PROFIBUS/PROFINET
Manual

Servo drives ARS 2000

Metronix Meßgeräte und Elektronik GmbH
Kocherstraße 3
38120 Braunschweig
Germany

Phone: +49-(0)531-8668-0
Telefax: +49-(0)531-8668-555
E-mail: vertrieb@metronix.de
http://www.metronix.de

Translation of the original instructions

Copyrights

The information and data in this document have been composed to the best of our knowledge. However, deviations between the document and the product cannot be excluded entirely. For the devices and the corresponding software in the version handed out to the customer, Metronix guarantees the contractual use in accordance with the user documentation. In the case of serious deviations from the user documentation, Metronix has the right and the obligation to repair, unless it would involve an unreasonable effort. A possible liability does not include deficiencies caused by deviations from the operating conditions intended for the device and described in the user documentation.

Metronix does not guarantee that the products meet the buyer’s demands and purposes or that they work together with other products selected by the buyer. Metronix does not assume any liability for damages resulting from the combined use of its products with other products or resulting from improper handling of machines or systems.

Metronix Meßgeräte und Elektronik GmbH reserves the right to modify, amend, or improve the document or the product without prior notification.

Trademarks

Any product names in this document may be registered trademarks. The sole purpose of any trademarks in this document is the identification of the corresponding products.

ServoCommander® is a registered trademark of Metronix Meßgeräte und Elektronik GmbH.

Revision Information	
Author:	Metronix Meßgeräte und Elektronik GmbH
Manual title:	PROFIBUS/PROFINET Manual „Servo drives ARS 2000“
File name:	Profi-HB_ARS2000_EN.docx
Version 5.0	May 2016

Table of Contents

List of Figures

List of Tables

1 General

1.1 Documentation

This manual describes the field bus connection of ARS 2000 servo drives under PROFIBUS-DP and PROFINET. It describes the adjustment of the physical parameters, the activation of the PROFIBUS or PROFINET communication, the integration into the master connection and the communication with the servo drive. It is intended for persons who are familiar with this servo drive model.

It contains safety notes which have to be complied with.

> The term “servo drive ARS 2000” in this manual includes the servo drives of the series ARS 2000 and ARS 2000 FS.
>
> PROFINET is only available for the ARS 2000 FS devices.
>
> Further information on the individual devices can be found in the corresponding product manuals.
>
> You can find these documents on our homepage for download: http://www.metronix.de

1.2 PROFIBUS and PROFINET documentation

PROFIBUS (**PRO**cess **FI**eld**BUS**) and PROFINET (**PRO**cess **FI**eld **NET**work) is a standard developed by the German association PROFIBUS Nutzerorganisation e.V. A complete description of the fieldbus systems can be found in the following standards:

IEC 61158 "Digital data communication for measurement and control – Fieldbus for use in industrial control systems": This standard has several parts and defines the "Fieldbus Protocol Types". In accordance with these types, PROFIBUS is specified as type 3 and PROFINET as type 10. There are two different PROFIBUS versions. One of them is PROFIBUS-DP for the rapid data exchange in production engineering and building automation applications (DP = decentralised peripherals). The standard also describes the integration in the ISO/OSI layer model.

IEC 61784-2 "Industrial communication networks - Profiles - Part 2: Additional fieldbus profiles for real-time networks based on ISO/IEC 8802-3 (IEC 61784-2:2014)": This standard defines the real-time classes (RTC = real-time class) based on the reaction time.

Further information, contact addresses etc. can be found under **www.profibus.com**.

Further documentation (in German) concerning the use of PROFIBUS-DP and PROFINET:

1. PROFIBUS-DP
   Grundlagen, Tips und Tricks für Anwender
   Manfred Popp
   Hüthig-Verlag, Heidelberg 1998

2. Dezentralisieren mit PROFIBUS-DP
   Aufbau, Projektierung und Einsatz des PROFIBUS-DP mit Simatic S7
   Josef Weigmann, Gerhard Kilian
   Siemens, Erlangen/München 1998

3. Der neue Schnelleinstieg für PROFIBUS DP
   Von DP-V0 bis DP-V2
   Manfred Popp
   PROFIBUS Nutzerorganisation e.V., Karlsruhe 2002

4. PROFIdrive – Profile Drive Technology,
   Vers. 3.1,
   PROFIBUS Nutzerorganisation e.V., Karlsruhe

5. IEC 61158 - Feldbus für industrielle Leitsysteme

6. Industrielle Kommunikation mit PROFINET
   Manfred Popp
   PROFIBUS Nutzerorganisation e.V., 2014

2 Safety notes for electrical drives and controllers

2.1 Symbols and signs

Information

Important information and notes.

Caution!

Nonobservance may result in severe property damages.

DANGER!

Nonobservance may result in **property damages** and in **personal injuries**.

Caution! High voltage.

The safety note indicates a possible perilous voltage.

The parts of this document marked with this sign should give examples to make it easier to understand the use of single objects and parameters.

2.2 General notes

In case of damage resulting from non-compliance with the safety notes in this manual, Metronix Meßgeräte und Elektronik GmbH will not assume any liability.

Prior to the initial use you must read the chapter *Safety notes for electrical drives and controllersstarting on page 11.*

If the documentation in the language at hand is not understood accurately, please contact and inform your supplier.

Sound and safe operation of the servo drive requires proper and professional transportation, storage, mechanical installation, and project planning – with a consideration of the risks as well as the protective and emergency measures – plus the proper and professional electrical installation, operation, and maintenance of the devices.

Only trained and qualified personnel is authorised to handle electrical devices and systems:

TRAINED AND QUALIFIED PERSONNEL

in the sense of this product manual or the safety notes on the product itself are persons who are sufficiently familiar with the project, the setup, assembly, commissioning and operation of the product as well as all warnings and precautions as per the instructions in this manual and who are sufficiently qualified in their field of expertise:

- Education and instruction concerning the standards and accident prevention regulations for the application, or authorisation to switch devices/systems on and off and to ground them as per the standards of safety engineering and to efficiently label them as per the job demands.
- Education and instruction as per the standards of safety engineering regarding the maintenance and use of adequate safety equipment.
- First aid training.

The following notes must be read prior to the initial operation of the system to prevent personal injuries and/or property damages:

These safety notes must be complied with at all times.

Do not try to install or commission the servo drive before carefully reading all safety notes for electrical drives and controllers contained in this document. These safety instructions and all other user notes must be read prior to any work with the servo drive.

In case you do not have any user notes for the servo drive, please contact your sales representative. Immediately demand these documents to be sent to the person responsible for the safe operation of the servo drive.

If you sell, rent and/or otherwise make this servo drive available to others, these safety notes must also be included.

The user must not open the servo drive for safety and warranty reasons.

Professional control process design is a prerequisite for sound functioning of the servo drive!

DANGER!

Inappropriate handling of the servo drive and non-compliance of the warnings as well as inappropriate intervention in the safety features may result in property damage, personal injuries, electric shock or in extreme cases even death.

2.3 Danger resulting from misuse

DANGER!

High electrical voltages and high load currents!

Danger to life or serious personal injury from electrical shock!

DANGER!

High electrical voltage caused by wrong connections!

Danger to life or serious personal injury from electrical shock!

DANGER!

Surfaces of device housing may be hot!

Risk of injury! Risk of burning!

DANGER!

Dangerous movements!

Danger to life, serious personal injury or property damage due to unintentional movements of the motors!

2.4 Safety notes

2.4.1 General safety notes

⚠ The servo drive corresponds to IP20 degree of protection as well as pollution degree 2. Make sure that the environment corresponds to this degree of protection and pollution degree.

⚠ Only use replacement parts and accessories approved by the manufacturer.

⚠ The servo drives must be connected to the mains supply as per EN regulations, so that they can be cut off the mains supply by means of corresponding separation devices (for example main switch, contactor, power switch).

ℹ The servo drive may be protected using an AC/DC sensitive 300 mA fault current protection switch, type B (RCD = Residual Current protective Device).

⚠ Gold contacts or contacts with a high contact pressure should be used to switch the control contacts.

ℹ Preventive interference rejection measures should be taken for control panels, such as connecting contactors and relays using RC elements or diodes.

ℹ The safety rules and regulations of the country in which the device will be operated must be complied with.

⚠ The environment conditions defined in the product documentation must be kept. Safety-critical applications are not allowed, unless specifically approved by the manufacturer.

ℹ For notes on installation corresponding to EMC, please refer to the Product Manual for the respective servo drive. The compliance with the limits required by national regulations is the responsibility of the manufacturer of the machine or system.

⚠ The technical data and the connection and installation conditions for the servo drive are to be found in this product manual and must be met.

⚠ **DANGER!**

The general setup and safety regulations for work on power installations (for example DIN, VDE, EN, IEC or other national and international regulations) must be complied with.

Non-compliance may result in death, personal injury or serious property damages.

Without claiming completeness, the following regulations and others apply:

VDE 0100	Erection of power installations with nominal voltages up to 1000 V
EN 60204	Safety of machinery - Electrical equipment of machines -
EN 50178	Electronic equipment for use in power installations

2.4.2 Safety notes for assembly and maintenance

The appropriate DIN, VDE, EN and IEC regulations as well as all national and local safety regulations and rules for the prevention of accidents apply for the assembly and maintenance of the system. The plant engineer or the operator is responsible for compliance with these regulations:

The servo drive must only be operated, maintained and/or repaired by personnel trained and qualified for working on or with electrical devices.

