23.70mA = max. value before over range (32767) 0...20mA = rated range (0...27648) no under range available	<sup>1)</sup> $\pm 0.2\%$ of final value

<sup>1)</sup> determined at an ambient temp. of 25°C, conversion rate of 15/s

**Numeric notation in Siemens S5-format**

In Siemens S5-format, input data is saved in a word. The word consists of the binary value and the information bits.

*Numeric notation:*

Byte	Bit 7 ... Bit 0
0	Bit 0: overflow bit 0: value within measuring range 1: measuring range exceeded Bit 1: error bit (set by internal errors) Bit 2: activity bit (always 0) Bit 3 ... 7: binary measured value
1	Bit 0 ... 6: binary measured value Bit 7: sign 0 positive 1 negative

+/- 10V

Voltage	Decimal	Hex
-10V	-16384	C000
-5V	-8192	E000
0V	0	0
+5V	8192	2000
+10V	+16384	4000

Formulas for the calculation:

$$Value = 16384 \cdot \frac{U}{10}, \quad U = Value \cdot \frac{10}{16384}$$

U: voltage, Value: decimal value

0...10V

Voltage	Decimal	Hex
0V	0	0000
5V	8192	2000
10V	16384	4000

Formulas for the calculation:

$$Value = 16384 \cdot \frac{U}{10}, \quad U = Value \cdot \frac{10}{16384}$$

U: voltage, Value: decimal value

1...5V

Voltage	Decimal	Hex
+1V	0	0
+3V	+8192	2000
+5V	+16384	4000

Formulas for the calculation:

$$Value = 16384 \cdot \frac{U-1}{4}, \quad U = Value \cdot \frac{4}{16384} + 1$$

U: voltage, Value: decimal value

4....20mA

Current	Decimal	Hex
+4mA	0	0
+12mA	+8192	2000
+20mA	+16384	4000

Formulas for the calculation:

$$Value = 16384 \cdot \frac{I-4}{16}, \quad I = Value \cdot \frac{16}{16384} + 4$$

I: current, Value: decimal value

+/- 20mA

Current	Decimal	Hex
-20mA	-16384	C000
-10mA	-8192	E000
0mA	0	0
+10mA	+8192	2000
+20mA	+16384	4000

Formulas for the calculation:

$$Value = 16384 \cdot \frac{I}{20}, \quad I = Value \cdot \frac{20}{16384}$$

I: current, Value: decimal value

### Numeric notation in Siemens S7-format

The analog values is represented in two's complement format.

*Numeric representation:*

Byte	Bit 7 ... Bit 0
0	Bit 0 ... 7: binary measured vale
1	Bit 0 ... 6: binary measured vale Bit 7: sign 0 positive 1 negative

+/- 10V

Voltage	Decimal	Hex
-10V	-27648	9400
-5V	-13824	CA00
0V	0	0
+5V	13824	3600
+10V	+27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{U}{10}, \quad U = Value \cdot \frac{10}{27648}$$

U: voltage, Value: decimal value

0...10V

Voltage	Decimal	Hex
0V	0	0000
5V	8192	2000
10V	16384	4000

Formulas for the calculation:

$$Value = 16384 \cdot \frac{U}{10}, \quad U = Value \cdot \frac{10}{16384}$$

U: voltage, Value: decimal value

1...5V

Voltage	Decimal	Hex
+1V	0	0
+3V	+13824	3600
+5V	+27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{U-1}{4}, \quad U = Value \cdot \frac{4}{27648} + 1$$

U: voltage, Value: decimal value

+/-4V

Voltage	Decimal	Hex
-4V	-27648	9400
0V	0	0
4V	27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{U}{4}, \quad U = Value \cdot \frac{4}{27648}$$

U: voltage, Value: decimal value

+/-400mV

Voltage	Decimal	Hex
-400mV	-27648	9400
0V	0	0
400mV	27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{U}{400}, \quad U = Value \cdot \frac{400}{27648}$$

U: voltage, Value: decimal value

4....20mA

Current	Decimal	Hex
+4mA	0	0
+12mA	+13824	3600
+20mA	+27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{I-4}{16}, \quad I = Value \cdot \frac{16}{27648} + 4$$

I: current, Value: decimal value

+/- 20mA

Current	Decimal	Hex
-20mA	-27648	9400
-10mA	-13824	CA00
0mA	0	0
+10mA	+13824	3600
+20mA	+27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{I}{20}, \quad I = Value \cdot \frac{20}{27648}$$

I: current, Value: decimal value

**Data output**

The value of the output data is entered into the data output area. For every channel you may configure the relationship between the output value and the respective current or voltage by means of a function no.

The following table shows the structure of the data output area:

*Data output area:*

Byte	Bit 7 ... Bit 0
0	High-Byte channel 0
1	Low-Byte channel 0
2	High-Byte channel 1
3	Low-Byte channel 1
4	High-Byte channel 2
5	Low-Byte channel 2
6	High-Byte channel 3
7	Low-Byte channel 3

**Parameter data**

6Byte of parameter data are available for the configuration data. These parameters are stored in non-volatile memory and are available after the unit has been powered off.

The following table shows the structure of the parameter data:

*Parameter area:*

Byte	Bit 7 ... Bit 0
0	Diagnostic alarm byte: Bit 0 ... 5: reserved Bit 6: 0: diagnostic alarm inhibited 1: diagnostic alarm enabled Bit 7: reserved
1	reserved
2	Function-no. channel 0
3	Function-no. channel 1
4	Function-no. channel 2
5	Function-no. channel 3

**Parameter***Diagnostic alarm*

You can enable diagnostic alarms by means of Bit 6 of Byte 0. When an error occurs 4 diagnostic bytes are transmitted to the master system.

*Function no.*

Here you enter the function no. of the output function for every channel. The relationship between the function number and the output functions is available from the function no. allocation table.

**Diagnostic data**

When you enable alarms in Byte 0 of the parameter area, modules will transfer 4 diagnostic bytes with pre-defined contents to your master in case of an error. Please note that analog modules only use the first two bytes for diagnostic purposes. The remaining bytes are not used.

The structure of the diagnostic bytes is as follows:

*Diagnostic data:*

Byte	Bit 7 ... Bit 0	Default
0	Bit 0: Module malfunction Bit 1: Constant 0 Bit 2: External error Bit 3: Channel error present Bit 4 ... 7: reserved	-
1	Bit 0 ... 3: class of module 0101 analog module Bit 4: channel information available	-
2	not assigned	-
3	not assigned	-

**Technical data**

Electrical data	VIPA 232-1BD50
Number of outputs	4
Voltage range	$\pm 10V$ , 1 ... 5V, 0 ... 10V
Current range	$\pm 20mA$ , 4 ... 20mA, 0 ... 20mA
Actuator resistance	min. 500 $\Omega$ (voltage range) max. 500 $\Omega$ (current range)
Short circuit current	30mA
Power supply	5V via backplane bus 24V $\pm 20\%$ externally
Current consumption	via backplane bus: 20mA DC 24V externally: 200mA
Isolation	500Vrms (field voltage - backplane bus)
Status indicators	via LEDs on the front
Programming specifications	
Input data	-
Output data	8Byte (1 word per channel)
Parameter data	6Byte
Diagnostic data	4Byte
Dimensions and weight	
Dimensions (WxHxD)	25.4x76x76mm
Weight	100g

## AO 4x12Bit f, multioutput

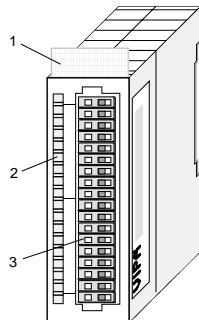
**Order data** AO 4x12Bit f multioutput VIPA 232-1BD60

**Description** This module provides 4 outputs that may be configured individually. The module occupies a total of 8Byte of output data (2Byte per channel) in the process image.

Galvanic isolation between the channels on the module and the backplane bus is provided by means of DC/DC converters and optocouplers. The module requires an external supply of DC 24V.

- Properties**
- Using all 4 channels, the cycle time is  $< 600\mu\text{s}$
  - 4 outputs with common grounds
  - Outputs with individually configurable functions
  - Suitable for actuators requiring  $\pm 10\text{V}$ ,  $0 \dots 10\text{V}$ ,  $-10\text{V} \dots 0$ ,  $0 \dots 20\text{mA}$  inputs
  - Diagnostic LED and diagnostic function
  - Resolution 12Bit

**Construction**

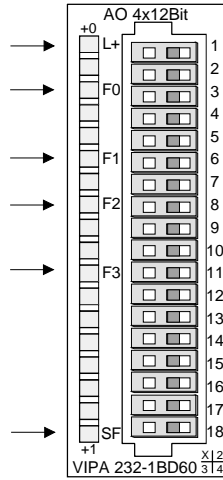


- [1] Label for the name of the module
- [2] LED status indicator
- [3] Edge connector



**Status indicator pin assignment**

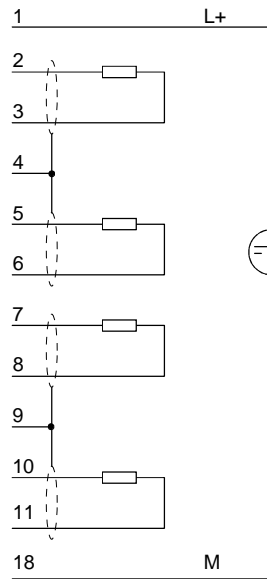
LED	Description
L+	LED (yellow) Supply voltage available
F0...F3	LED (green) for active channel
SF	LED (red) collective indication error



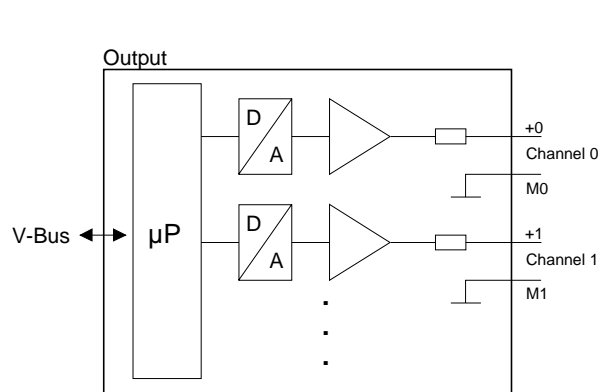
Pin	Assignment
1	DC 24V supply voltage
2	+ Channel 0
3	Channel 0 common
4	Screen Channel 0, 1
5	+ Channel 1
6	Channel 1 common
7	+ Channel 2
8	Channel 2 common
9	Screen Channel 2, 3
10	+ Channel 3
11	Channel 3 common
12...17	not connected
18	Supply voltage common

**Wiring and schematic diagram**

**Wiring diagram**



**Schematic diagram**



**Function no.  
allocation**

The assignment of a function no. to a certain channel happens during parameterization. The function no. 00h does not influence the function no. stored in the permanent parameterization data.

Assigning FFh deactivates the according channel.

No.	Function	Output range
00h	Does not affect permanently stored configuration data	
01h	Voltage $\pm 10V$ Siemens S5-format	$\pm 12.5V$ 12.5V = max. value before over range (20480) -10...10V = rated range (-16384...16384) -12.5V = min. value before under range (-20480)
05h	Voltage 0...10V Siemens S5-format	0...12.5V 12.5V = max. value before over range (20480) 0...10V = rated range (0...16384) no under range available
06h	Current 0...20mA Siemens S5-format	0...25mA 25mA = max. value before over range (20480) 0...20mA = rated range (0...16384) no under range available
07h	Voltage -10...0V Siemens S5-format	-12.5V...0V -12.5V = min. value before under range (-20480) -10V...0 = rated range (-16384...0) no over range available
09h	Voltage $\pm 10V$ Siemens S7-format (two's complement)	$\pm 11.76V$ 11.76V = max. value before over range (32511) -10V...10V = rated range (-27648...27648) -11.76 = min. value before under range (-32512)
0Dh	Voltage 0...10V Siemens S7-format (two's complement)	0...11.76V 11.76V = max. value before over range (32511) 0...10V = rated range (0...27648) no under range available
0Eh	Current 0...20mA Siemens S7-format (two's complement)	0...23.52mA 23.52mA = max. value before over range (32511) 0...20mA = rated range (0...27648) no under range available
0Fh	Voltage -10V...0V Siemens S7-format (two's complement)	-11.76V...0V -11.76V = min. value before under range (-32512) -10V...0 = rated range (-27648...0) no over range available
FFh	Channel not active (turned off)	

**Note!**

In all modes the value is 0 when over range or under range occurs.

The internal resistance of calibration for voltage ranges is 2.7 k $\Omega$ .

The internal resistance of calibration for current ranges is 35  $\Omega$ .

The Modul is at factory preset to the range " $\pm 10V$  voltage" in S7-format from Siemens.

**Numeric notation  
in Siemens  
S5 format**

In Siemens S5 format, input data is saved into a word. The word consists of the binary value and the information bits.