Prevention of accidents, injuries and/or damages:

Additionally secure vertical axes against falling down or lowering after the motor has been switched off, e.g. by means of:

- Mechanical locking of the vertical axle,
- External braking, catching or clamping devices or
- Sufficient balancing of the axle.

The motor holding brake supplied by default or an external motor holding brake driven by the servo drive alone is not suitable for personal protection!

Keep the electrical equipment voltage-free using the main switch and protect it from being switched on again until the DC bus circuit is discharged, in the case of:

- Maintenance and repair work
- Cleaning
- long machine shutdowns

Prior to carrying out maintenance work make sure that the power supply has been turned off, locked and the DC bus circuit is discharged.

The external or internal brake resistor carries dangerous DC bus voltages during operation of the servo drive and up to 5 minutes thereafter. Contact may result in death or serious personal injury.

Be careful during the assembly. During the assembly and also later during operation of the drive, make sure to prevent drill chips, metal dust or assembly parts (screws, nuts, cable sections) from falling into the servo drive.

Also make sure that the external power supply of the servo drive (24 V) is switched off.

The DC bus circuit or the mains supply must always be switched off prior to switching off the 24 V servo drive supply.

⚠ Carry out work in the machine area only, if AC and/or DC supplies are switched off. Deactivated output stages or deactivated servo drive enable signals are no suitable means of locking. In the case of a malfunction the drive may accidentally be put into action.

⚠ Initial operation must be carried out with idle motors, to prevent mechanical damages e.g. due to the wrong direction of rotation.

⚠ Electronic devices are never fail-safe. It is the user's responsibility, in the case an electrical device fails, to make sure the system is transferred into a secure state.

⚠ The servo drive and in particular the brake resistor, externally or internally, can assume high temperatures, which may cause serious burns.

2.4.3 Protection against contact with electrical parts

This section only concerns devices and drive components carrying voltages exceeding 50 V. Contact with parts carrying voltages of more than 50 V can be dangerous for people and may cause electrical shock. During operation of electrical devices some parts of these devices will inevitably carry dangerous voltages.

⚠ **DANGER!**

High electrical voltage!

Danger to life, danger due to electrical shock or serious personal injury!

The appropriate DIN, VDE, EN and IEC regulations as well as all national and local safety regulations and rules for the prevention of accidents apply for the assembly and maintenance of the system. The plant engineer or the operator is responsible for compliance with these regulations:

⚠ Before switching on the device, install the appropriate covers and protections against accidental contact. Rack-mounted devices must be protected against accidental contact by means of a housing, e.g. a switch cabinet. The national regulations for safety/accident prevention must be complied with!

⚠ Always connect the ground conductor of the electrical equipment and devices securely to the mains supply. Due to the integrated line filter the leakage current exceeds 3.5 mA!

⚠ Comply with the minimum copper cross-section for the ground conductor over its entire length (see for example EN 60800-5-1).

⚠ Prior to the initial operation, even for short measuring or testing purposes, always connect the ground conductor of all electrical devices as per the terminal diagram or connect it to the ground wire. Otherwise the housing may carry high voltages which can cause electrical shock.

⚠ Do not touch electrical connections of the components when switched on.

⚠ Prior to accessing electrical parts carrying voltages exceeding 50 Volts, disconnect the device from the mains or power supply. Protect it from being switched on again.

⚠ For the installation the amount of DC bus voltage must be considered, particularly regarding insulation and protective measures. Ensure proper grounding, wire dimensioning and corresponding short-circuit protection.

⚡ ⚠ The device comprises a rapid discharge circuit for the DC bus as per EN 60204-1. In certain device constellations, however, mostly in the case of parallel connection of several servo drives in the DC bus or in the case of an unconnected brake resistor, this rapid discharge may be rendered ineffective. The servo drives can carry voltage until up to 5 minutes after being switched off (residual capacitor charge).

2.4.4 Protection against electrical shock by means of protective extra-low voltage (PELV)

All connections and terminals with voltages of up to 50 Volts at the servo drive are protective extra-low voltage, which are designed safe from contact in correspondence with the following standards:

- International: IEC 60364-4-41
- European countries within the EU: EN 61800-5-1

⚠ **DANGER!**

High electrical voltages due to wrong connections!

Danger to life, risk of injury due to electrical shock!

Only devices and electrical components and wires with a protective extra low voltage (PELV) may be connected to connectors and terminals with voltages between 0 to 50 Volts.

Only connect voltages and circuits with protection against dangerous voltages. Such protection may be achieved by means of isolation transformers, safe optocouplers or battery operation.

2.4.5 Protection against dangerous movements

Dangerous movements can be caused by faulty control of connected motors, for different reasons:

- Improper or faulty wiring or cabling
- Error in handling of components
- Error in sensor or transducer
- Defective or non-EMC-compliant components
- Software error in superordinate control system

These errors can occur directly after switching on the device or after an indeterminate time of operation.

The monitors in the drive components for the most part rule out malfunctions in the connected drives. In view of personal protection, particularly the danger of personal injury and/or property damage, this may not be relied on exclusively. Until the built-in monitors come into effect, faulty drive movements must be taken into account; their magnitude depends on the type of control and on the operating state.

DANGER!

Dangerous movements!

Danger to life, risk of injury, serious personal injuries or property damage!

For the reasons mentioned above, personal protection must be ensured by means of monitoring or superordinate measures on the device. These are installed in accordance with the specific data of the system and a danger and error analysis by the manufacturer. The safety regulations applying to the system are also taken into consideration. Random movements or other malfunctions may be caused by switching the safety installations off, by bypassing them or by not activating them.

2.4.6 Protection against contact with hot parts

DANGER!

Housing surfaces may be hot!

Risk of injury! Risk of burning!

Do not touch housing surfaces in the vicinity of heat sources! Danger of burning!

Before accessing devices let them cool down for 10 minutes after switching them off.

Touching hot parts of the equipment such as the housing, which contain heat sinks and resistors, may cause burns!

2.4.7 Protection during handling and assembly

Handling and assembly of certain parts and components in an unsuitable manner may under adverse conditions cause injuries.

DANGER!

Risk of injury due to improper handling!

Personal injury due to pinching, shearing, cutting, crushing!

The following general safety notes apply:

⚠ Comply with the general setup and safety regulations on handling and assembly.

⚠ Use suitable assembly and transportation devices.

⚠ Prevent incarcerations and contusions by means of suitable protective measures.

⚠ Use suitable tools only. If specified, use special tools.

⚠ Use lifting devices and tools appropriately.

⚠ If necessary, use suitable protective equipment (for example goggles, protective footwear, protective gloves).

⚠ Do not stand underneath hanging loads.

⚠ Remove leaking liquids on the floor immediately to prevent slipping.

3 Wiring and pin assignment

3.1 PROFIBUS

3.1.1 Pin assignment

In the ARS 2000 product range, the PROFIBUS interface has been realised in the form of an optional technological plug-in module. Normally it is pre-plugged at the factory, but it can also be retrofitted.

In accordance with EN 50170, the PROFIBUS connector is a female 9-pin DSUB connector (at the technological plug-in module).

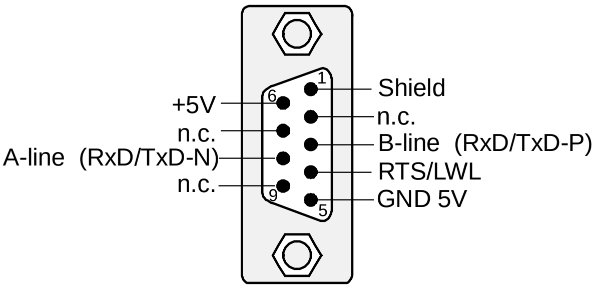


Figure 3-1: PROFIBUS connector for ARS 2000

PROFIBUS wiring

Because of the very high baud rates we recommend using standardised cables and connectors. They offer additional diagnose functionalities and facilitate rapid analysis of the field bus hardware in the event of malfunctions.

When setting up the PROFIBUS network, compliance with the information given in the standard literature and the following information and notes is strongly recommended in order to obtain a stable, trouble-free system. Improper wiring may lead to problems with the PROFIBUS causing the servo drive to automatically deactivate itself while signalling an error.

3.1.2 Termination and bus terminating resistors

Every bus segment of the PROFIBUS network has to be equipped with bus terminating resistors to minimise line reflections, to guarantee practically constant load behaviour on the bus and to adjust a defined rest potential on the line. Termination has to be provided at the **beginning** and at the **end of every bus segment**.

The terminating resistors of the PROFIBUS module of the ARS 2000 product range are already integrated in the module so that external connections (special connectors) are not required. The terminating resistors can be activated by setting the two DIP switches on the module (switches set to ON).

Bus terminating resistors

The DIP switches for activating the bus terminating resistors are already integrated in the technological modules of the ARS 2000 product range.

External connection is also possible. The power supply of 5 V required for the externally connected terminating resistors is supplied to the PROFIBUS connector of the module (see pin assignment).

Faulty bus termination is the most common cause of errors in the event of malfunctions.

If the adjusted baud rate is > 1.5 MBaud, connectors with integrated line inductors (110 nH) have to be used because of the capacitive load of the device and the resulting line reflections.

3.1.3 Bus cable

PROFIBUS cable:

SIMATIC NET, PB FC standard bus cable, twisted pair, shielded, special design for quick connection, 20 m.

Manufacturer: SIEMENS

Order no.: 6XV1 830-0EN20

3.2 PROFINET

3.2.1 Connection and display elements of the technology module

For the ARS 2000 series, the PROFINET interface has been realised in the form of an optional technology plug-in module with two RJ45 jacks. Normally, it is already plugged in at the factory, but it can also be retrofitted.