*Numeric notation:*

Byte	Bit 7 ... Bit 0
0	Bit 0: overflow bit 0: value within measuring range 1: measuring range exceeded Bit 1: error bit (set by internal errors) Bit 2: activity bit (always 0) Bit 3 ... 7: binary measured value
1	Bit 0 ... 6: binary measured value Bit 7: sign 0 positive 1 negative

+/- 10V

Voltage	Decimal	Hex
-10V	-16384	C000
-5V	-8192	E000
0V	0	0
+5V	8192	2000
+10V	+16384	4000

Formulas for the calculation:

$$Value = 16384 \cdot \frac{U}{10}, \quad U = Value \cdot \frac{10}{16384}$$

U: voltage, Value: decimal value

0...10V

Voltage	Decimal	Hex
0V	0	0000
5V	8192	2000
10V	16384	4000

Formulas for the calculation:

$$Value = 16384 \cdot \frac{U}{10}, \quad U = Value \cdot \frac{10}{16384}$$

U: voltage, Value: decimal value

-10...0V

Voltage	Decimal	Hex
-10V	-16384	C000
-5V	-8192	E000
0V	0	0000

Formulas for the calculation:

$$Value = 16384 \cdot \frac{U}{10}, \quad U = Value \cdot \frac{10}{16384}$$

U: voltage, Value: decimal value

0...20mA

Current	Decimal	Hex
+0mA	0	0
+10mA	+8192	2000
+20mA	+16384	4000

Formulas for the calculation:

$$Value = 16384 \cdot \frac{I - 4}{16}, \quad I = Value \cdot \frac{16}{16384} + 4$$

I: current, Value: decimal value

**Numeric notation  
in Siemens  
S7 format**

The analog values is represented in two's complement format.

*Numeric representation:*

Byte	Bit 7 ... Bit 0
0	Bit 0 ... 7: binary measured vale
1	Bit 0 ... 6: binary measured vale Bit 7: sign 0 positive 1 negative

+/- 10V

Voltage	Decimal	Hex
-10V	-27648	9400
-5V	-13824	CA00
0V	0	0
+5V	+13824	3600
+10V	+27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{U}{10}, \quad U = Value \cdot \frac{10}{27648}$$

U: voltage, Value: decimal value

0...10V

Voltage	Decimal	Hex
0V	0	0000
5V	+13824	3600
10V	+27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{U}{10}, \quad U = Value \cdot \frac{10}{27648}$$

U: voltage, Value: decimal value

-10...0V

Voltage	Decimal	Hex
-10V	-27648	9400
-5V	-13824	CA00
-0V	0	0000

Formulas for the calculation:

$$Value = 27648 \cdot \frac{U-1}{4}, \quad U = Value \cdot \frac{4}{27648} + 1$$

U: voltage, Value: decimal value

0....20mA

Current	Decimal	Hex
+0mA	0	0
+10mA	+13824	3600
+20mA	+27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{I-4}{16}, \quad I = Value \cdot \frac{16}{27648} + 4$$

I: current, Value: decimal value

**Data output**

The value of the output data has to be entered into the data output area. For every channel you may configure the relationship between the output value and the respective current or voltage by means of a function no.

The following table shows the structure of the data output area:

*Data output area:*

Byte	Bit 7 ... Bit 0
0	High-Byte channel 0
1	Low-Byte channel 0
2	High-Byte channel 1
3	Low-Byte channel 1
4	High-Byte channel 2
5	Low-Byte channel 2
6	High-Byte channel 3
7	Low-Byte channel 3

**Parameter data**

6Byte of parameter data are available for the configuration. These parameters are stored in non-volatile memory and are available after the unit has been powered off.

The following table shows the structure of the parameter data:

*Parameter area:*

Byte	Bit 7 ... Bit 0
0	Diagnostic alarm byte: Bit 0 ... 5: reserved Bit 6: 0: diagnostic alarm inhibited 1: diagnostic alarm enabled Bit 7: reserved
1	reserved
2	Function no. channel 0
3	Function no. channel 1
4	Function no. channel 2
5	Function no. channel 3

**Parameter***Diagnostic alarm*

You enable diagnostic alarms by means of Bit 6 of Byte 0. When an error occurs, 4 diagnostic bytes are transmitted to the master system.

*Function no.*

Here you enter the function no. of the output function for every channel. The relationship between the function number and the output functions is available from the function no. allocation table.

**Diagnostic data**

When you enable alarms in Byte 0 of the parameter area, modules will transfer 4 diagnostic bytes with pre-defined contents to your master if an error occurs. Please note that analog modules only use the first two bytes for diagnostic purposes. The remaining two bytes are not used.

The structure of the diagnostic bytes is as follows:

*Diagnostic data:*

Byte	Bit 7 ... Bit 0	Default
0	Bit 0: Module malfunction Bit 1: Constant 0 Bit 2: External error Bit 3: Channel error present Bit 4: External auxiliary supply error Bit 5: Open circuit of current out Short circuit of voltage out Bit 6: reserved Bit 7: Wrong parameter at the module	-
1	Bit 0 ... 3: class of module 0101 analog module Bit 4: channel information available	-
2	not assigned	-
3	not assigned	-

**Technical data**

Electrical data	VIPA 232-1BD60
Number of outputs	4
Cycle time (all 4 channel)	< 600 $\mu$ s
Voltage range	$\pm$ 10V, 0 ... 10V, -10V ... 0V
Current range	0 ... 20mA
Actuator resistance	min. 500 $\Omega$ (voltage range) max. 500 $\Omega$ (current range)
Short circuit current	30mA
Power supply	5V via backplane bus 24V $\pm$ 20% externally
Current consumption	via backplane bus: 50mA DC 24V externally: 200mA
Isolation	500Vrms (field voltage - backplane bus)
Status indicators	via LEDs on the front
Programming specifications	
Input data	-
Output data	8Byte (1 word per channel)
Parameter data	6Byte
Diagnostic data	4Byte
Dimensions and weight	
Dimensions (WxHxD)	25.4x76x76mm
Weight	100g





## Chapter 18 Analog input/output modules

### Overview

This chapter contains a description of the construction and the operation of the VIPA analog input/output modules.

Below follows a description of:

- A system overview of the analog input/output modules
- Properties
- Construction
- Wiring and schematic diagram
- Parameter data
- Function number allocation
- Technical data

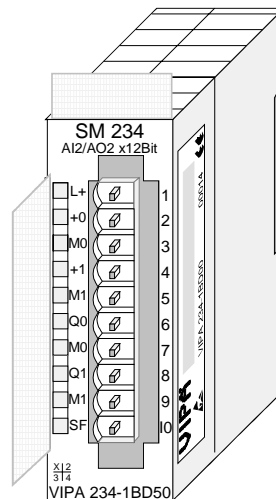
### Content

Topic	Page
<b>Chapter 18 Analog input/output modules .....</b>	<b>18-1</b>
System overview.....	18-2
Security note for range allocation.....	18-2
General.....	18-3
AI 2/AO 2x12Bit, multiin-/output.....	18-4

## System overview

### Input/output modules SM 234

Here follows a summary of the analog input/output modules that are currently available from VIPA:



### Order data input/output modules

Type	Order number
AIO 2x12Bit, multiin-/output	VIPA 234-1BD50

## Security note for range allocation



### Attention!

Please regard that the described module has no hardware protection against wrong parameterization. The allocation of the according measuring res. output range is only during project engineering.

For example, the module may be damaged when you connect a voltage at parameterized current measuring.

Please be extremely careful during project engineering.

## General

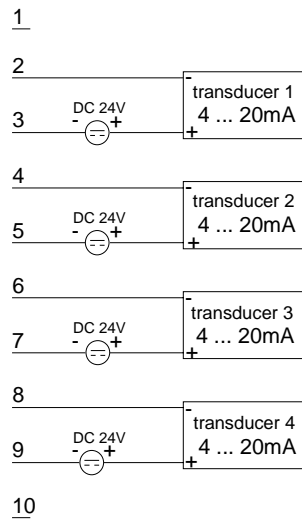
### Cabling for analog signals

You should only use screened twisted-pair cable for analog signals. These cables reduce the effect of electrical interference. The screen of the analog signal cable should be grounded at both ends. In situations where the cable ends are at different electrical potentials, it is possible that a current will flow to equalize the potential difference. This current could interfere with the analog signals. Under these circumstances it is advisable to ground the screen of the signal cable at one end only.

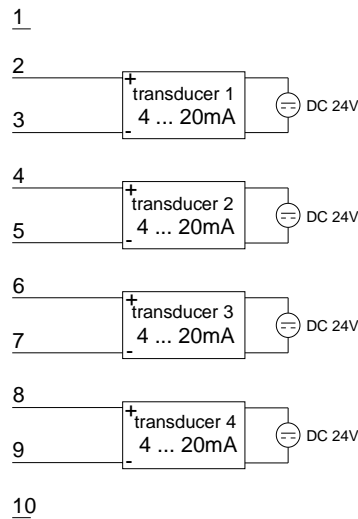
### Connecting transducers

Our analog modules provide a large number of configuration options suitable for 2wire and 4wire transducers. Please remember that transducers require an external power source. You have to connect an external power supply in line with any 2wire transducer. The following diagram explains the connection of 2- and 4wire transducers:

#### 2wire interfacing



#### 4wire interfacing



### Connecting loads and actuators

Due to the fact that actuators also require a source of external power, they may also be connected with 2 or 4 wires. Where control signals are supplied to 2wire actuators a power supply has to be connected in series with the control cable. 4wire actuators need an external power source.



#### Note!

Please ensure that you connect actuators to the correct polarity!  
Unused output terminals must not be connected!

### Parameterization and diagnosis during runtime

By using the SFCs 55, 56 and 57 you may change the parameters of the analog modules during runtime via the CPU 21x.

For diagnosis evaluation during runtime, you may use the SFCs 51 and 59. They allow you to request detailed diagnosis information and to react to it.

## AI 2/AO 2x12Bit, multiin-/output

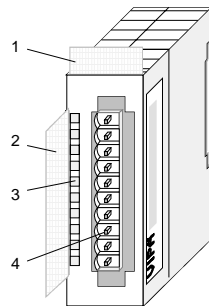
**Order data** AI 2/AO 2x12Bit Multiin-/output VIPA 234-1BD50

**Description** This module has 2 analog inputs and 2 analog outputs that may be configured individually. The module occupies a total of 4Byte of input and 4Byte of output data.

Galvanic isolation between the channels on the module and the backplane bus is provided by means of DC/DC converters and optocouplers. The module requires an external supply of DC 24V.

- Properties**
- 2 inputs and 2 outputs with common ground
  - In-/Outputs with individually configurable functions
  - Suitable for encoder res. actuators with in- res. output ranges of:  $\pm 10V$ , 1...5V, 0...10V,  $\pm 20mA$ , 0...20mA or 4...20mA
  - Diagnostic LED

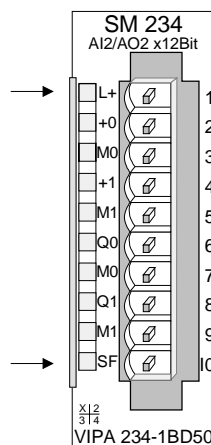
**Construction**



- [1] Label for the name of the module
- [2] Label for the bit address with description
- [3] LED status indicator
- [4] Edge connector

**Status indicator**  
**Pin assignment**

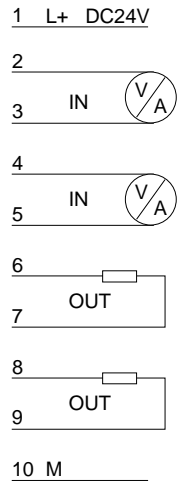
LED	Description
L+	LED (yellow) Supply voltage present
SF	Sum error LED (red) turned on as soon as an channel error is detected res. an entry in the diagnostic bytes happened



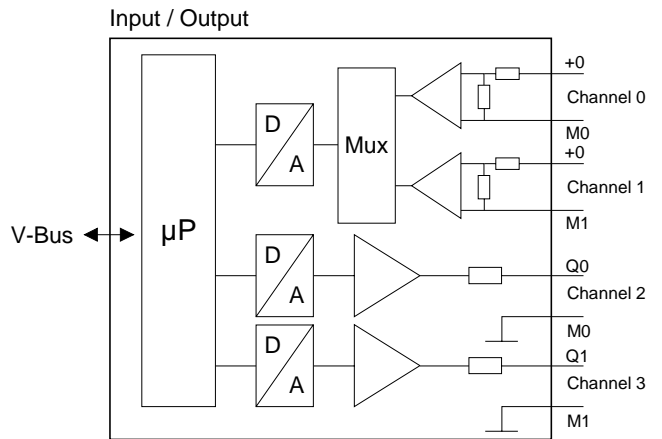
Pin	Assignment
1	DC 24V supply voltage
2	pos. connection E.0
3	Ground Channel 0
4	pos. connection E.1
5	Ground Channel 1
6	pos. connection A.2
7	Ground Channel 2
8	pos. connection A.3
9	Ground Channel 3
10	Supply voltage Ground

**Circuit and schematic diagram**

**Circuit diagram**



**Schematic diagram**



**Attention!**

Temporarily not used inputs have to be connected with the concerning ground at activated channel. When deactivating unused channels by means of FFh, this is not required.



**Note!**

Every channel is individually parameterizable. For the parameterization 12Byte parameterization data are available. They are stored permanently and remain in memory during power-off.

**Function no. assignment** The assignment of a function no. to a certain channel happens during parameterization. The function no. 00h does not influence the function no. stored in the permanent parameterization data.

**Input range (channel 0, ch. 1)** Assigning FFh deactivates the according channel.

**Note!**

When exceeding the overdrive region, the value 7FFFh (32767) is thrown, at underrun of the underdrive region the value is 8000h (-32768).

No.	Function	Measuring range / representation
00h	Does not affect permanently stored configuration data.	
3Bh	Voltage $\pm 10V$ Siemens S5-format	$\pm 12.5V$ / 12.5V = End overdrive region (20480) -10...10V = nominal range (-16384...16384) -12.5V = End underdrive region (-20480) two's complement
2Bh	Voltage $\pm 10V$ Siemens S5-format	$\pm 12.5V$ / 12.5V = End overdrive region (20480) -10...10V = nominal range (-16384...16384) -12.5V = End underdrive region (-20480) Value and sign
72h	Voltage 1...5V Siemens S5-format	0...6V 6V = End overdrive region (20480) 1...5V = nominal range (0...16384) 0V = End underdrive region (-4096) Value and sign
75h	Voltage 0...10V Siemens S5-format	0...12.5V 12.5V = End overdrive region (20480) 0...10V = nominal range (0...16384) no underdrive region available
28h	Voltage $\pm 10V$ Siemens S7-format	$\pm 11.76V$ / 11.76V = End overdrive region (32511) -10...10V = nominal range (-27648...27648) -11.76V = End underdrive region (-32512) two's complement
7Ah	Voltage 1...5V Siemens S7-format	0...5.704V 5.704V = End overdrive region (32511) 1...5V = nominal range (0...27648) 0V = End underdrive region (-6912) two's complement

*continue...*

... continue funktion no. input chanel 0, 1

7Dh	Voltage 0...10V Siemens S7-format	0...11.76V 11.76V = End overdrive region (32511) 0...10V = nominal range (0...27648) no underdrive region available
3Ah	Current $\pm 20$ mA Siemens S5-format	$\pm 25.0$ mA / 25.0mA = End overdrive region (20480) -20...20mA = nominal range (-16384...16384) -25.0mA = End underdrive region (-20480) two's complement
2Fh	Current $\pm 20$ mA Siemens S5-format	$\pm 25.0$ mA / 25.0mA = End overdrive region (20480) -20...20mA = nominal range (-16384...16384) -25.0mA = End underdrive region (-20480) value and sign
2Eh	Current 4...20mA Siemens S5-format	0.8...+24.0mA / 24.0mA = End overdrive region(20480) 4 ... 20mA = nominal range (0...16384) 0.8mA = End underdrive region (-3277) value and sign
76h	Current 0...20mA Siemens S5-format	0...25mA 25mA = End overdrive region (20480) 0...20mA = nominal range (0...16384) no underdrive region available
2Ch	Current $\pm 20$ mA Siemens S7-format	$\pm 23.51$ mA / 23.51mA = End overdrive region (32511) -20...20mA = nominal range (-27648...27648) -23.51mA = End underdrive region (-32512) two's complement
2Dh	Current 4...20mA Siemens S7-format	1.185...+22.81mA / 22.81mA = End overdrive region (32511) 4...20mA = nominal range (0...27648) 1.18mA = End underdrive region (-4864) two's complement
7Eh	Current 0...20mA Siemens S7-format	0...23.52mA 23.52mA = End overdrive region (32511) 0...20mA = nominal range (0...27648) no underdrive region available
FFh	Channel not active (turned off)	



**Note!**

The module is preset to the range " $\pm 10$ V voltage" in S7-format from Siemens.

**Function no.  
assignment  
output range  
Channel 2, Ch. 3**

The assignment of a function no. to a certain channel happens during parameterization. The function no. 00h does not influence the function no. stored in the permanent parameterization data.

Assigning FFh deactivates the according channel.



**Note!**

When exceeding the predefined range, 0V res. 0A is shown as value!

No.	Function	Output or input range
00h	Does not affect permanently stored configuration data	
01h	Voltage $\pm 10V$ Siemens S5-format	$\pm 12.5V$ 12.5V = End overdrive region (20480) -10...10V = nominal range (-16384...16384) -12.5V = End underdrive region (-20480)
02h	Voltage 1...5V Siemens S5-format	0...6V 6V = End overdrive region (20480) 1...5V = nominal range (0...16384) 0V = End underdrive region (-4096)
05h	Voltage 0...10V Siemens S5-format	0...12.5V 12.5V = End overdrive region (20480) 0...10V = nominal range (0...16384) no underdrive region available
09h	Voltage $\pm 10V$ Siemens S7-format	$\pm 11.76V$ 11.76V = End overdrive region (32511) -10V...10V = nominal range (-27648...27648) -11.76 = End underdrive region (-32512) two's complement
0Ah	Voltage 1...5V Siemens S7-format	0...5.704V 5.704V = End overdrive region (32511) 1...5V = nominal range (0...27648) 0V = End underdrive region (-6912) two's complement
0Dh	Voltage 0...10V Siemens S7-format	0...11.76V 11.76V = End overdrive region (32511) 0...10V = nominal range (0...27648) no underdrive region available
03h	Current $\pm 20mA$ Siemens S5-format	$\pm 25.0mA$ 25mA = End overdrive region (20480) -20...20mA = nominal range (-16384...16384) -25mA = End underdrive region (20480)
04h	Current 4...20mA Siemens S5-format	0...24mA 24mA = End overdrive region (20480) 4...20mA = nominal range (0...16384) 0mA = End underdrive region (-4096)

*continue...*



... continue funktion no. output chanel 2, 3

06h	Current 0...20mA Siemens S5-format	0...25mA 25mA = End overdrive region (20480) 0...20mA = nominal range (0...16384) no underdrive region available
0Bh	Current $\pm$ 20mA Siemens S7-format	$\pm$ 23.52mA 23.52mA = End overdrive region (32511) -20...20mA = nominal range (-27648...27648) -23.52mA = End underdrive region (-32512) two's complement
0Ch	Current 4...20mA Siemens S7-format	0...22.81mA 22.81mA = End overdrive region (32511) 4...20mA = nominal range (0...27648) 0mA = End underdrive region (-6912) two's complement
0Eh	Current 0...20mA Siemens S7-format	0...23.52mA 23.52mA = End overdrive region (32511) 0...20mA = nominal range (0...27648) no underdrive region available
FFh	Channel not active (turned off)	

**Numeric notation in Siemens S5 format**

In Siemens S5 format, input data is saved into a word. The word consists of the binary value and the information bits.

*Numeric notation:*

Byte	Bit 7 ... Bit 0
0	Bit 0: overflow bit 0: value within measuring range 1: measuring range exceeded Bit 1: error bit (set by internal errors) Bit 2: activity bit (always 0) Bit 3 ... 7: binary measured value
1	Bit 0 ... 6: binary measured value Bit 7: sign 0 positive 1 negative

**+/- 10V (two's complement)**

Voltage	Decimal	Hex
-10V	-16384	C000
-5V	-8192	E000
0V	0	0000
+5V	+8192	2000
+10V	+16384	4000

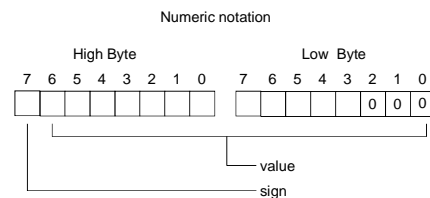
Formulas for the calculation:

$$Value = 16384 \cdot \frac{U}{10}, \quad U = Value \cdot \frac{10}{16384}$$

U: voltage, Value: Decimal value

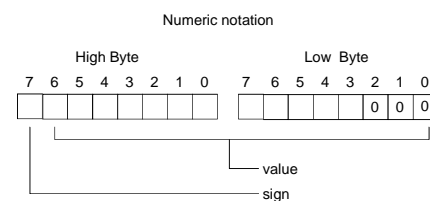
**+/- 10V (value and sign)**

Voltage	Decimal	Hex
-10V	-16384	C000
-5V	-8192	A000
0V	0	0000
+5V	+8192	2000
+10V	+16384	4000



**4 ... 20mA / 1 ... 5V (value and sign)**

Current / Voltage	Decimal	Hex
+4mA / 1V	0	0000
+12mA / 3V	+8192	2000
+20mA / 5V	+16384	4000



**+/- 20mA (two's complement)**

Current	Decimal	Hex
-20mA	-16384	C000
-10mA	-8192	E000
0mA	0	0000
+10mA	+8192	2000
+20mA	+16384	4000

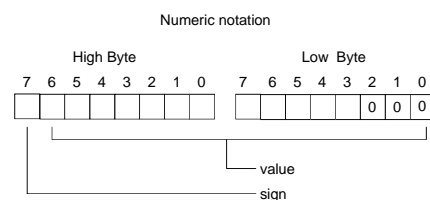
Formula for the calculation:

$$Value = 16384 \cdot \frac{I}{20}, \quad I = Value \cdot \frac{20}{16384}$$

I: Current, Value: Decimal value

**+/- 20mA (value and sign)**

Current	Decimal	Hex
-20mA	-16384	C000
-10mA	-8192	A000
0mA	0	0000
+10mA	+8192	2000
+20mA	+16384	4000



### Numeric notation in Siemens S7 format

The analog values are represented in two's complement format.

*Numeric representation:*

Byte	Bit 7 ... Bit 0
0	Bit 0 ... 7: binary measured vale
1	Bit 0 ... 6: binary measured vale Bit 7: sign 0 positive 1 negative

+/- 10V

Voltage	Decimal	Hex
-10V	-27648	9400
-5V	-13824	CA00
0V	0	0
+5V	13824	3600
+10V	+27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{U}{10}, \quad U = Value \cdot \frac{10}{27648}$$

U: voltage, Value: decimal value

0...10V

Voltage	Decimal	Hex
0V	0	0000
5V	8192	2000
10V	16384	4000

Formulas for the calculation:

$$Value = 27648 \cdot \frac{U}{10}, \quad U = Value \cdot \frac{10}{27648}$$

U: voltage, Value: decimal value

1...5V

Voltage	Decimal	Hex
+1V	0	0
+3V	+13824	3600
+5V	+27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{U-1}{4}, \quad U = Value \cdot \frac{4}{27648} + 1$$

U: voltage, Value: decimal value

+/-4V

Voltage	Decimal	Hex
-4V	-27648	9400
0V	0	0
4V	27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{U}{4}, \quad U = Value \cdot \frac{4}{27648}$$

U: voltage, Value: decimal value

+/-400mV

Voltage	Decimal	Hex
-400mV	-27648	9400
0V	0	0
400mV	27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{U}{400}, \quad U = Value \cdot \frac{400}{27648}$$

U: voltage, Value: decimal value

4....20mA

Current	Decimal	Hex
+4mA	0	0
+12mA	+13824	3600
+20mA	+27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{I-4}{16}, \quad I = Value \cdot \frac{16}{27648} + 4$$

I: current, Value: decimal value

+/- 20mA

Current	Decimal	Hex
-20mA	-27648	9400
-10mA	-13824	CA00
0mA	0	0
+10mA	+13824	3600
+20mA	+27648	6C00

Formulas for the calculation:

$$Value = 27648 \cdot \frac{I}{20}, \quad I = Value \cdot \frac{20}{27648}$$

I: current, Value: decimal value

**Data input/  
data output range**

For in- and output range, 4Byte are available with the following assignment:

*Data input range:*

During the measuring, the measuring values are stored in the data input area.

Byte	Bit 7 ... Bit 0
0	High-Byte channel 0
1	Low-Byte channel 0
2	High-Byte channel 1
3	Low-Byte channel 1

**Note!**

At 3wire res. 4wire measuring, only channel 0 is used.

*Data output range:*

For output of the data you set a value in the data output area.

Byte	Bit 7 ... Bit 0
0	High-Byte channel 2
1	Low-Byte channel 2
2	High-Byte channel 3
3	Low-Byte channel 3

**Parameter data**

12Byte of parameter data are available for the configuration. These parameters are stored in non-volatile memory and are available after the unit has been powered off.