1	ACT-LED
2	LNK-LED
3	SF-LED
4	BF-LED
5	PROFINET interface

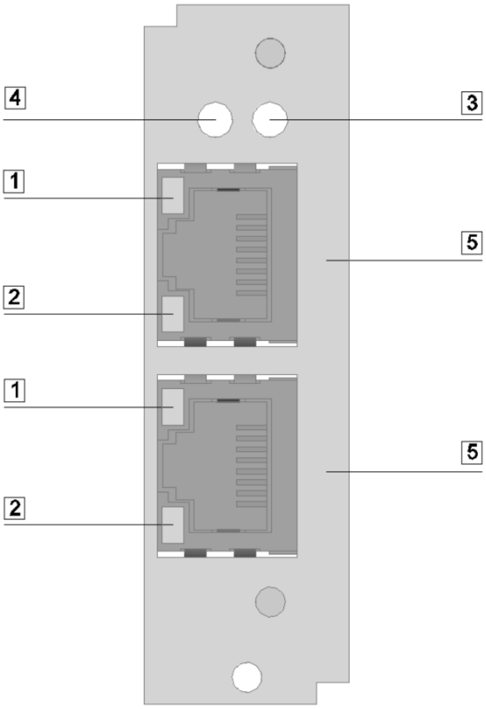


Figure 3-2: Connection and display elements of the PROFINET module

LED	Status	Meaning
SF	Off	No system errors
	Lights up red	Watchdog timeout Channel diagnostics General or extended diagnostics System error
	Flashes red (1 Hz for 3 s)	PROFINET device identification
BF	Off	No bus errors
	Lights up red	No configuration Fault concerning the physical link No physical link
	Flashes red (2 Hz)	No data transfer

LNK	Off	No link available
	Lights up green	Link available
ACT	Off	No Ethernet communication available
	Lights up orange	Ethernet communication available
	Flashes orange	Ethernet communication active

Table 3-1: PROFINET LEDs

3.2.2 Pin assignment

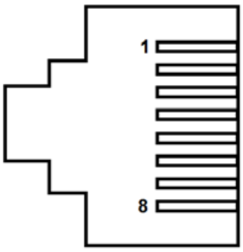
Socket	Pin no.	Name	Description
	1	RX-	Receive signal-
	2	RX+	Receive signal+
	3	TX-	Transmission signal-
	4	-	-
	5	-	-
	6	TX+	Transmission signal+
	7	-	-
	8	-	-

Table 3-2: Pin assignment of the PROFINET interface

3.2.3 Bus cable

A PROFINET copper cable is usually a 4-core, shielded copper cable. Like in the case of standard Ethernet applications, the maximum distance that can be covered by way of copper cables is limited to 100 m between the communication endpoints. The link is defined as a PROFINET end-to-end link.

PROFINET cabling

The only cables that can be used in automation systems are PROFINET cables. PROFINET cables have a corresponding manufacturer's declaration.

The joint use of power cables and copper cables for communication purposes is subject to regulations in order to minimise the electromagnetic influence of the power cables on the communication cables.

When setting up the PROFINET network, it is essential to follow the advice that is given in the technical literature and to comply with the information and notes hereinafter in order to realise a stable, trouble-free system.

4 Fundamental principles: PROFIBUS-DP, PROFINET and PROFIdrive

The technology plug-in module of the ARS 2000 servo drive can either be combined with a PROFIBUS plug-in module or with a PROFINET plug-in module. As shown in Figure 4-1, the selection of the fieldbus system does not affect the control of the application. The control and monitoring data of the application are transferred to the fieldbus via the active fieldbus and transmission system.

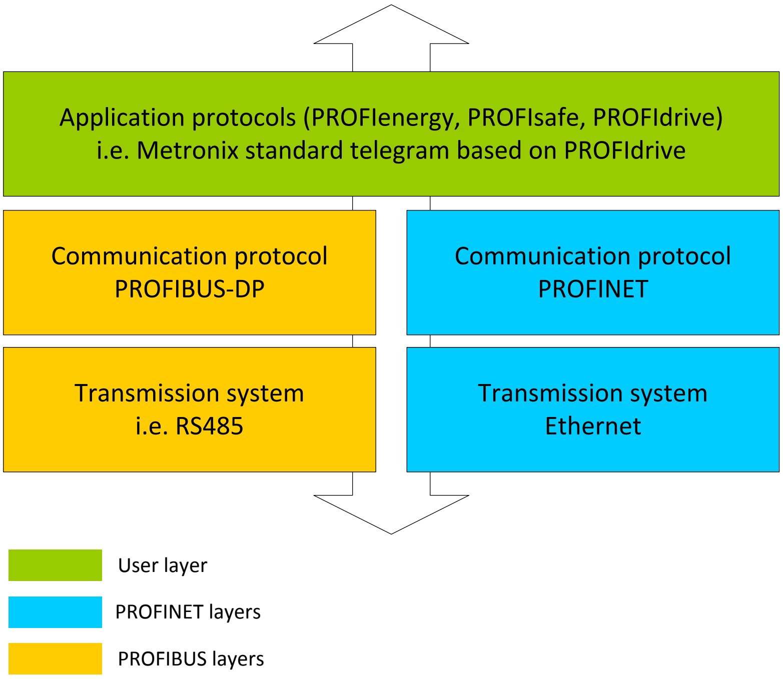

Application protocols (PROFIenergy, PROFIsafe, PROFIdrive)
i.e. Metronix standard telegram based on PROFIdrive
Communication protocol PROFIBUS-DP
Communication protocol PROFINET
Transmission system i.e. RS485
Transmission system Ethernet
User layer
PROFINET layers
PROFIBUS layers

Figure 4-1: Comparison of PROFIBUS and PROFINET

The following chapters provide a brief description of the fundamental characteristics of PROFIBUS-DP and PROFINET. This is followed by a presentation of the Metronix-specific application profile based on PROFIdrive.

4.1 Overview PROFIBUS-DP

The DP communication protocol (DP = decentralised peripherals) is intended for quick data exchange on the field level. On this level, central automation devices, such as PLC, PC or process control systems communicate via a quick serial connection with decentralised field devices, such as I/O devices, drives, valves, measuring transducers and analysis devices. Data exchange with the decentralised devices is mainly cyclic. The required communication functions are defined by the DP basic functions (version DP-V0).

To meet special requirements of different areas of application, the basic functions of DP were extended and special functions were added so that now three versions of DP, i.e. DP-V0, DP-V1 and DP-V2, are available. Every version has a special main focus. They can be described as follows:

- DP-V0 This version offers the basic functionality of DP, such as cyclic data exchange and station-, module- and channel-specific diagnosis.

- DP-V1 This version includes extensions for process automation like, for example, acyclic data exchange for parameterisation, operation, monitoring and alarm management of intelligent field devices in parallel to cyclic exchange of useful data. Bus devices can be accessed online through engineering tools. In addition, DP-V1 includes alarms like the status alarm, the update alarm and a manufacturer-specific alarm.

- DP-V2 This version includes further extensions and is mainly intended to meet the requirements of drive applications. Additional functions such as, for example, isochronous slave operation and slave-to-slave communication allow DP-V2 to be used as a drive bus for controlling quick sequences of motion in drive shafts.

  This version requires special hardware. At present, the control systems and target applications of the ARS 2000 product range are not equipped with this hardware. This is why support of DV-V2 is not intended.

The versions of DP are specified in detail in IEC 61158 (see chapter 1.2).

Every DP system comprises different types of devices. Three types can be distinguished: DP master class 1, DP master class 2 and DP slaves. DP can be used to realise mono- and multi-master systems and thus offers a high level of flexibility for system configuration. A maximum of 126 devices (masters or slaves) can be connected to the bus.

Due to the extensive diagnosis functions of DP, errors can be localised very quickly. Diagnosis messages are transmitted on the bus and collected at the master.

DP master class 1 (DPM1) is a central control system which exchanges information cyclically with decentralised stations (slaves) in a defined message cycle. Typical DPM1 devices are programmable logic controls (PLCs) or PCs, for example.

DP master class 2 (DPM2) devices are engineering, project or operating devices. They are used for start-up, maintenance and diagnosis, to configure the connected devices, to evaluate measured values and parameters and to check the device status. A DPM2 device does not have to be permanently connected to the bus system.

A slave is peripheral device (I/O device, drive, etc.) which reads in process information and/or uses output information to intervene in the process. In terms of communication, slaves are passive devices as they reply only to direct requests from a DPM1 or DPM2 device.

The transfer of data with the DP-V0 service requires a specification on the master side as well as on the slave side about the quantity of the data to be transferred and also about their interpretation. When planning a PROFIBUS connection, the user must make this specification. Only then should the parameterisation of the fieldbus connection on both sides commence.

4.2 PROFINET IO overview

4.2.1 Fundamental principles

PROFINET IO (Input - Output) enables the connection of decentralised field devices, such as I/Os, drives, valves, transducers, or analysis devices, to a central automation device, such as a PLC, PC, or process control system. It can be considered as the direct successor of PROFIBUS-DP. Data transfer is based on the Fast Ethernet standard transmission with 100 Mbit/s. PROFINET IO follows the provider-consumer model for the data exchange. Depending on the conformance class (abbreviation: CC), it is suitable for bus cycle times of several milliseconds up to 31.25 µs.

Three conformance classes (CC-A, CC-B, and CC-C) that build upon one another specify the functionality and real-time characteristics of PROFINET IO.