The following table shows the structure of the parameter data:

*Parameter area:*

Byte	Bit 7 ... Bit 0	Default
0	Wire break recognition and diag. alarm: Bit 0: Wire break recognition channel 0 0: deactivated 1: activated Bit 1: Wire break recognition channel 1 0: deactivated 1: activated Bit 2 ... 5: reserved Bit 6: 0: diagnostic alarm inhibited 1: diagnostic alarm enabled Bit 7: reserved	00h
1	reserved Bit 0: reserved Bit 1: reserved Bit 2: CPU-Stop reaction for channel 2 0: Set replacement value channel 2 <sup>)</sup> 1: Store last value channel 2 Bit 3: CPU-Stop reaction for channel 3 0: Set replacement value channel 3 1: Store last value channel 3 Bit 4 ... 7: reserved	00h
2	Function-no. channel 0 (see table input ranges)	28h
3	Function-no. channel 1 (see table input ranges)	28h
4	Function-no. channel 2 (see table input ranges)	09h
5	Function-no. channel 3 (see table input ranges)	09h
6	Option-Byte channel 0	00h
7	Option-Byte channel 1	00h
8	High-Byte replacement value channel 2	00h
9	Low-Byte replacement value channel 2	00h
10	High-Byte replacement value channel 3	00h
11	Low-Byte replacement value channel 3	00h

<sup>)</sup> If you want to get 0A res. 0V as output value at CPU-STOP, you have to set the following replacement values at current output (4...20mA) res. voltage output (1...5V):  
E500h for the S7-format from Siemens and F000h for the S5-format from Siemens.

**Parameters***Wire break recognition*

The wire break recognition is on at the measuring range 4 .. 20mA. Via the Bits 0 and 1 of Byte 0, the wire break recognition is activated.

If the current underruns 0.8mA in 4...20mA current measuring, a wire break is detected. With activated wire break recognition and diagnostic alarm, a diagnostic message is sent to the superordinated system.

*Diagnostic alarm*

With the help of Bit 6 of Byte 0, you may release the diagnostic alarm. In case of an error, the *record set 0* with a size of 4Byte is transferred to the superordinated system.

More detailed information is to find below under "Diagnostic data".

*CPU-Stop reaction and replacement value*

With Bit 2 and 3 of Byte 1 and Byte 8 ... 11 you may set the reaction of the module at CPU-Stop for every output channel.

Via Byte 8 ... 11 you predefine a replacement value for the output channel as soon as the CPU switches to Stop.

By setting Bit 2 res. 3, the last output value remains in the output at CPU-Stop. A reset sets the replacement value.

*Function No.*

Here you set the function no. of your measuring res. output function for every channel. Please see the according table above.

*Option-Byte*

Here you may set the transducer velocity for every input channel. Please regard that a higher transducer velocity causes a lower resolution because of the lower integration time.

The data transfer format remains unchanged. Only the lower Bits (LSBs) are not longer relevant for the analog value.

*Structure Option-Byte:*

Byte	Bit 7 ... Bit 0	Resolution	Default
6 ... 7	Bit 0 ... 3: Velocity per channel		00h
	0000 15 conversions/s	16	
	0001 30 conversions/s	16	
	0010 60 conversions/s	15	
	0011 123 conversions/s	14	
	0100 168 conversions/s	12	
	0101 202 conversions/s	10	
	0110 3,7 conversions/s	16	
0111 7,5 conversions/s	16		
	Bit 4 ... 7: reserved		

**Diagnostic data**

The diagnostic data uses 12Byte and are stored in the record sets 0 and 1 of the system data area.

When you enable the diagnostic alarm in Byte 0 of the parameter area, modules will transfer *record set 0* to the superordinated system when an error is detected.

*Record set 0* has a predefined content and a length of 4Byte. The content of the record set may be read in plain text via the diagnostic window of the CPU.

For extended diagnosis during runtime, you may evaluate the 12Byte wide *record set 1* via the SFCs 51 and 59.

**Evaluate diagnosis**

At present diagnosis, the CPU interrupts the user application and branches into the OB82. This OB gives you detailed diagnostic data via the SFCs 51 and 59 when programmed correctly.

After having processed the OB82, the user application processing is continued. Until leaving the OB82, the data remain consistent.

**Record set 0**

*Byte 0 to 3:*

*Record set 0 (Byte 0 to 3):*

Byte	Bit 7 ... Bit 0	Default
0	Bit 0: Module malfunction Bit 1: reserved Bit 2: External error Bit 3: Channel error present Bit 4: external supply voltage is missing Bit 5 ... 6: reserved Bit 7: Wrong parameters in the module	00h
1	Bit 0 ... 3: Module class 0101 Analog module Bit 4: Channel information present Bit 5 ... 7: reserved	15h
2	reserved	00h
3	reserved	00h

**Record set 1***Byte 0 to 11:*

The *record set 1* contains the 4Byte of record set 0 and additional 8Byte module specific diagnostic data.

The diagnostic bytes have the following assignment:

*Record set 1 (Byte 0 to 11):*

Byte	Bit 7 ... Bit 0	Default
0 ... 3	Content record set 0 (see page before)	-
4	Bit 0 ... 6: Channel type 70h: Digital input 71h: Analog input 72h: Digital output 73h: Analog output 74h: Analog in-/output Bit 7: reserved	74h
5	Bit 0 ... 7: Number of diagnostic bits of the module per channel	08h
6	Bit 0 ... 7: Number of identical channels of a module	04h
7	Bit 0: Channel error Channel 0 Bit 1: Channel error Channel 1 Bit 2: Channel error Channel 2 Bit 3: Channel error Channel 3 Bit 4 ... 7: reserved	00h
8	Bit 0: Wire break Channel 0 Bit 1: Parameterization error Channel 0 Bit 2: Measuring range underflow Channel 0 Bit 3: Measuring range overflow Channel 0 Bit 4 ... 7: reserved	00h
9	Bit 0: Wire break Channel 1 Bit 1: Parameterization error Channel 1 Bit 2: Measuring range underflow Channel 1 Bit 3: Measuring range overflow Channel 1 Bit 4 ... 7: reserved	00h
10	Bit 0: Wire break at current output res. short circuit at voltage output Channel 2 Bit 1: Parameterization error Channel 2 Bit 2 ... 7: reserved	00h
11	Bit 0: Wire break at current output res. short circuit at voltage output Channel 3 Bit 1: Parameterization error Channel 3 Bit 2 ... 7: reserved	00h



**Technical data**

Electrical Data	VIPA 234-1BD50							
Number of in-/outputs	2/2							
Voltage supply	DC5V über Rückwandbus DC24V (20,4 ... 28,8V)							
Current consumption	Rückwandbus: 100mA DC 24V extern: 110mA							
Short circuit current	30mA							
Analog value calculation inputs	Calculation time/Resolution (per channel)							
Parameterized velocity (Hz)	3,7	7,5	15	30	60	123	168	202
Basic calculation time (ms)	268	135	69	35,5	19	10	8	6,75
Additional calculation time (executed once per cycle) (ms)	10	10	10	10	10	10	10	10
Additional calculation time for wire break recognition (ms)	6,5	6,5	6,5	6,5	6,5	6,5	6,5	6,5
Resolution in Bit	16	16	16	16	15	14	12	10
Analog value calculation outputs ±10V, ±20mA 4 ... 20mA, 1 ... 5V 0 ... 10V, 0 ... 20mA	Resolution (incl. overdrive region) 11Bit + Vorzeichen 10Bit 11Bit							
Cycle time	2,5ms							
Settling time - Ohm resistive load - Capacitive load - Inductive load	0,05ms 0,5ms 0,1ms							
Error limits	Measuring range				Tolerance			
- Voltage in-/output	±10V				±0,2%			
	0 ... 10V				±0,4%			
	1 ... 5V				±0,6%			
- Current in-/output	±20mA				±0,3%			
	0 ... 20mA				±0,6%			
	4 ... 20mA				±0,8%			

*continue ...*

... continue technical data

Electrical Data	
Data for choosing an encoder - Voltage input - Current input	100 k $\Omega$ 50 $\Omega$
Data for choosing an actuator - Voltage outputs - Current outputs	Load resistor Ohm resistive load - min. 1 k $\Omega$ Capacitive load - max. 1 $\mu$ F Ohm resistive load - max. 500 $\Omega$ Capacitive load - max. 10 mH
Diagnose alarm	parameterizable
Potential separation	500Vrms (field voltage – backplane bus)
Status monitor	Via LEDs at the front side
Status monitor	Via LEDs at the front side
Input data	4Byte (1 Word per channel)
Output data	4Byte (1 Word per channel)
Parameter data	12Byte
Diagnostic data	12Byte
Measurements and Weight	
Measurements (WxHxD)	25,4x76x76mm
Weight	100g

## Chapter 19 System expansion modules

### Overview

The chapter contains a description of additional components and accessories that are available from VIPA for the System 200V.

A general overview is followed by the description of the bus expansion module that is used to split a single System 200V row over up to 4 rows.

The 4port fast Ethernet mini switch completes the System 200V network technology.

The chapter concludes with the terminal modules. These modules provide connection facilities for signaling cables as well as supply voltages for your System 200V.

Below follows a description of:

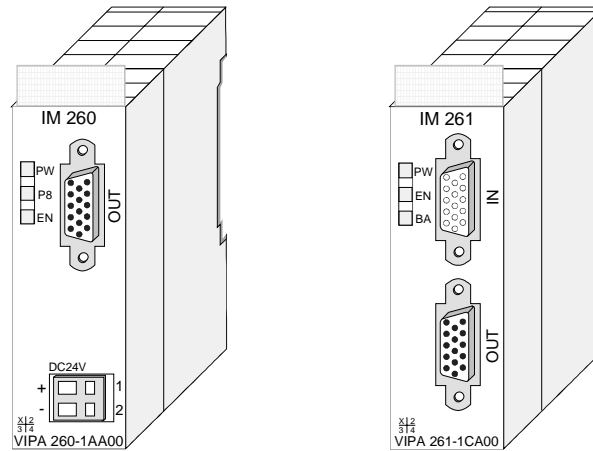
- System overview of additional components
- Bus expansion with IM 260 and IM 261
- 4port mini switch CM 240
- Terminal module CM 201

### Contents

Topic	Page
<b>Chapter 19 System expansion modules</b> .....	<b>19-1</b>
System overview.....	19-2
Bus expansion IM 260, IM 261.....	19-4
4port mini switch CM 240.....	19-7
Terminal module CM 201.....	19-10

## System overview

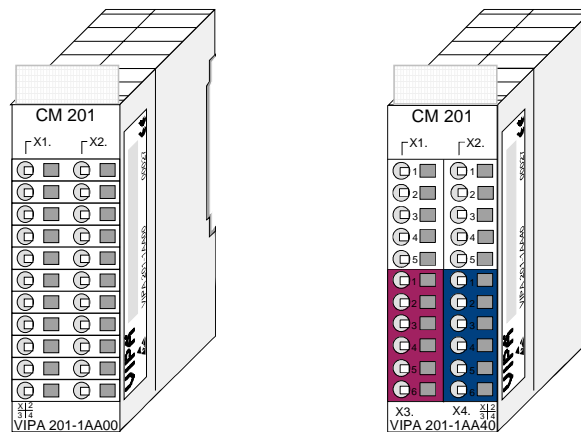
### Bus expansion



### Order data bus expansion

Type	Order number	Description
IM 260	VIPA 260-1AA00	Basic interface row 1
IM 261	VIPA 261-1CA00	Interface for rows 2 ... 4
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

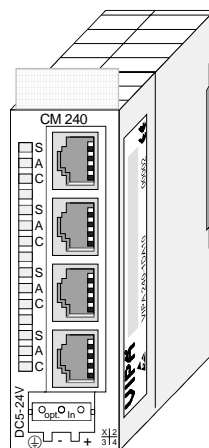
### Terminal module



### Order data terminal modules

Type	Order number	Description
CM 201	VIPA 201-1AA00	Dual terminals gray/gray
CM 201	VIPA 201-1AA10	Dual terminals green-yellow/gray
CM 201	VIPA 201-1AA20	Dual terminals red/blue
CM 201	VIPA 201-1AA40	Quad terminals gray/red/blue

**4port mini switch**



**Order data  
4port mini switch**

Type	Order number	Description
CM 240	VIPA 240-1DA10	4port mini switch
	VIPA 970-0CM00	optional front-facing connector at external power supply DC 5-24V



**Note order number change!**

Because of an order number alteration there are the following changes for the 4port mini switch:

Old Order no.	New Order no.
243-1DA10	240-1DA10

## Bus expansion IM 260, IM 261

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.



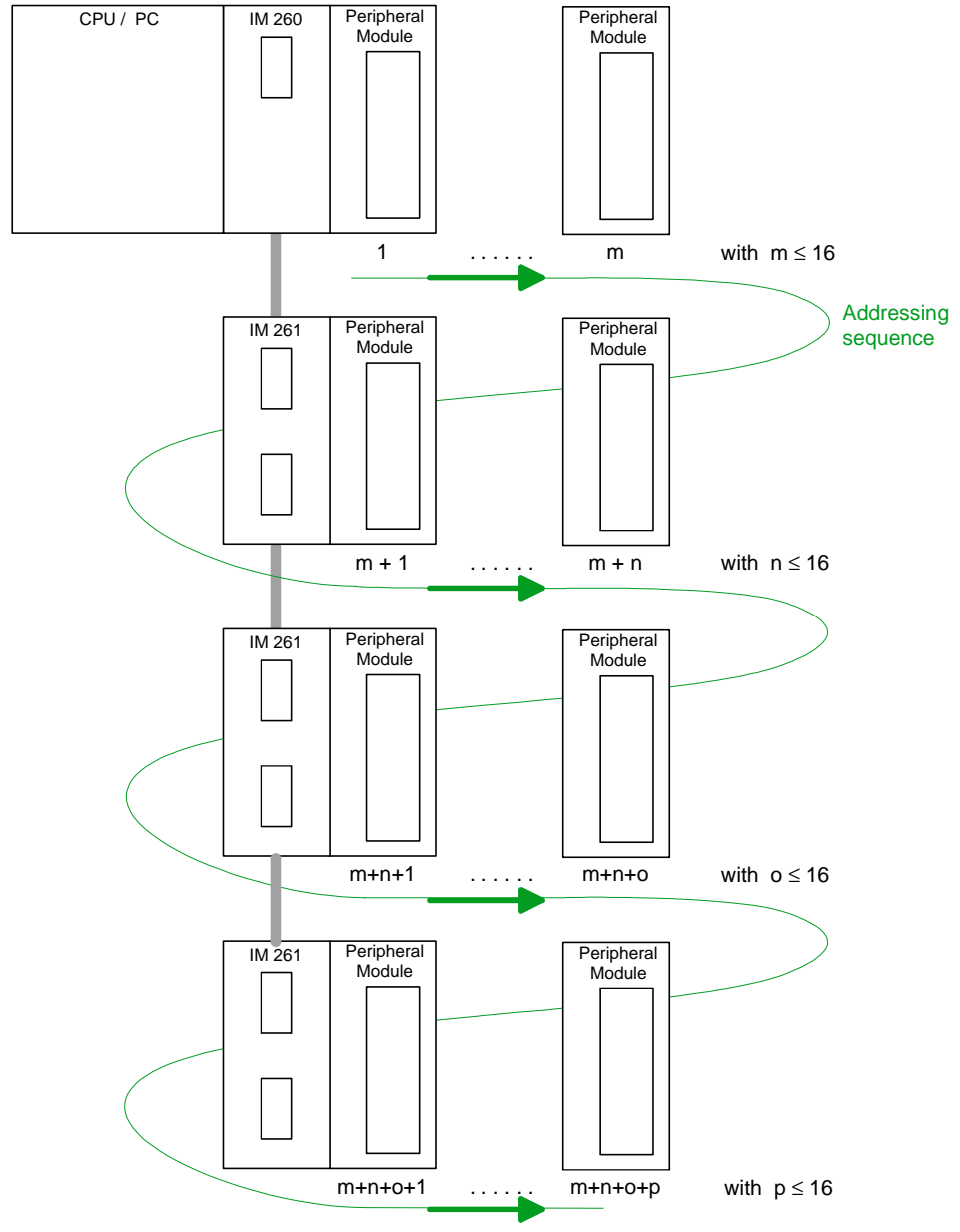
### Please note!

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-CPU's 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!

**Construction**

The following figure shows the construction 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-CPU's are also permitted)!

The bus expansion module is supported as of the following minimum firm-ware revision levels:

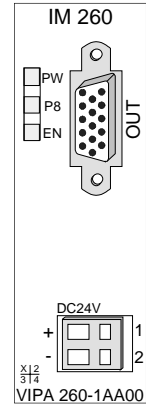
CPU compatible with Siemens STEP<sup>®</sup>5: from Version 2.07

CPU compatible with Siemens STEP<sup>®</sup>7: from Version 1.0

CPU for IEC 61131-3: from Version 1.0

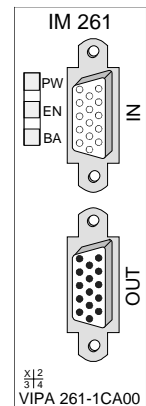
**Status indicator  
Basic interface  
IM 260**

LED	Color	Description
PW	yellow	Supply voltage available
P8	yellow	Supply voltage for subsequent rows is active
EN	yellow	Backplane bus communications active



**Status indicator  
row interface  
IM 261**

LED	Color	Description
PW	yellow	Supply voltage available via IM 260
EN	yellow	Backplane bus communication active
BA	red	Outputs inhibited (BASP) is active



**Order data  
Cables**

Type	Order number	Description
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

**Technical data**

Electrical data	VIPA 260-1AA00	VIPA 261-1CA00
Power supply	24V DC via front	-
Current consumption	1.9A	-
Current consumption backplane bus	30mA	-
Power supply backplane bus at IM 261	-	max. 1.5A per row (max. total 4A)
max. cable distance betw. 1 <sup>st</sup> and last row	2.5m	
Dimensions and weight		
Dimensions (WxHxD) in mm	25.4x76x76	25.4x76x76
Weight	80g	50g



## 4port mini switch CM 240

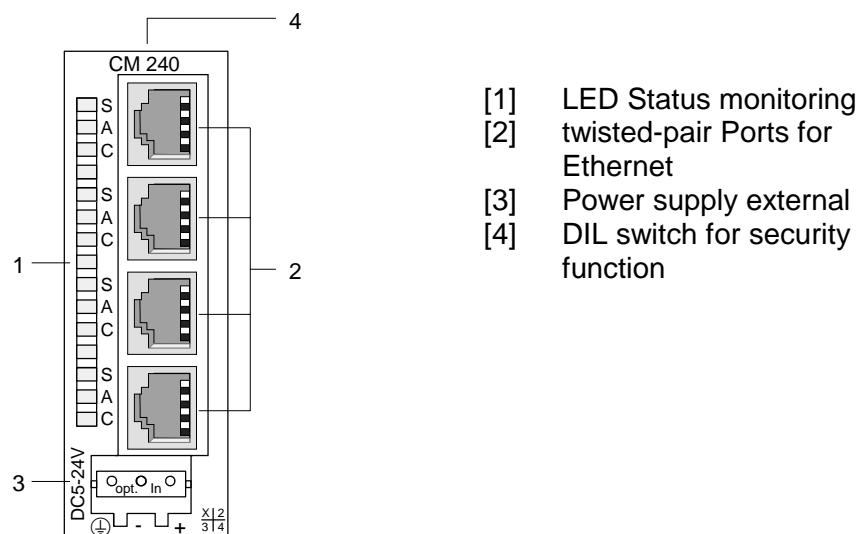
**Order data** 4port mini switch CM 240 VIPA 240-1DA10  
**Attention: the 4port mini switch had the order no. 243-1DA10 before!**

**Overview** The 4port mini switch completes the System 200V network technology. Auto-Negotiation, Speed-Auto-Sensing and the Auto-MDI/MDIX-Crossover for every port enable the module for "plug & play". The module is provided with the needed operating voltage via the backplane bus. Alternatively you may supply the module via the front. The status indication of the 4 ports happens via LEDs on the frontside.

### Properties

- 4 ports for 10 resp. 100MBit/s,
- "plug and play" through Auto-MDI/MDIX-crossover for 100BASE-TX and 10BASE-T,
- Auto-Negotiation and Speed-Auto-Sensing
- for every port automatic switch between 10 and 100MBit/s resp. half- and full-duplex operation
- LEDs for activity, speed and collision
- Supports IEEE 802.3, IEEE 802.3u and IEEE 802.3x
- Extra high performance up to 150m at UTP (unscreened twisted-pair cable)
- Back-pressure-based flow control at half-duplex operation
- Pause-frame-based flow control at full-duplex operation
- Store-and-forward switching mode
- Shared memory based switch

### Front view CM 240



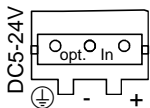
## Components

### LEDs

For every twisted-pair jack there are 3 LEDs at the frontside. The LEDs have the following function:

Name	Color	Function	Description
S	green	Speed	on: 100MBit, off: 10MBit
A	yellow	Activity	on: physically connected, off: no physical connection blinking: shows bus activity
C	yellow	Collision	on: full-duplex operation active, off: half-duplex operation active blinking: Collision detected

### Power supply



The power supply takes place via the backplane bus of the System 200V. You may also deploy the switch as stand-alone device. Here you have to provide it with external DC 5...24V.

The plug for connecting an external power supply is under a flap that you have to break out.

For connecting an external power supply there is a connection jack available from VIPA under the order number VIPA 970-0CM00.



#### Attention!

The power supply has to take place either internal via backplane bus or external.

**A simultaneous supply must be avoided!**

### Twisted-pair ports

The twisted-pair jacks are used to build-up a twisted-pair network in star topology. This allows you to connect up to 4 Ethernet components, where 1 connection has to be deployed as uplink port to the ongoing network.

The uplink port is detected automatically as long as you haven't activated the security function via DIL switch.

**DIL switch for security function**

The activation of the security function prevents the listening between the lower three twisted-pair jacks. Precondition in this case is the uplink via the upper twisted-pair port.

To activate the security function there is a DIL switch on the upper side of the module on the platine with the following function:

- Switch 1            on: Security function deactivated (Default)  
                          off: Security function active
- Switch 2            no function at the moment

**Technical data**

Electrical Data	VIPA 240-1DA10
Number of ports	4
Current consumption via backplane bus	460mA
Power supply (intern)	DC 5V, via backplane bus
Power supply (extern)	optional connection jack VIPA 970-0CM00 DC 5...24V
Status monitor	via LEDs at the frontside
Interface	
	RJ45 twisted-pair, UTP, S/FTP
Dimensions and Weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	50g

## Terminal module CM 201

### 2 x 11 pole

The terminal module is available under order no.: VIPA 201-1AA00.

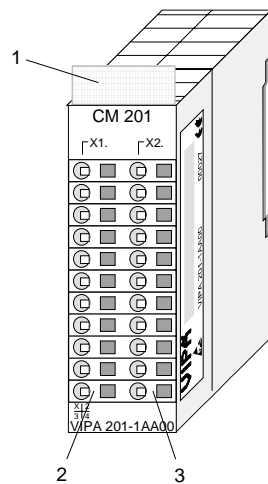
This module is a complementary module providing 2- or 3wire connection facilities. The module is not connected to the system bus.

### Properties

- 2 separate rows of 11 electrically interconnected terminals.
- No connection to the system bus.
- Maximum terminal current 10A.

### Construction and schematic diagram

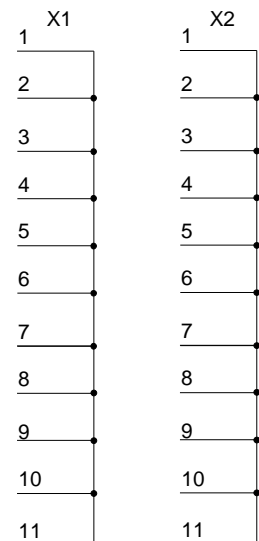
#### Construction



#### Description

- [1] Label
- [2] 1<sup>st</sup> terminal strip
- [3] 2<sup>nd</sup> terminal strip

#### Schematic diagram



### Technical data

Electrical data	VIPA 201-1AA00	VIPA 201-1AA10	VIPA 201-1AA20
Number of rows	2	2	2
Number of terminals per row	11	11	11
Maximum terminal current	10A	10A	10A
Terminal color	gray/gray	green-yellow/gray	red/blue
Dimensions and weight			
Dimensions (WxHxD) in mm	25.4x76x76	25.4x76x76	25.4x76x76
Weight	50g	50g	50g

**2 x 5 pole**  
**2 x 6 pole**

The terminal module has the order no: VIPA 201-1AA40.

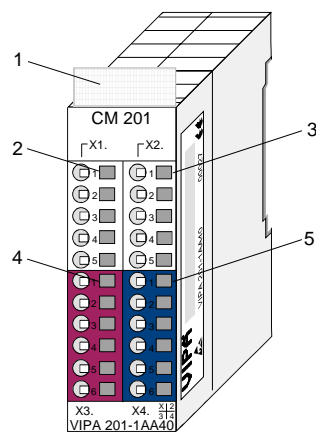
This module is a complementary module providing 2- or 3wire connection facilities. The module is not connected to the system bus.

**Properties**

- 4 separate rows with 2 x 5 and 2 x 6 electrically interconnected terminals.
- No connection to the system bus.
- Maximum terminal current 10A.