- CC-A This class provides basic functions for PROFINET IO with RT communication. All IT services can be used without restriction. Typical applications are found, for example, in building automation. Wireless communication is specified for this class.
- CC-B This class extends the concept to include network diagnostics via IT mechanisms as well as topology information. The system redundancy function, which is important for process automation, is contained in an extended version of CC-B named CC-B(PA).
- CC-C This class describes the basic functions for devices with hardware-supported bandwidth reservation and synchronisation (IRT communication) and is thus the basis for isochronous applications.

The conformance classes also serve as the basis for the certification and for the cabling guidelines.

4.2.2 Diagnostic alarms

PROFINET includes diagnostic alarms. A diagnostic alarm is transmitted to the controller in the case of a special servo drive problem (short circuit, angle encoder error, etc.). It actuates the red diagnostics LED in the controller.

For a rapid analysis, the cause of the problem is stated in plain text in the PROFINET controller (see chapter 14.3.5), e.g.

E08: Angle encoder: E08-6: Angle encoder communication error

The error number consists of a main index (HH) and subindex (S). The main index is transferred in the manufacturer-specific section of the channel diagnostics (ChannelErrorType) 0x0100 … 0x7FFF. The subindex is transferred in the manufacturer-specific section of the extended channel diagnostics (ExtChannelErrorType) 0x1000 … 0x100F.

Error number	ChannelErrorType	ExtChannelErrorType
08-6	$HH_h + 1000_h = 1008_h$	$S_h + 1000_h = 1006_h$

Table 4-1: Channel diagnostics example

4.3 PROFIdrive overview

The “PROFIBUS profile for drive technology” or short PROFIdrive is a manufacturer standard for implementing PROFIBUS interfaces for drives. It has been defined by the association of PROFIBUS users. Just like CANopen, its aim is to offer the user a defined interface for programming servo drives. This interface should be as manufacturer-independent as possible.

PROFIdrive specifies the configuration, diagnosis, data exchange and state machines with a master. Furthermore, so-called application classes are defined. There are different versions of the PROFIdrive specification with significant differences. The ARS 2000 product range follows exclusively version 3.1 (4).

The PROFIdrive specification (4) also defines a state machine for device control. This state machine is addressed using a control word and a status word. The meaning of the individual bits is also defined in the specification. The functions of these two words have been adopted to a large extent. Manufacturer-specific deviations are related to certain details and documented and marked from chapter 6 onwards.

The ARS 2000 product range covers a part of the application classes defined in the PROFIdrive specification.

The fieldbus technologies PROFIBUS-DP and PROFINET do not specify the form and meaning of the useful data themselves. This is why the concept of parameter numbers (PNUs) has been adopted. These parameter numbers have an optional subindex and there are pre-defined or reserved areas among these PNUs. In addition, there is room for manufacturer-specific PNUs.

There are several manufacturer-specific PNUs for connecting the ARS 2000 servo drives. In addition, further object directories can be accessed (in some cases with certain restrictions) as of a certain state of extension level. Among these directories are the object directory of Metronix communication objects and the CANopen object directory.

5 PROFIBUS and PROFINET connection

5.1 Introduction

The set-up of a fully functional PROFIBUS or PROFINET connection requires several steps. Some of these settings should or must be performed prior to the activation or start of the bus communication. This chapter provides an overview of the required steps. The exact procedure is described in the following chapters.

The data transfer is realised by way of telegrams. The number of data to be transferred and the meaning of these data must be specified on the master side as well as on the slave side prior to the start of the data exchange. It is only then that the parameterisation of the fieldbus connection on both sides should commence. We recommend parameterising the slave first. This should then be followed by the configuration of the master. Provided that the parameterisation is correct, the application is immediately ready without causing any communication errors.

There are only minor differences between PROFINET and PROFIBUS in terms of the configuration and operation. This is why the next chapters apply to both fieldbus systems. Any differences are clearly indicated to the user.

5.2 Slave overview

This section provides an overview of the necessary steps for the parameterisation and configuration of the slave. Since some of the parameters will not become effective until after they have been stored and a reset has been performed, we recommend the following workflow:

1. Configuration of the telegrams with the **telegram editor, chapter 6**
2. Selection and parameterisation of the **physical units, chapter 7**
3. Configuration and activation of the **operating parameters, chapter 8**

On the slave side, the meaning of the data is specified by entering the corresponding parameter numbers (PNU). This is realised by way of the **telegram editor** of the Metronix ServoCommander® parameterisation program. The number of bytes of the PNU is displayed automatically. For checking purposes, the total length of the telegram is always indicated in the telegram editor.

Important process data concerning the position, speed, and acceleration are handed over in **physical units**. These should be parameterised prior to starting the communication, since they define the interpretation of the data in the servo drive.

In the speed control mode, the setpoint selector cannot be changed when the PROFIBUS or PROFINET communication is **active**. This is why it must be parameterised accordingly prior to activating the communication.

Following the completion of these steps, the **operating parameters** of the PROFIBUS connection must be configured.

Prior to activating the PROFIBUS communication, the slave address must be configured correctly. For this purpose, additional options for controlling the address assignment can be activated via an external connection.

5.3 Master overview

5.3.1 PROFIBUS

This section provides an overview of the necessary steps for the parameterisation and configuration of the master. The following procedure is recommended:

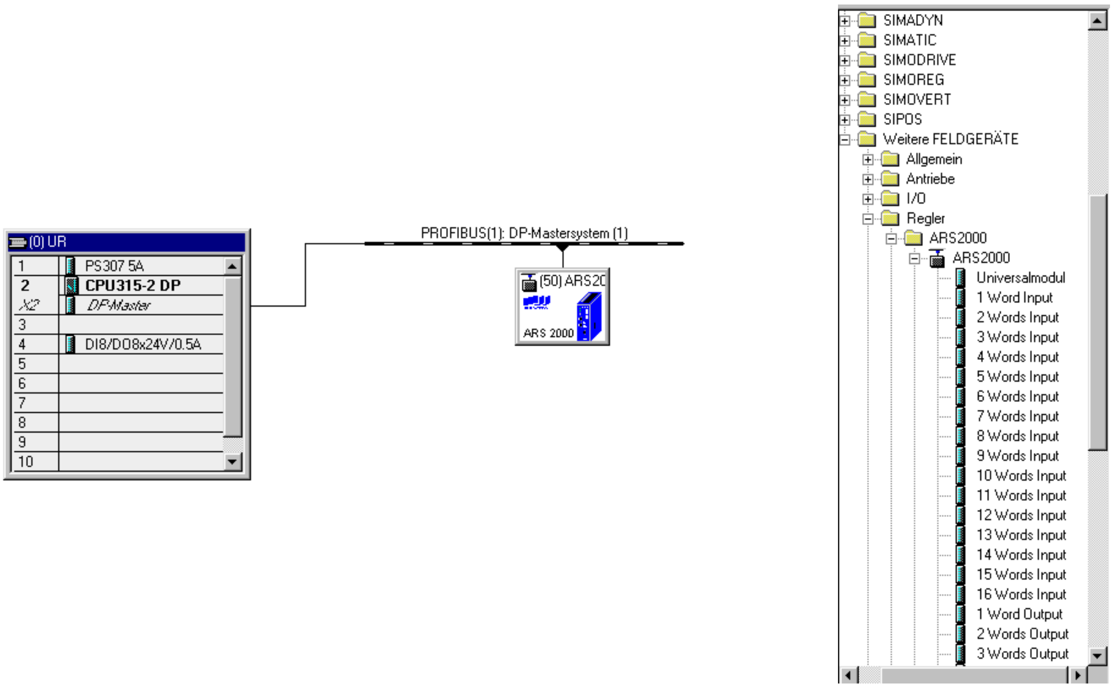
1. Installation of the **GSD file**
2. Specification of the **slave address**
3. Configuration of the **input and output data**

The servo drive must be integrated into the PROFIBUS on the master side. For this purpose, the **GSD file** must be installed, if this has not been done yet. Then, the address as well as the input and output data for the slave must be configured.

The following section describes this based on an example using integration under SIEMENS SIMATIC S7.

In order to integrate the servo drive into the PROFIBUS network, the servo drive must be selected in the hardware catalog. The folder ARS 2000 must be transferred to the PROFIBUS DP master system by drag & drop. Immediately afterwards, the system requests the **slave address**.

The bitmap of the servo drive is displayed on the master side. It must be selected. Then, the length of the **input and output data** must be specified. For this purpose, special modules have been prepared in the GSD file. One module for the input data and one module for the output data, including the corresponding length, must be added to one port each.



(0) UR

1	PS307 5A
2	**CPU315-2 DP**
X2	*DP-Master*
3	
4	DI8/DO8x24V/0.5A
5	
6	
7	
8	
9	
10	

Figure 5-1: Hardware configuration under SIEMENS S7

In order to ensure perfect operation, we recommend specifying only one section each for the input and output data (see the example above).

Further information concerning the hardware configuration can be found in chapter 6 **Telegram editor** and chapter 13 or in the Application Note with complete example projects, including the function blocks and data blocks for SIMATIC S7.

5.3.2 PROFINET

This section provides an overview of the necessary steps for the parameterisation and configuration of the master. The following procedure is recommended:

1. Installation of the **GSDML file**
2. Specification of the **device name**
3. Configuration of the **input and output data**

On the master side, the servo drive must be integrated into the PROFINET network. The following section describes this based on an example using integration under SIEMENS SIMATIC S7.

If the servo drive is not included in the device or hardware catalog, the **GSDML file** must be installed first. A brief description can be found in chapter 14.2.1.