**Construction and schematic diagram**

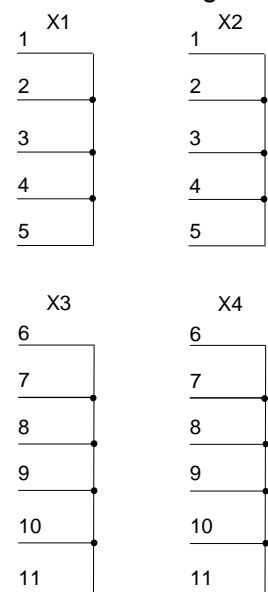
**Construction**



**Description**

- [1] Label
- [2] 1<sup>st</sup> terminal strip
- [3] 2<sup>nd</sup> terminal strip
- [4] 3<sup>rd</sup> terminal strip
- [5] 4<sup>th</sup> terminal strip

**Schematic diagram**



**Technical data**

Electrical data	VIPA 201-1AA40
Number of rows	2
Number of terminals per row	5 + 6
Maximum terminal current	10A
Terminal color	gray/red/blue
Dimensions and weight	
Dimensions (WxHxD) in mm	25.4x76x76
Weight	50g



## Chapter 20 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
- table for the assembly Regarding the current consumption
- EMC guidelines for assembling the System 200V

### Content

Topic	Page
<b>Chapter 20 Assembly and installation guidelines .....</b>	<b>20-1</b>
Overview.....	20-2
Assembly.....	20-4
Wiring.....	20-9
Installation dimensions.....	20-11
Automatic labeling .....	20-12
Installation guidelines.....	20-13

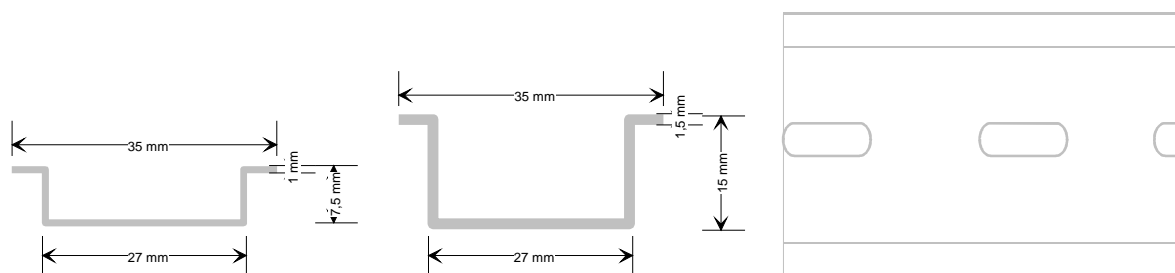
## Overview

### 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

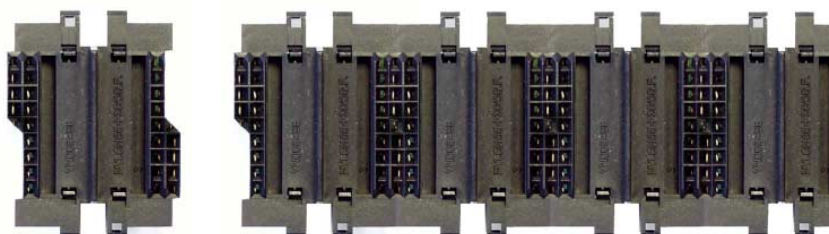
You may use the following standard 35mm profile rail to mount the System 200V modules:



### 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.

### Order data

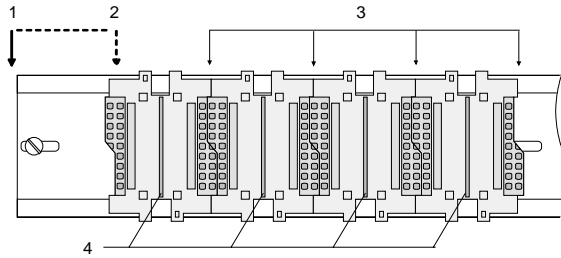
Type	Order number	Description
Bus connector	VIPA 290-0AA10	Bus connector 1tier width
Bus connector	VIPA 290-0AA20	Bus connector 2tier width
Bus connector	VIPA 290-0AA40	Bus connector 4tier width
Bus connector	VIPA 290-0AA80	Bus connector 8tier width
Profile rail	VIPA 290-0AF30	35x15mm, 530mm length 1.5mm gauge, punctured



**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.

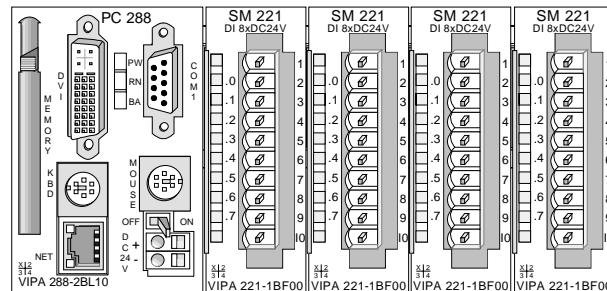
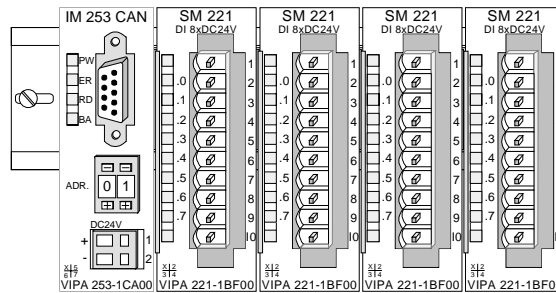


- [1] Header module like PC, CPU, bus coupler, if double width
- [2] Main module (single width)
- [3] Peripheral module
- [4] Guide rails

**Note**

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

Further below you'll find an overview about the current consumptions.

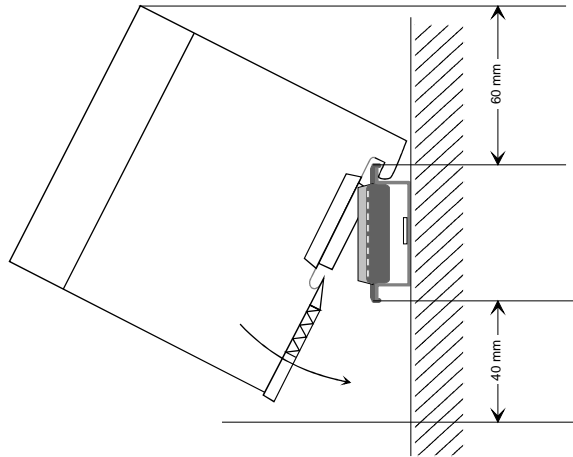


## Assembly

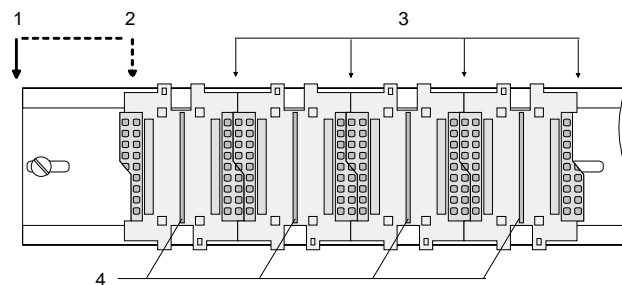


### Please follow these rules during the assembly!

- Turn off the power supply before you insert or remove any modules!
- Make sure that a clearance of at least 60mm exists above and 40mm below 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.

**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. The following table shall help you:

*Current consumption of the peripheral modules*

Order no.	Module	Current consumption
<b>Digital input modules SM 221</b>		
VIPA 221-1BF00	DI 8xDC24V	20mA
VIPA 221-1BF10	DI 8xDC24V (0,2ms)	20mA
VIPA 221-1BF20	DI 8xDC24V Alarm	140mA
VIPA 221-1BF50	DI 8xDC24V NPN	20mA
VIPA 221-1FD00	DI 4xAC/DC 90...230V	80mA
VIPA 221-1FF20	DI 8xAC/DC 60...230V	80mA
VIPA 221-1FF30	DI 8xAC/DC 24...60V	80mA
VIPA 221-1FF40	DI 8xAC 230V, 20mA	80mA
VIPA 221-1FF50	DI 8xAC/DC 180...265V	30mA
VIPA 221-1BH00	DI 16xDC 24V, UB4x	20mA
VIPA 221-1BH10	DI 16xDC 24V	30mA
VIPA 221-1BH20	DI 16xDC 24V, 1 Counter, 100kHz	100mA
VIPA 221-1BH50	DI 16xDC 24V, NPN	20mA
VIPA 221-2BL10	DI 32xDC 24V	50mA
<b>Digital output modules SM 222</b>		
VIPA 222-1BF00	DO 8xDC 24V, 1A	50mA
VIPA 222-1BF10	DO 8xDC 24V, 2A	50mA
VIPA 222-1BF20	DO 8xDC 24V, 2A, floating	50mA
VIPA 222-1BH00	DO 16xDC 24V, 0,5A, UB4x	100mA
VIPA 222-1BH10	DO 16xDC 24V, 1A	80mA
VIPA 222-1BH20	DO 16xDC 24V, 2A	100mA
VIPA 222-1BH50	DO 16xDC 24V, 0,5A, NPN	100mA
VIPA 222-2BL10	DO 32xDC 24V, 1A	140mA
VIPA 222-1HF00	DO 8xrelay	270mA
VIPA 222-1HD10	DO 4xrelay, floating	150mA
VIPA 222-1HD20	DO 4xrelay, floating, bistable	40mA
VIPA 222-1FF00	DO 8xrelay, Solid State	140mA
VIPA 222-1FD10	DO 4xrelay, Solid State	100mA

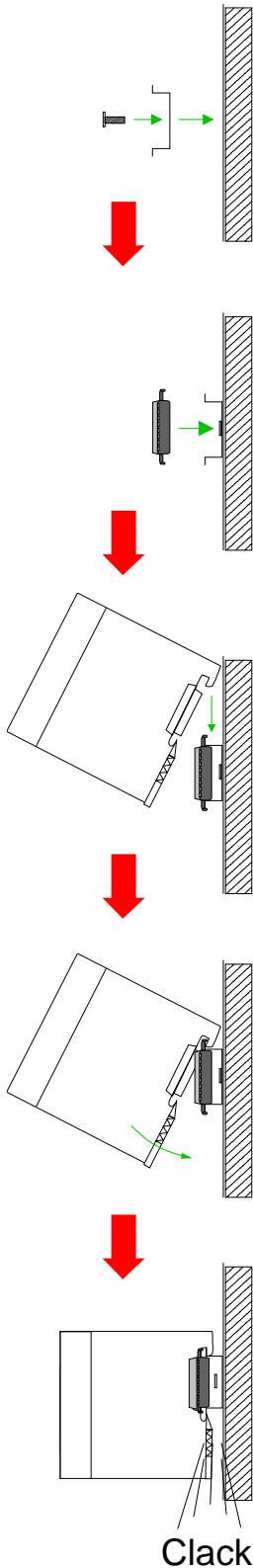
*continue ...*

... continue

Order no.	Module	Current consumption
<b>Digital in-/output modules SM 223</b>		
VIPA 223-1BF00	DIO 8xDC 24V, 1A	65mA
VIPA 223-2BL10	DI 16xDC 24V, DO 16xDC 24V, 1A	100mA
<b>Analog input modules SM 231</b>		
VIPA 231-1BD52	AI 4x16Bit, Multiinput	240mA
VIPA 231-1BD60	AI 4x12Bit, 4...20mA	280mA
VIPA 231-1BF00	AI 8x16Bit, (2L) 4x16Bit (4L)	280mA
VIPA 231-1FD00	AI 4x16Bit f, U/I	300mA
<b>Analog output modules SM 232</b>		
VIPA 232-1BD50	AO 4x12Bit	30mA
VIPA 232-1BD60	AO 4x12Bit f	50mA
<b>Analog in-/output modules SM 234</b>		
VIPA 234-1BD50	AI/AO 2x12Bit, Multiin-/output	110mA
<b>Additional modules</b>		
VIPA 208-1CA01	IM 208CAN - CANopen master	250mA
VIPA 208-1DP01	IM 208DP for CPU 21x	450mA
VIPA 208-2DP10	IM 208DPO	500mA
VIPA 240-1BA00	CP 240, RS232C	200mA
VIPA 240-1BA10	CP 240, RS232C, Modbus	200mA
VIPA 240-1CA00	CP 240, RS422/485	200mA
VIPA 240-1CA10	CP 240, RS422/485, Modbus	200mA
VIPA 240-1DA10	CM 240, 4port Mini-Switch	460mA
VIPA 250-1BA00	Counter 2x32Bit FM 250	80mA
VIPA 250-1BS00	SSI-Modul FM250S	200mA
VIPA 253-1BA00	MotionControl Stepper FM 253	200mA
VIPA 254-1BA00	MotionControl Servo FM 254	295mA

**Assembly procedure**

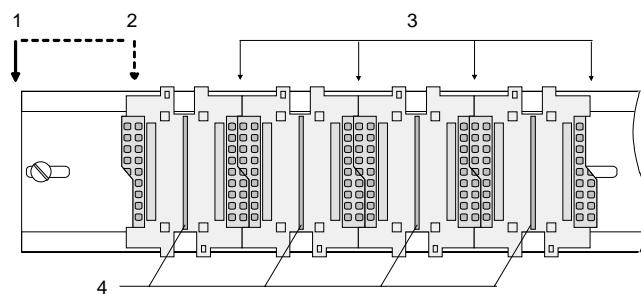
The following sequence represents the assembly procedure as viewed from the side.



- Install the profile rail. Please ensure that you leave a module installation clearance of at least 60mm above the rail and at least 40mm below the 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.

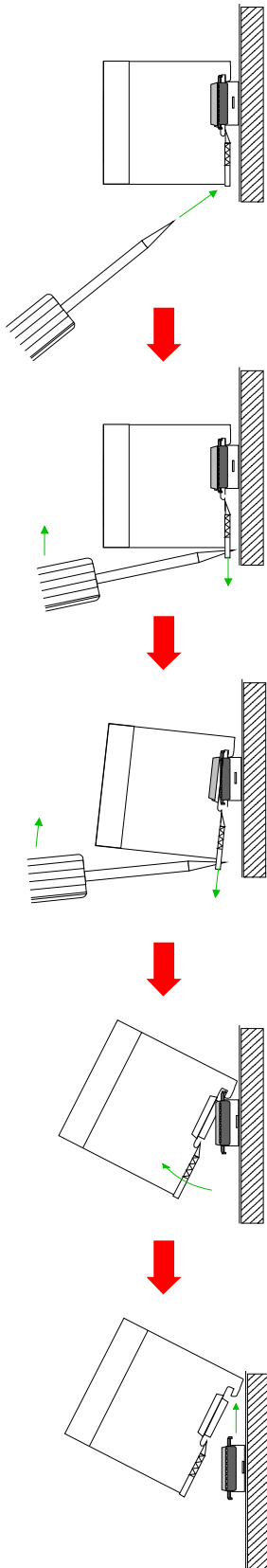


**Attention!**

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.

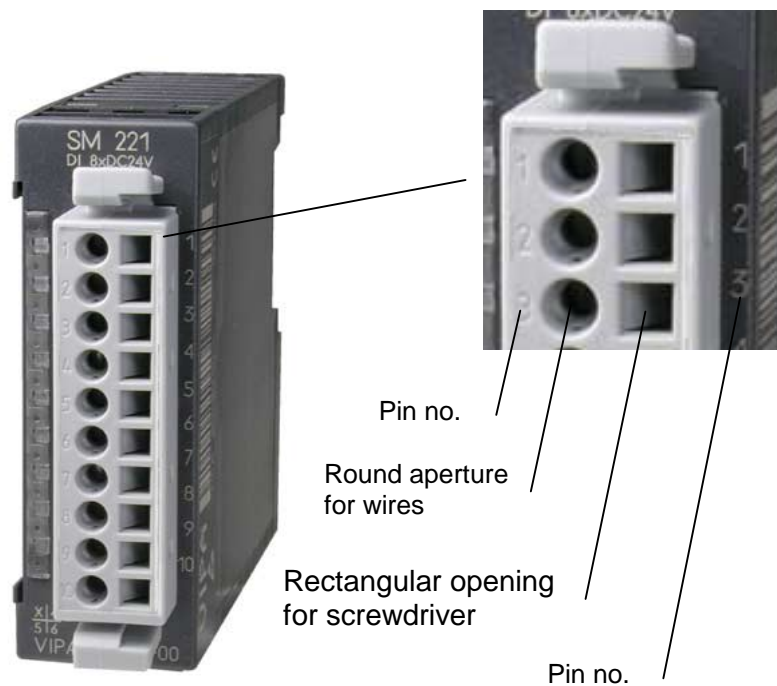
The spring-clip connector technology simplifies the wiring requirements for signaling and power cables.

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<sup>2</sup> up to 2.5mm<sup>2</sup> for 18pole connectors.

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

Folgende Abbildung zeigt ein Modul mit einem 10poligen Steckverbinder.

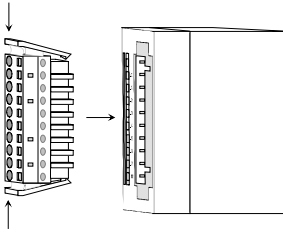


### Note!

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

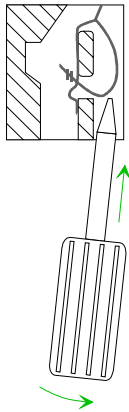
Make sure that you only insert the screwdriver into the square hole of the connector!

## Wiring procedure

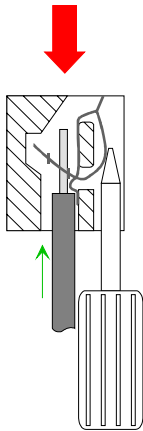


- Install the connector on the module until it locks with an audible click. For this purpose you press the two clips together as shown. The connector is now in a permanent position and can easily be wired.

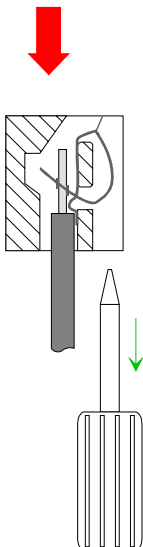
The following section shows the wiring procedure from above.



- Insert a screwdriver at an angle 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  $0.08\text{mm}^2$  to  $2.5\text{mm}^2$  ( $1.5\text{mm}^2$  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)



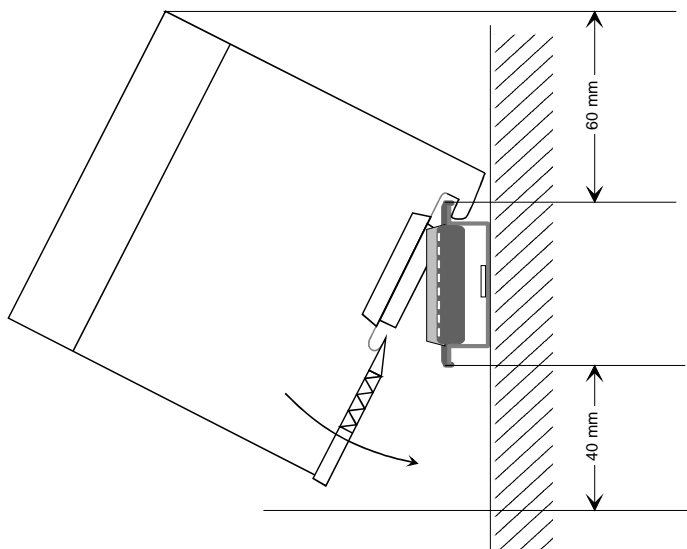
## Installation dimensions

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

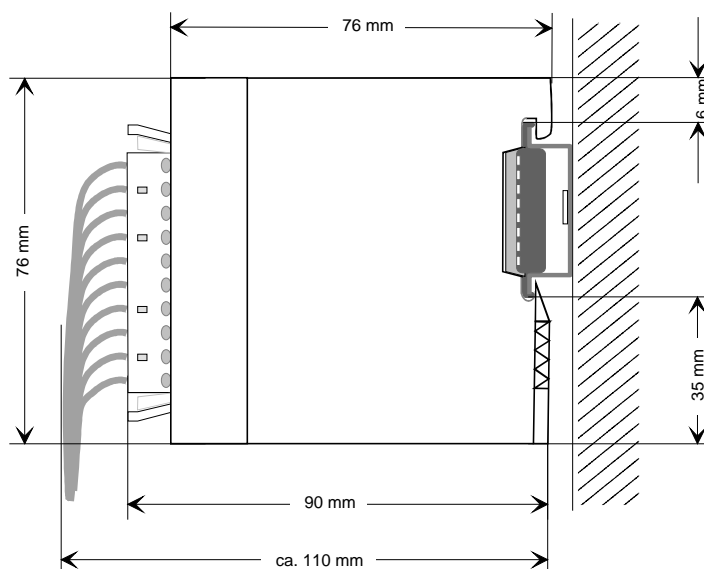
**Dimensions**  
**Basic enclosure**

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

**Installation dimensions**



**Installed and wired dimensions**



## Automatic labeling

**General** The System 200V peripheral modules have a label that can be used for automatic labeling.

**Labeling by means of WinNCS** You may use the labeling components of WinNCS to print the required labels. WinNCS is the VIPA configuration tool that has a special label printing feature for the System 200V labels.

**Labeling by means of WinLP** VIPA supplies the label printing package WinLP to create the labels for a Siemens S7 project. This generates labels for the System 200V from the Siemens S7-cfg file.

### Order data

Type	Order number	Description
WinNCS	VIPA SW-WinNCS	Configuration and labeling software for the System 200V under Win9x/NT incl. WinLP
Demo Software	VIPA SW-Tool Demo	Demo versions of all VIPA tools incl. full labeling functions under WinNCS
Block of labels	VIPA 292-1XY00	10 label cards with covers
Sheets of labels	VIPA 292-1XY10	10 perforated sheets of labels 8 labels each

## Installation guidelines

### General

The installation guidelines contain information on the proper assembly of System 200V systems. 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 not 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  $\mu$ A 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



## Appendix

### A Index

- 2  
 20mA interface.....9-11; 9-15  
 2wire interfacing..... 16-4
- 3  
 3964(R) .....9-5; 9-24  
   with RK512 .....9-6; 9-24
- 4  
 4wire interfacing..... 16-4
- A**  
 active low input ..... 13-10; 13-36  
 Address selector  
   CAN-Bus..... 4-17  
   DeviceNet ..... 5-6  
   Profibus DPR .....2-38; 2-42  
   Profibus-DP ..... 2-35  
   SERCOS..... 6-6  
 AI 2/AO 2x12Bit ..... 18-4  
 Alarm input..... 13-8  
 Analog input..... 16-1  
 Analog input/output ..... 18-1  
 Analog output..... 17-1  
 Analog signal cables ..... 16-4  
 Application layer..... 1-10  
 ASCII.....9-4; 9-22  
 Assembly ..... 20-1; 20-4  
 Auto Reload ..... 10-13  
 Automatic labeling..... 20-12
- B**  
 Backplane bus connector ..... 20-2  
 Basic CMOS Configuration..... 8-15  
 BASP signal ..... 10-7  
 Baudrate  
   CAN-Bus..... 4-17  
   CP ..... 9-26  
   DeviceNet ..... 5-9  
   SSI-Interface..... 10-6  
 BCC-Byte ..... 9-7
- BIOS setup ..... 8-13  
 Bit communication layer ..... 1-9  
 Boot drive..... 8-18  
 Bus connector..... 20-2  
 Bus expansion ..... 19-4  
 BWZ ..... 9-29
- C**  
 CAN identifier ..... 4-13; 4-21  
 CAN-Bus..... 4-1  
   Address selector..... 4-17  
   Baudrate ..... 4-17  
   Bus access ..... 4-4  
   CANopen ..... 4-3  
   Communication types ..... 4-22  
   Coupler IM 253CAN ..... 4-5  
   Coupler IM 253CAN DO ..... 4-9  
   fast introduction ..... 4-13  
   Message structure..... 4-18  
   Object directory ..... 4-26  
   Slave..... 4-5  
   Slave with DO 24xDC 24V ..... 4-9  
   Wiring ..... 4-7; 4-12  
 Centralized system ..... 1-5  
 Character frame ..... 9-3  
 CLK..... 10-13  
 COB-ID ..... 4-21  
 Coding SSI-Interface ..... 10-7  
 Com1 Mode ..... 8-19  
 Commissioning  
   DeviceNet ..... 5-8  
   Interbus..... 3-15  
   Profibus ..... 2-64  
 Communication layers ..... 1-11  
 Communication processor..... 9-1  
 CompactFlash ..... 8-5; 8-9; 8-15  
 Compare Load..... 10-13  
 Components System 200V..... 1-5  
 Connecting actuators..... 17-3  
 Connecting transducers..... 16-4  
 Controlbyte ..... 10-11

- Control-interface ..... 11-25
- Core cross-section..... 1-7
- Counter modes ... 10-12; 10-14; 10-50
  - (0) 32Bit counter ..... 10-14; 10-50
  - (1) Encoder 1 edge .... 10-16; 10-52
  - (12,13) 32 bit counter..... 10-25
  - (14,15) 32 bit counter..... 10-26
  - (16) Frequency ..... 10-28
  - (17) Period ..... 10-30
  - (18) Frequency ..... 10-32
  - (19) Period ..... 10-34
  - (20) Pulse ..... 10-36
  - (21) Pulse width ..... 10-38
  - (22) Pulse width ..... 10-40
  - (23) One Shot ..... 10-42
  - (24) One Shot ..... 10-44
  - (25) One Shot ..... 10-46
  - (26) One Shot ..... 10-48
  - (3) Encoder 2 edges .. 10-18; 10-54
  - (5) Encoder 4 edges .. 10-20; 10-56
  - (6) Pulse-width ..... 10-22
  - (8...11) Counter, 2 inputs ..... 10-24
- Counter module FM 250 ..... 10-9
  - Counter modes ..... 10-12
  - Input/output data ..... 10-11
  - remanent..... 10-11
- Counter modules ..... 10-1
- CP
  - Communication..... 9-45
  - Configuration ..... 9-22
  - Diagnostics ..... 9-13
  - Handler blocks
    - for CPU 21x..... 9-61
    - for CPU 24x..... 9-46
  - Parameter description ..... 9-26
  - Protocols..... 9-3
  - Software handshake ..... 9-31
  - Wiring ..... 9-14; 9-19
    - with 20mA/RS232C ..... 9-11
    - with RS422/RS485 ..... 9-16
- current consumption ..... 20-5; 20-6
  
- D**
  - Data bits..... 9-27
  - Data consistency
    - Interbus..... 3-14
    - Profibus-DP ..... 2-8
  - DBL..... 9-29
  