In order to integrate the servo drive into the PROFINET network, the servo drive must be selected in the hardware catalog. The ARS 2000 head module must be dragged and dropped into the topology view of the device configuration. After the selection of the network view, the servo drive must be assigned to an I/O controller via the link "Not assigned".

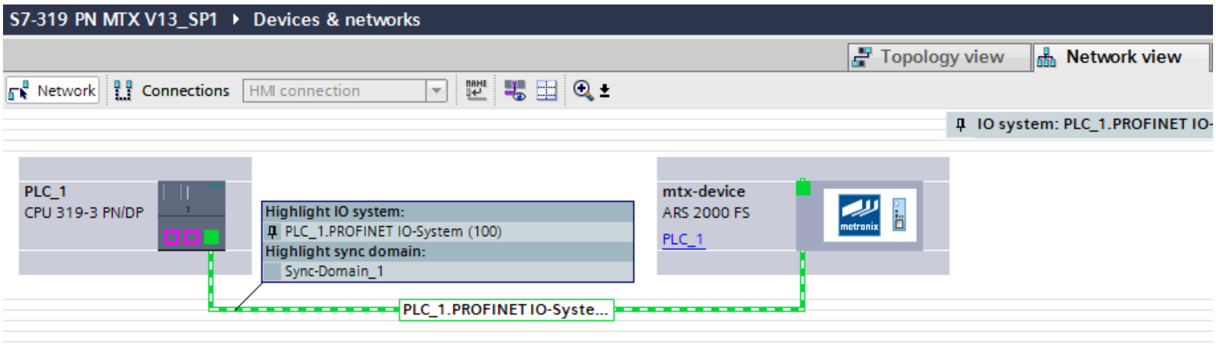


Figure 5-2: IO controller assignment under SIEMENS S7

Then, the **device name** must be specified. This method has been chosen for PROFINET, since names are easier to handle than complex IP addresses. After clicking the Metronix logo of the servo drive, the device name can be changed under "General" in the "Properties" window.

Then, the length of the **input and output data** must be specified. For this purpose, special modules have been prepared in the GSDML file. One module for the input data and one module for the output data, including the corresponding length, must be added to one port each.

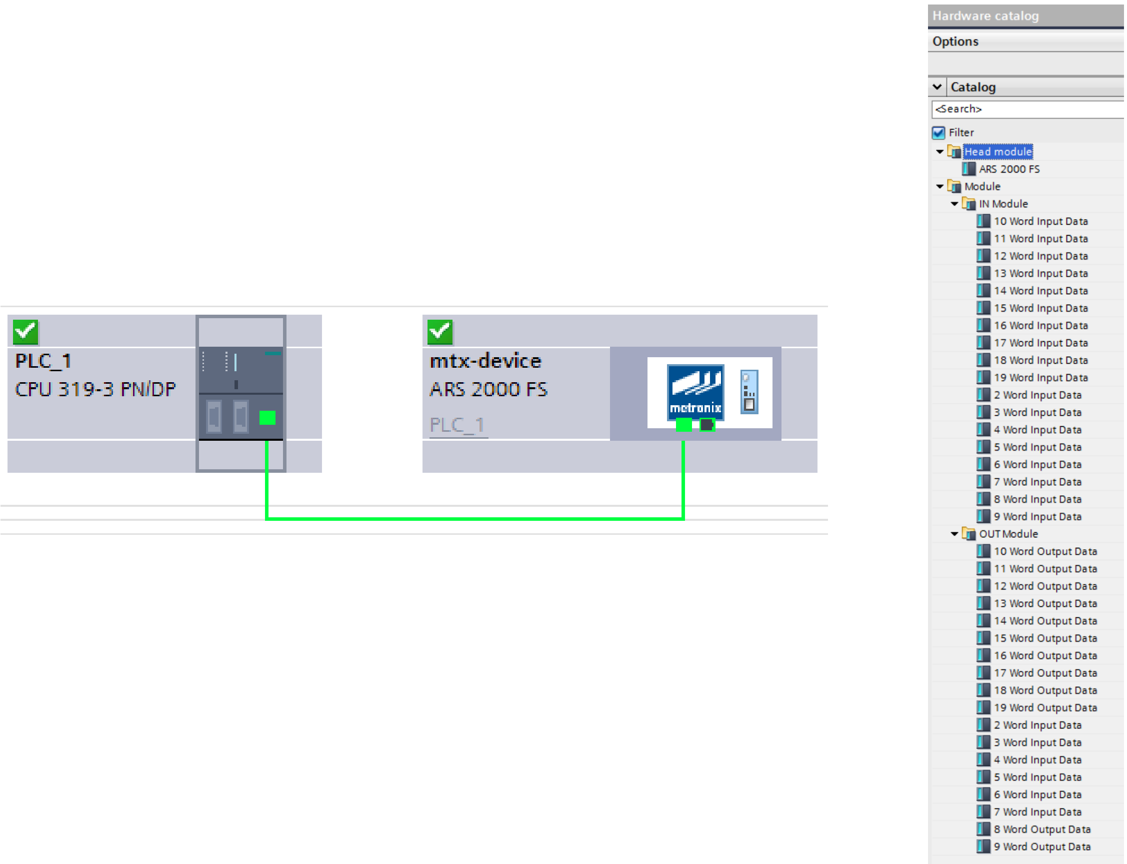


Figure 5-3: Hardware configuration under SIEMENS S7

Further information concerning the hardware configuration can be found in chapter 6 **Telegram editor** or chapter 14 **Function blocks for the ARS 2000 servo drives**.

6 Telegram editor

6.1 Introduction

The telegram editor is used to define the interpretation of the received or transmitted data by the servo drive. Metronix ServoCommander® has separate telegram editors for PROFINET and PROFIBUS:

- **PROFINET:** The telegram editor can be found on the menu bar under **Parameters - Field bus - PROFINET – Telegram editor**
- **PROFIBUS:** The telegram editor can be found on the menu bar under **Parameters - Field bus - PROFIBUS – Telegram editor**

Data are exchanged cyclically by way of telegrams. The following two groups can be distinguished:

- **Receive telegrams:** Data that are transferred from the master to the slave; also known as **output data**.
- **Response telegrams:** Data that are transferred from the slave to the master; also known as **input data**.

A telegram can have a maximum of 10 entries.

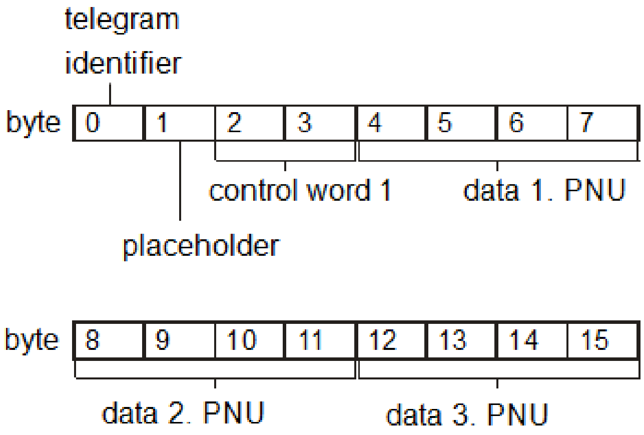


Figure 6-1: Example of the ARS 2000 telegram format

Figure 6-1 shows an example of an ARS 2000 standard telegram from the master to the slave. Apart from the identifier in byte 0, this telegram type requires the PROFIdrive control word 1 for the device control in bytes 2 and 3. The content of the subsequent bytes can be configured as desired. In this example, 3 additional data are transferred, each with 4 bytes. The resulting length of the complete telegram is 16 bytes.

In the project of the master, data areas are created, e.g. data blocks. These data areas are used to store the input and output data of the master and slave. During the set-up of the project, the user must specify the contents and the sequence of the contents as well as the size of the data area in an identical manner on the master side and on the slave side.

This parameterisation should be performed prior to activating the communication.

6.2 Receive telegrams

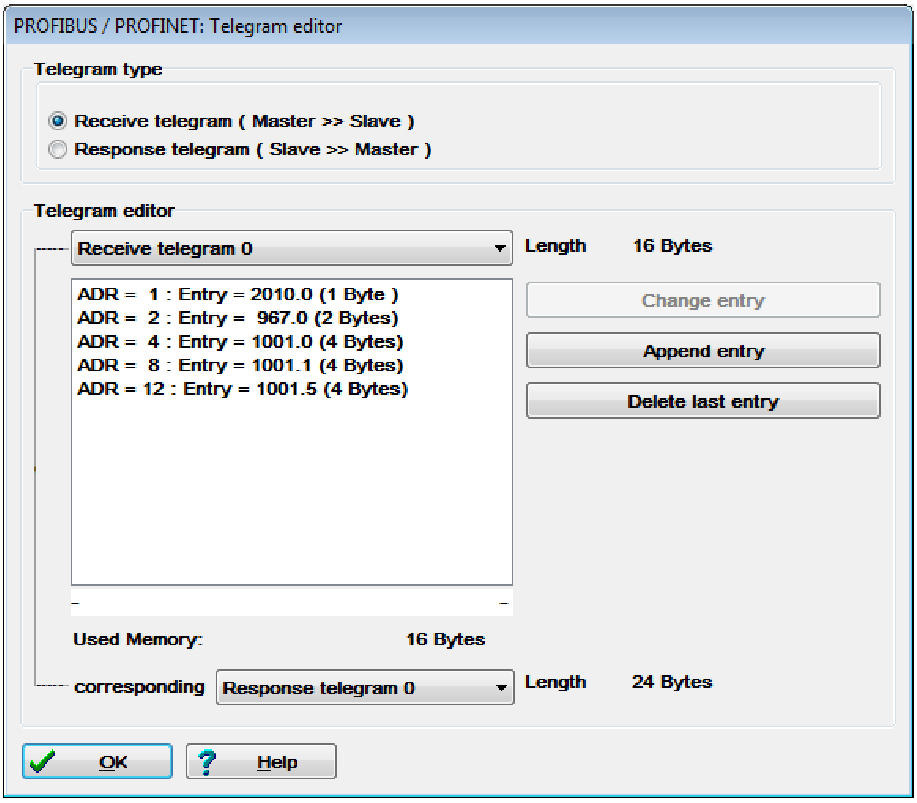
The ARS 2000 servo drive supports 4 receive telegrams. Some of these telegrams are reserved strictly for one specific operating mode. This makes it easier for the user to switch from one operating mode to another. An additional parameter for the operating mode does not have to be transmitted together with the telegram. The following table shows how the different receive telegrams are connected to specific operating modes:

Telegram	**Operating mode**
Receive telegram 0	Positioning
Receive telegram 1	Speed control
Receive telegram 2	None (reserved for torque control)
Receive telegram 3	None (free telegram format)

Once a corresponding telegram identifier is read in the servo drive, a check is performed and the associated operating mode is parameterised.