- Decentralized system ..... 1-5
- De-insulation length ..... 2-56
- Device Details..... 5-10
- DeviceNet ..... 5-1
  - Address ..... 5-9
  - Addressing..... 5-4
  - baudrate ..... 5-9
  - Bus access ..... 5-4
  - Configuration ..... 5-8
  - Example ..... 5-14
  - Interfacing..... 5-6
    - Manager ..... 5-8
  - Profibus interface ..... 5-22
    - Scanner ..... 5-16
  - Test ..... 5-10
- Device-related diagnostics ..... 2-48
- DI16/1C
  - in-output..... 13-29
  - modes..... 13-31; 13-32
  - overview ..... 13-27
- Diagnostic cable ..... 9-13
- Diagnostic functions
  - AI 4x16Bit..... 16-15
  - AI 4x16Bit f ..... 16-28
  - AI 8x16Bit f ..... 16-39
  - Analog input/output..... 18-15
  - AO 4x12Bit ..... 17-10
  - AO 4x12Bit f ..... 17-18
  - CP..... 9-13
  - DeviceNet..... 5-17
  - Ethernet coupler ..... 7-16
  - Profibus-DP ..... 2-47
- Digital input ..... 13-1
- Digital input/output ..... 15-1
- Digital output..... 14-1
- DIN rail..... 20-2
- DIR ..... 10-13
- Disturbances..... 20-13
- DLE-character ..... 9-7
- DP cycle..... 2-7
- DVI socket ..... 8-7
  
- E**
  - EDS file..... 5-4
  - EMC..... 20-13
    - Basic rules ..... 20-14
  - Emergency Object ..... 4-15; 4-67
  - Encoder ..... 11-26



- Encoder interface..... 11-25
- End flag..... 9-28
- Environmental conditions..... 1-7
- Error messages
  - AI 2/AO 2x12Bit ..... 18-15
  - AI 4x16Bit..... 16-15
  - AI 4x16Bit f ..... 16-28
  - AI 8x16Bit f ..... 16-39
  - AO 4x12Bit..... 17-10
  - AO 4x12Bit f..... 17-18
  - CAN-Bus..... 4-6; 4-10; 4-15; 4-67
  - CP..... 9-48
  - DeviceNet ..... 5-17
  - Interbus..... 3-16
  - Profibus-DP ..... 2-50
- Ethernet ..... 7-3
- Ethernet coupler..... 5; 7-1
  - Access ..... 7-11
  - Addressing..... 7-14
  - Delivery default ..... 7-9
  - Diagnosis ..... 7-16
  - Ethernet connection..... 7-10
  - Function codes ..... 7-21
  - http web server ..... 7-12; 7-16
  - Include GSD ..... 7-15
  - Modbus range 0x...4x ..... 7-21
  - ModbusTCP..... 7-6; 7-20
  - Network planning..... 7-7
  - OPC server ..... 7-12
  - ORG format ..... 7-25
  - PLC header..... 7-26
  - Project engineering..... 7-11; 7-15
  - Protocols..... 7-4
  - Siemens S5 Header7-6; 7-13; 7-25
  - Socket programming ..... 7-27
  - Structure ..... 7-9
  - Technical data ..... 7-28
- F**
  - FETCH (FB20) ..... 9-51
  - FETCH\_RK512 (FC 2) ..... 9-66
  - Flash Memory Card ..... 2-24
  - Flow control..... 9-27
  - FO link ..... 2-57
  - FO link interface..... 2-24
  - Fref ..... 10-13
  - Frequency calculation ..... 10-29
  - Function no.
- AI 2/AO 2x12Bit..... 18-6
- AI 4x16Bit ..... 16-7; 16-24
- AI 4x16Bit f ..... 16-24
- AI 8x16Bit ..... 16-35
- AO 4x12Bit ..... 17-5
- AO 4x12Bit f ..... 17-14
- G**
  - Gate..... 10-13
  - Gray Code ..... 10-7
  - Green Cable
    - Hints ..... 1-3
  - Group 2 only Device ..... 5-5
  - GSD-file ..... 2-9
- H**
  - Heartbeat..... 4-15; 4-69
  - Hold function..... 10-7
  - Hold input..... 10-3
  - Hub ..... 7-3
  - Hysterese..... 13-18
- I**
  - ID code ..... 3-13
  - ID length ..... 3-13
  - ID register ..... 3-3
  - Installation dimensions ..... 20-11
  - Installation guidelines ..... 20-13
  - Interbus..... 3-1
    - Commissioning..... 3-15
    - Configuration of the master .... 3-17
    - Coupler IM 253IBS ..... 3-7
    - Data consistency ..... 3-14
    - Data transfer..... 3-5
    - Master..... 3-3
    - Modes of operation..... 3-4
    - Process data allocation ..... 3-11
    - Wiring ..... 3-10
  - Introduction..... 1-1
  - ISO/OSI reference model ..... 1-8
- M**
  - Measurement data acquisition
    - AI 4x16Bit ..... 16-13
    - AI 4x16Bit f ..... 16-26
    - AI 8x16Bit ..... 16-36
  - Measurement gate..... 10-13
  - Measuring data

- AI 2/AO 2x12Bit ..... 18-12
- Message structure
  - CAN-Bus..... 4-18
- min\_slave\_interval ..... 2-8
- Mini switch ..... 19-7
- MMC ..... 2-11
- Modbus
  - Function codes ..... 7-21; 7-22
  - Include GSD ..... 9-33
  - Include library..... 9-33
  - Parameter ..... 9-25
- Modes of operation
  - Profibus master ..... 2-12
- Module-ID ..... 4-17
- MotionControl Modules ..... 11-1
- MotionControl Servo ..... 11-23
  - Cabling..... 11-27
  - Components ..... 11-24
  - Data transfer to CPU ..... 11-37
  - Data transfer to FM 254..... 11-30
  - Drive ..... 11-26
  - Encoder connection ..... 11-26
  - Field identifier ..... 11-30
  - Operating modes ..... 11-11; 11-31
  - Outputs ..... 11-26
  - Parameterization..... 11-29
  - Parameters ..... 11-28
- MotionControl Stepper ..... 11-3
  - Cabling..... 11-7
  - Components ..... 11-4
  - Data transfer to CPU ..... 11-15
  - Data transfer to FM 253..... 11-8
  - Inputs ..... 11-6
  - Outputs ..... 11-6
  - Standard function modules... 11-21
- Multiinput..... 16-5
- Multi-Output ..... 17-4
- N**
- Network layer ..... 1-9
- NMT ..... 4-68
- Node Guarding..... 4-14; 4-69
- NPN ..... 14-16
- O**
- Operating modes ..... 3-4
  - Profibus master ..... 2-12
- Optical Profibus ..... 2-62
- Optical waveguide ..... 2-9
- Option byte
  - AI 4x16Bit ..... 16-14
  - AI 8x16Bit ..... 16-38
- Opto-electrical Profibus ..... 2-63
- Overview System 200V ..... 1-4
- P**
- Parameterization
  - AI 2/AO 2x12Bit ..... 18-5; 18-13
  - AI 4x16Bit ..... 16-6; 16-13
  - AI 4x16Bit f ..... 16-26
  - AI 8x16Bit ..... 16-37
  - AO 4x12Bit ..... 17-9
  - AO 4x12Bit f ..... 17-17
  - Counter module FM 250..... 10-11
  - CP..... 9-22
  - DeviceNet ..... 5-11
  - MotionControl Servo ..... 11-29
  - MotionControl Stepper..... 11-9
  - Profibus-DP slave..... 2-46
  - SERCOS ..... 6-8
  - SSI-Interface ..... 10-6
- Parity..... 9-27
- Passive operation ..... 9-6
- PC 288 - CPU ..... 8-1
  - Configuration ..... 8-3
  - Process image..... 8-10
  - Register ..... 8-21
- PDO ..... 4-20
  - linking ..... 4-22
  - transmission type..... 4-23
- Period calculation ..... 10-31
- Peripheral modules..... 1-5
- Poll only Device ..... 5-5
- Power supplies
  - Installation ..... 12-8
  - PS 207/2, 2A ..... 12-4
  - PS 207/2CM, 2A..... 12-6
  - Safety precautions..... 12-2
  - Wiring ..... 12-9
- Presentation layer ..... 1-10
- Priority..... 9-29
- Procedures ..... 9-5
- Process image
  - Counter module ..... 10-11
  - DeviceNet ..... 5-16
  - Interbus..... 3-11

- PC 288 - CPU ..... 8-10
  - SSI-module ..... 10-8
  - Profibus-DP ..... 2-1
    - Addressing ..... 2-9
    - Commissioning ..... 2-64
    - Connectors ..... 2-56
    - Data transfer ..... 2-7
    - Data transfer protocol ..... 2-6
    - DeviceNet interface ..... 5-22
    - Examples ..... 2-66
    - Installation guidelines ..... 2-54
    - Master ..... 2-5
    - Master with FO ..... 2-23
      - Deployment at CPU 21x..... 2-26
      - Flash Memory Card..... 2-24
      - Operating modes..... 2-25
      - Project engineering ..... 2-27
    - Master with RS485 ..... 2-10
      - Deployment at CPU 21x..... 2-13
      - Operating modes..... 2-12
      - Overall-Reset ..... 2-22
      - Project engineering ..... 2-14
  - Multi master system..... 2-61
  - Networks..... 2-60
  - Redundancy state ..... 2-53
  - Slave ..... 2-5; 2-33
    - Block diagram ..... 2-43
    - Data communication ..... 2-6
    - Diagnostic functions ..... 2-47
    - IM 308-B from Siemens ..... 2-45
    - Include GSD ..... 2-44
    - Parameter ..... 2-46
    - Project engineering ..... 2-44
    - redundant ..... 2-36
    - S7-400 from Siemens ..... 2-45
    - with 24xDO ..... 2-39
    - Token-passing procedure..... 2-6
  - Protocols ..... 9-4
  - PS2 sockets..... 8-6
  - Pulse ..... 10-13
- Q**
- QVZ..... 9-29
- R**
- RECEIVE (FB4)..... 9-49
  - Receive buffers ..... 9-28
  - RECEIVE\_ASCII\_3964 (FC 1)..... 9-64
  - Reference frequency ..... 10-13
  - Register PC 288 - CPU ..... 8-21
  - Relay output..... 14-20; 14-22
    - bistable ..... 14-24
  - Removal ..... 20-8
  - RES ..... 10-13
  - RS232/RS422 switch..... 8-22
  - RS232C interface ..... 9-11
  - RS232C interface ..... 9-14
  - RS422 interface..... 9-16; 9-19
  - RS485 interface..... 9-16; 9-20
    - Profibus ..... 2-11
- S**
- S/R\_ALL (FB23) ..... 9-55
  - S/R\_ALL\_RK512 (FC 4) ..... 9-72
  - Safety Information ..... 1-2
  - Screening of cables ..... 20-15
  - SDO ..... 4-24
  - SDO error codes..... 4-25
  - Security layer ..... 1-9
  - Segment length under Profibus .. 2-54
  - SEND (FB22)..... 9-53
  - SEND (FB3)..... 9-47
  - SEND\_ASCII\_STX\_3964 (FC 0) 9-62
  - SEND\_RK512 (FC 3) ..... 9-69
  - SERCOS ..... 6-1
    - Connection ..... 6-6
  - Session layer ..... 1-10
  - set\_address\_table ..... 8-11
  - Shift register ..... 3-3
  - Siemens S5 format16-11; 16-18; 16-21
  - Siemens S7-format..... 16-12; 16-25
  - SIP tool ..... 2-20
  - Software handshake..... 9-31
  - Solid State ..... 14-26
  - SSI-Interface FM 250S ..... 10-2
    - Configuration ..... 10-6
    - Input/Output data..... 10-8
    - Line distances..... 10-5
  - Standard diagnostic data..... 2-48
  - start flags ..... 9-28
  - Static level ..... 9-21
  - Status byte..... 10-11
  - Stop bits..... 9-27
  - STX repetitions ..... 9-29
  - STX/ETX..... 9-4; 9-23
  - Switch ..... 7-3

- DIL switch ..... 19-9
  - SYNCHRON\_RESET (FC 9) ..... 9-74
  - SYNCRON (FB25) ..... 9-57
  - System expansion modules ..... 19-1
- T**
- Terminal module ..... 19-10
  - Termination ..... 2-56
  - Timeout times ..... 9-6
  - TMO ..... 9-28
  - Transport layer ..... 1-9
  - Twisted Pair ..... 7-3
    - Restrictions ..... 7-8
  - Twisted-Pair-connection ..... 8-6
- U**
- UB4x ..... 13-23; 14-11
  - User acknowledgement ..... 9-29
- V**
- V-bus cycle ..... 2-7
  - Vbus\_api.c ..... 8-10
  - Vbus\_api.h ..... 8-10
  - Vbus\_read\_pword ..... 8-11
  - Vbus\_set\_param ..... 8-12
  - Vbus\_write\_pword ..... 8-11
  - Version data ..... 8-19
  - VIPA Configuration ..... 8-18
- W**
- Watchdog ..... 8-19
  - WinNCS
    - Ethernet coupler ..... 7-15
    - under Profibus-DP ..... 2-21; 2-32
  - Wiring ..... 20-9
- Z**
- ZNA ..... 9-28
  - ZVZ ..... 9-28