The necessary parameter numbers have to be entered for every receive telegram. They include the information concerning the meaning of the data in the telegram. In the case of receive telegrams 0..2, a so-called control word is entered at address 2 (length: 2 bytes). This consistent convention facilitates the creation of applications and the use of the Metronix example projects for SIEMENS SIMATIC S7. The other entries can be chosen as desired from the object directory of the parameter numbers. The only thing that has to be considered is whether they are suitable. It is, for instance, not possible to enter pure actual value data into receive telegrams.

As far as receive telegrams are concerned, the associated response telegrams have to be selected, too. The user can specify and configure a specific response telegram for every receive telegram. In most cases, however, it is easier to use the same response telegram for all operating modes (receive telegrams 0..2). This reduces the necessary programming workload at the master end. In addition, the master usually requires the same actual value data from the servo drive in all operating modes.



PROFIBUS / PROFINET: Telegram editor

Telegram type

● Receive telegram (Master >> Slave)
○ Response telegram (Slave >> Master)

Telegram editor

Receive telegram 0 — Length 16 Bytes

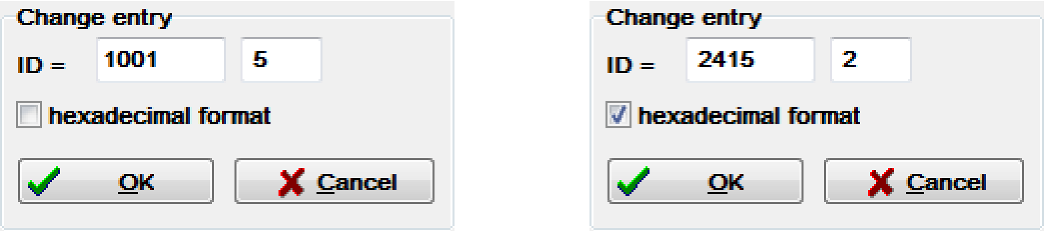
ADR = 1 : Entry = 2010.0 (1 Byte)
ADR = 2 : Entry = 967.0 (2 Bytes)
ADR = 4 : Entry = 1001.0 (4 Bytes)
ADR = 8 : Entry = 1001.1 (4 Bytes)
ADR = 12 : Entry = 1001.5 (4 Bytes)

Used Memory: 16 Bytes

corresponding Response telegram 0 — Length 24 Bytes

Figure 6-2: Composition of a receive telegram

Figure 6-2 shows the example of receive telegram 0 (positioning mode). The entries can be highlighted and changed directly or deleted successively starting with the last entry. When an entry is highlighted, an additional field is displayed into which the parameter number can be entered:



Change entry
ID = 1001 5
□ hexadecimal format

Change entry
ID = 2415 2
☑ hexadecimal format

Figure 6-3: Entry of a PNU (left) or of a CAN object (right)

The numbers of PNUs are entered in decimal format (checkbox “hexadecimal” not marked). The checkbox “hexadecimal” is marked if objects of the CANopen object dictionary are used. The given number can directly be taken from the CANopen manual. A handshake to the connected servo drive starts when an entry is entered into the field ID and confirmed by a click onto the button **OK**. It is checked if the parameter exists and its number of bytes. For this reason, this function is not available in offline mode of the parameterisation program.

New telegram entries are added to the end. An additional diagnostic information is displayed above the button „Change entry“, if a communication could have been established between Master and Slave. The really configured length of the telegram from Master to Slave is displayed.

The example telegram in Figure 6-2 transmits the following parameters:

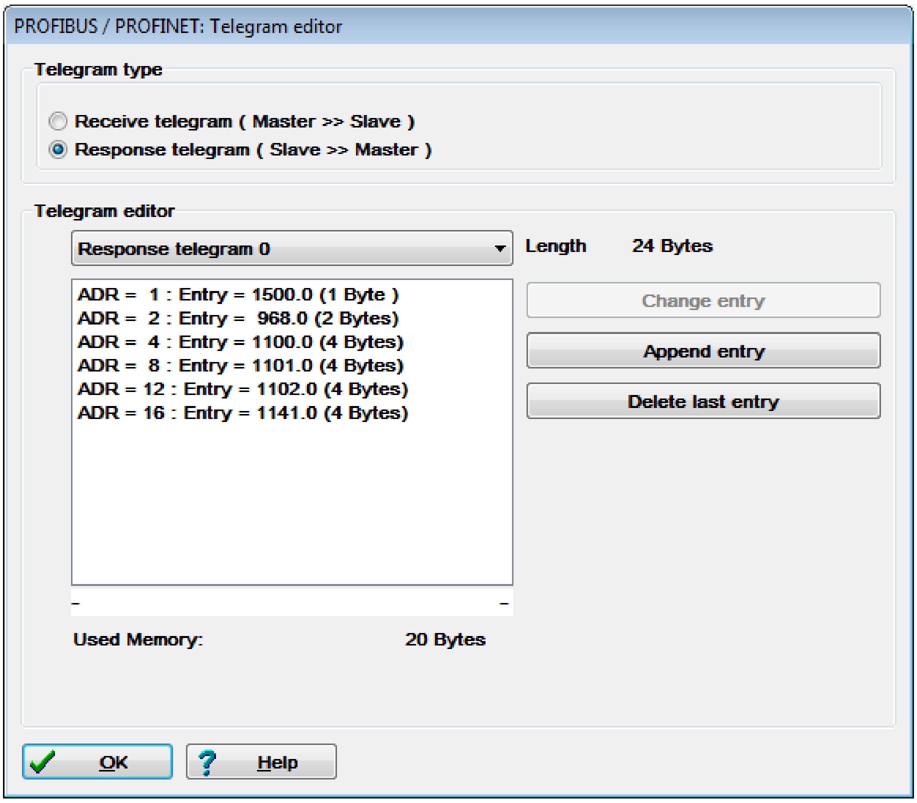
Addres s	Content (parameter number)	Description
0	Identifier (= 0xE0)	Fixed identifier
1	8-bit placeholder (PNU 2010.0)	free
2	Control Word 1 (PNU 967.0)	Control word for device control, has to be under this address
4	Target position (PNU 1001.0)	Target position, indication with the physical unit of a position set
8	Profile velocity (PNU 1001.1)	Profile velocity during positioning, indication with the physical speed unit set
12	Acceleration (PNU 1001.5)	Combination of the values for acceleration and deceleration, indication with physical unit for acceleration set

Detailed information on parameter numbers can be found in chapters 9, 10 and 12.

6.3 Response telegrams

The ARS 2000 servo drive supports 4 response telegrams. The necessary parameter numbers have to be entered for every response telegram. They include the information concerning the meaning of the data in the telegram. In the case of receive telegrams 0..2, a so-called status word is entered at address 2 (length: 2 bytes). This consistent convention facilitates the creation of applications and the use of the Metronix example projects for SIEMENS SIMATIC S7. The other entries can be chosen as desired from the object directory of the parameter numbers. The only thing that has to be considered is whether they are suitable. It is, for instance, not possible to enter write-only parameters into response telegrams.

Figure 6-4 shows the example of response telegram 0 (positioning mode). The entries can be highlighted and changed directly or deleted successively starting with the last entry. When an entry is highlighted, an additional field is displayed into which the parameter number can be entered. New telegram entries are added to the end. An additional diagnostic information is displayed above the button „Change entry“ if a communication could have been established between Master and Slave. The really configured length of the telegram from Master to Slave is displayed.



PROFIBUS / PROFINET: Telegram editor

Telegram type

○ Receive telegram (Master >> Slave)
● Response telegram (Slave >> Master)

Telegram editor

Response telegram 0

Length 24 Bytes

```
ADR =  1 : Entry = 1500.0 (1 Byte )
ADR =  2 : Entry =  968.0 (2 Bytes)
ADR =  4 : Entry = 1100.0 (4 Bytes)
ADR =  8 : Entry = 1101.0 (4 Bytes)
ADR = 12 : Entry = 1102.0 (4 Bytes)
ADR = 16 : Entry = 1141.0 (4 Bytes)
```

Used Memory: 20 Bytes

Figure 6-4: Composition of a response telegram

Take further descriptions from Chapter 6.2.

The telegram shown in Figure 6-4 transmits the following parameters:

Addres s	Content (parameter number)	Description
0	Identifier (= 0xF0)	Fixed identifier
1	Operating mode (PNU 1500.0)	Current operating mode of the servo drive
2	Status word 1 (PNU 968.0)	Status word for device control, has to be under this address
4	Position actual value (PNU 1100.0)	Current actual position, indication with the physical unit of a position set
8	Velocity actual value (PNU 1101.0)	Current actual speed value, indication with the physical speed unit set
12	Current actual value (PNU 1102.0)	Actual value of the current, indicated per thousand referring to the nominal motor current
16	Digital inputs (PNU 1141.0)	Current status of the digital inputs, see for the meaning of the bits in the detailed description of the PNU.

Detailed information on parameter numbers can be found in chapters 9, 10, 11 and 12.

7 Physical units

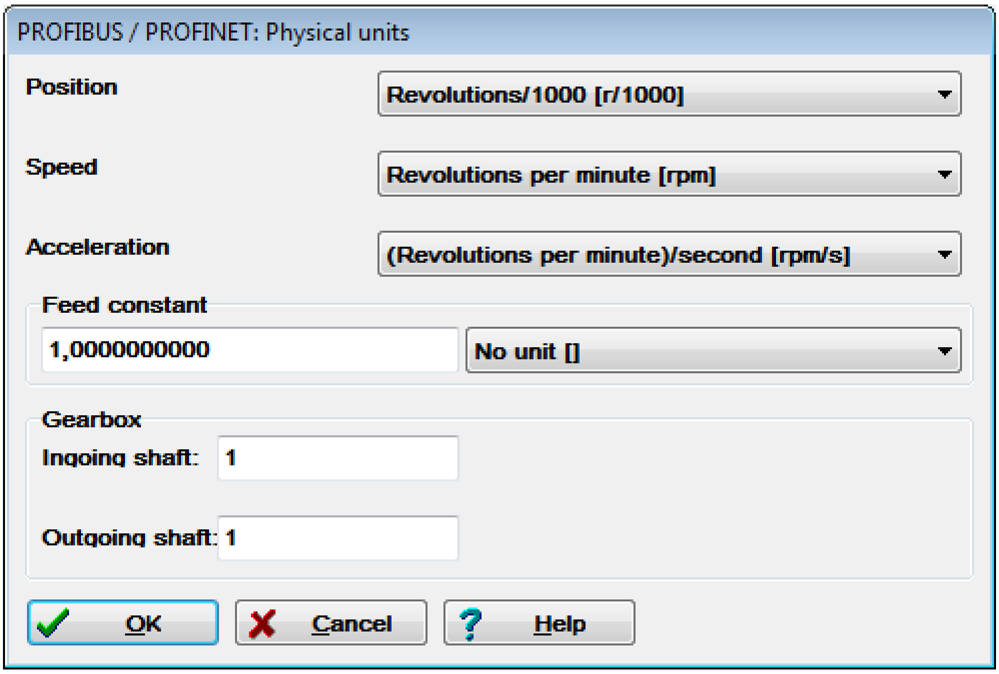
To ensure correct operation, the unit of the process data that will be transferred via the fieldbus must be specified. They can be configured via the menu item **Display units** for PROFIBUS and PROFINET:

- **PROFINET:** The settings can be found on the menu bar under **Parameters - Field bus - PROFINET – Display units**
- **PROFIBUS:** The settings can be found on the menu bar under **Parameters - Field bus - PROFIBUS – Display units**

The parameters for the physical units should be set once and they should not be changed during a running application.

The system calculates specific factors for the various units internally so that all the user has to do is to select the desired unit. The gear ratio and feed constant are stated as separate parameters.

The windows for PROFIBUS and PROFINET are identical. Figure 7-1 shows the window of the Metronix ServoCommander® program for the configuration of the physical units for PROFIBUS.



PROFIBUS / PROFINET: Physical units

Position: Revolutions/1000 [r/1000]

Speed: Revolutions per minute [rpm]

Acceleration: (Revolutions per minute)/second [rpm/s]

Feed constant: 1,0000000000 No unit []

Gearbox

Ingoing shaft: 1

Outgoing shaft: 1

Figure 7-1: Setting of physical units for PROFIBUS

When the physical units are entered, the firmware automatically calculates conversion factors. They consist of a numerator and a denominator which can comprise a maximum of 32 bits each. If an overflow occurs when the factors are entered, the value will be rejected. In this case, the factors or the physical units have to be corrected.

It has to be taken into consideration that some quantities cannot always be used in a sensible manner. A purely rotative system, for example, does not require a feed constant. Apart from this, the feed constant has a physical unit. If this unit is not parameterised correctly, the feed constant will be ignored.

Example:

1. Position in revolutions, feed constant in mm/revolution:
   => The feed constant will be **ignored**.
2. Position in mm, feed constant without a unit:
   => The feed constant will be **considered** like a gear ratio.
3. Position in mm, feed constant in µm/revolution
   => The feed constant will be **considered** with the factor 1000.

The value of the feed constant will be ignored for the associated physical unit, if the feed constant has a translatory unit and a rotatory unit has been selected for the physical quantity.

Problems might arise during operation only if the internal value or the value entered externally cannot be represented following the conversion. In such a case, an error is issued and the setting of the physical units has to be checked.

During the parameterisation of the display units intermediate states may occur in which the physical units have an overflow. This case generates the error 22-4. Perform Save & Reset in order to find out if the parameterisation is really invalid. If no error 22-4 is active after Reset, the settings are valid.

Perform Save & Reset if an error 22-4 occurs during the parameterisation of the physical units. The physical units are valid if there is no error after Reset.

Figure 7-2 shows the interpretation of the gear ratio. In the menu **Physical units PROFIBUS** of the ServoCommander® parameterisation program, the value "Ingoing shaft" refers to U_{EIN}, the value "Outgoing shaft" to U_{AUS}.

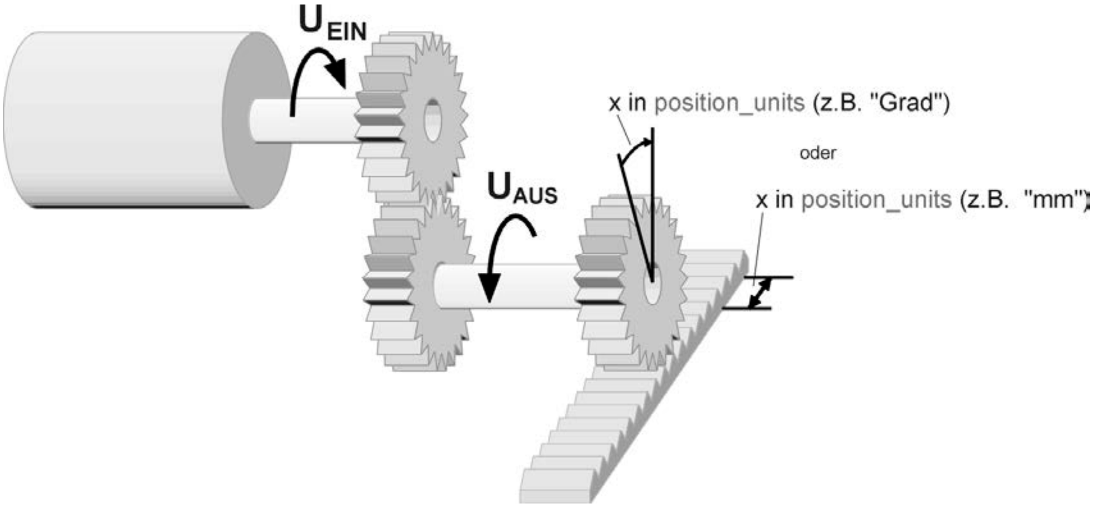

U_{EIN}
U_{AUS}
x in position_units (z.B. "Grad")
oder
x in position_units (z.B. "mm")

Figure 7-2: Gear ratio

Example:

If the motor performs 10 revolutions and a gear connected to the output performs one revolution, this corresponds to the following entries:

Ingoing shaft: 10

Outgoing shaft: 1

Now the system can be parameterised using the units of the output (outgoing shaft).

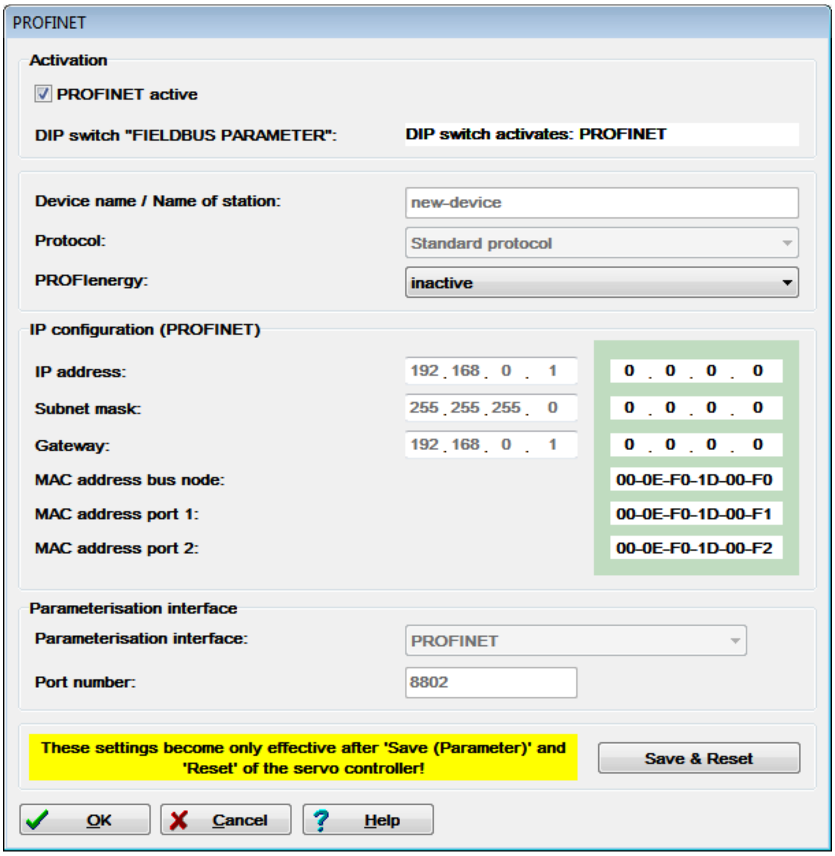
The gear ratio and the feed constant are defined as positive values. If the orientation of the application has to be changed, this can be done with the help of the gear ratio of the angle encoder in the Metronix ServoCommander® software.

8 Operating parameters

This chapter describes all the necessary measures to set up a communication using PROFIBUS-DP or PROFINET-IO. The parameters described in the following sections are set via the serial interface using the Metronix ServoCommander® program.

8.1 PROFINET operating parameters

The operating parameters menu can be opened in the Metronix ServoCommander® via the menu bar under **Parameters - Field bus - PROFINET – Operation parameters**. Figure 8-1 shows the window of the parameterisation program for the configuration of the operating parameters.



PROFINET

Activation

☑ PROFINET active

DIP switch "FIELDBUS PARAMETER": DIP switch activates: PROFINET

Device name / Name of station: new-device

Protocol: Standard protocol

PROFIenergy: inactive

IP configuration (PROFINET)

IP address:	192 . 168 . 0 . 1	0 . 0 . 0 . 0
Subnet mask:	255 . 255 . 255 . 0	0 . 0 . 0 . 0
Gateway:	192 . 168 . 0 . 1	0 . 0 . 0 . 0
MAC address bus node:		00-0E-F0-1D-00-F0
MAC address port 1:		00-0E-F0-1D-00-F1
MAC address port 2:		00-0E-F0-1D-00-F2

Parameterisation interface

Parameterisation interface: PROFINET

Port number: 8802

These settings become only effective after 'Save (Parameter)' and 'Reset' of the servo controller!

Figure 8-1: Settings of the operating parameters under PROFINET-IO

Activation:

The PROFINET communication is activated by way of the check box **PROFINET active** or via the **DIP switch “FIELDBUS PARAMETER”** of the FSM module.

It must be taken into consideration that the PROFINET communication settings will not become effective until after a Save & Reset has been performed. However, the deactivation of the communication becomes immediately effective.

The activation of the PROFINET communication via DIP switch **8** of the FSM module is described in the following table:

DIP switch 8 FSM module	Effect
All DIP switches = OFF	No effect. The configuration for the fieldbus activation will be adopted from the parameter set of the servo drive.
DIP switch 8 = OFF and at least one DIP switch set	Activation via DIP switch: fieldbus inactive
DIP switch 8 = ON	Activation via DIP switch: fieldbus active.

It is always displayed which fieldbus is currently activated/deactivated via the DIP switch.

The DIP switch always acts on the currently present fieldbus whereby the fieldbus modules take precedence over CANopen (onboard). This means:

➔ If DIP switch 8 = ON the currently available fieldbus will be activated.

➔ If a fieldbus module is present, this fieldbus will be activated.

➔ If there is no fieldbus module present, CANopen will be activated by default.

If DIP switch 8 = OFF and at least one other DIP switch is set, all fieldbuses will be deactivated.

Device name:

All that is required for the configuration of the communication on the servo drive side is the device name. The IP address is assigned based on the device name and with the aid of the DCP protocol (Discovery and Basic Configuration protocol). The assignment requires that a DCP-capable controller is included in the network.

Protocol:

Selection of the PROFINET protocol type. At present, only the Metronix **standard protocol** is supported.

PROFIenergy:

This options menu allows to activate or deactivate the standardised energy efficiency protocol PROFIenergy.

IP configuration:

A unique IP address must be assigned to the servo drive. If the addresses are assigned dynamically, the IP address and the associated subnet mask and gateway are assigned via the DCP protocol (based on the **device name**). Any static IP address, which may have been assigned beforehand, will be overwritten.

Parameterisation interface:

The parameterisation or diagnostics of servo drives with the Metronix ServoCommander® can be performed either via the die OnBoard-Ethernet interface (X18) or via the PROFINET network. If the Metronix ServoCommander® is used in the PROFINET network, **ProfiNet** must be configured as the parameterisation interface (see also the next chapter 8.1.1).

Save & Reset:

The settings of the operating parameters will not become effective until the button **Save & Reset** is selected. As a result, the settings will be saved in the parameter set, which is then followed by a reset of the servo drive.

8.1.1 Use of the Metronix ServoCommander® in a PROFINET network

The Metronix ServoCommander® can connect with any servo drive in a PROFINET network provided that the following conditions are fulfilled:

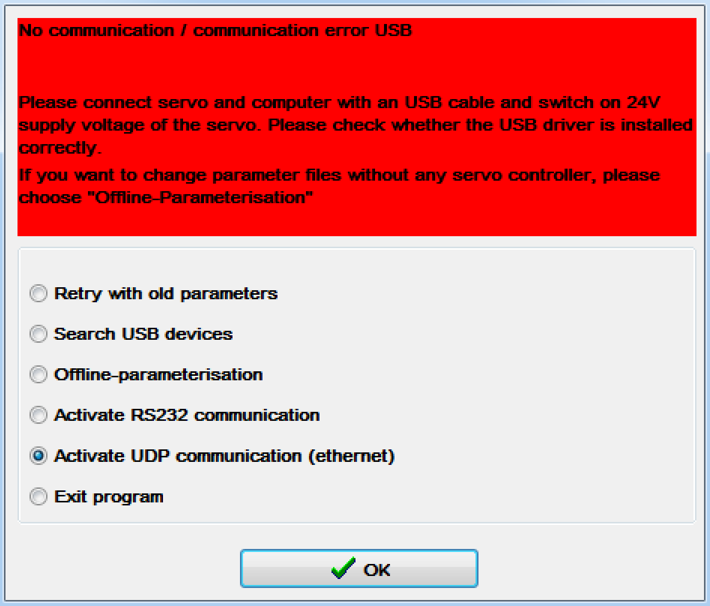
- The PROFINET network has been configured and is ready for use. (An IP address has been assigned to the slave; this is usually done by the controller.)
- The parameterisation interface for the operating parameters (see Figure 8-1) is set to **ProfiNet**.

Depending on whether the IP address is known or not, different steps are required in order to set up a communication between the Metronix ServoCommander® and the ARS 2000.

8.1.1.1 The IP address of the servo drive is known

If the IP address of a servo drive is known, the connection between the Metronix ServoCommander® and a servo drive must be set up as follows:

1. Connect the PC to the PROFINET network by way of an Ethernet cable. For this purpose, the second technology port of a servo drive can be used, for example.
2. Start the Metronix ServoCommander® and select the option **Activate UDP communication (Ethernet)**. Then, click OK to confirm (see Figure 8-2).



No communication / communication error USB

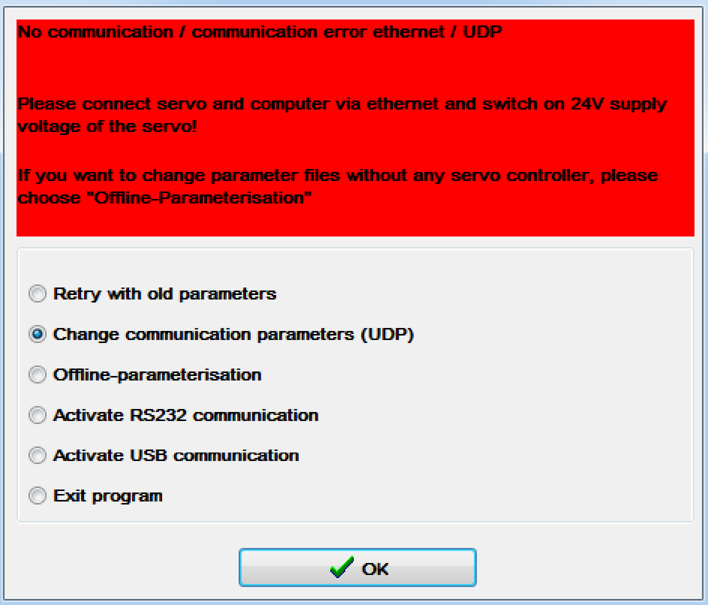
Please connect servo and computer with an USB cable and switch on 24V supply voltage of the servo. Please check whether the USB driver is installed correctly.

If you want to change parameter files without any servo controller, please choose "Offline-Parameterisation"

○ Retry with old parameters
○ Search USB devices
○ Offline-parameterisation
○ Activate RS232 communication
● Activate UDP communication (ethernet)
○ Exit program

Figure 8-2: Selection of the type of communication with the Metronix ServoCommander®

3. Then, the window in Figure 8-3 opens. Select the option **Change communication parameters (UDP)** and click OK to confirm.



No communication / communication error ethernet / UDP

Please connect servo and computer via ethernet and switch on 24V supply voltage of the servo!

If you want to change parameter files without any servo controller, please choose "Offline-Parameterisation"

○ Retry with old parameters

● Change communication parameters (UDP)

○ Offline-parameterisation

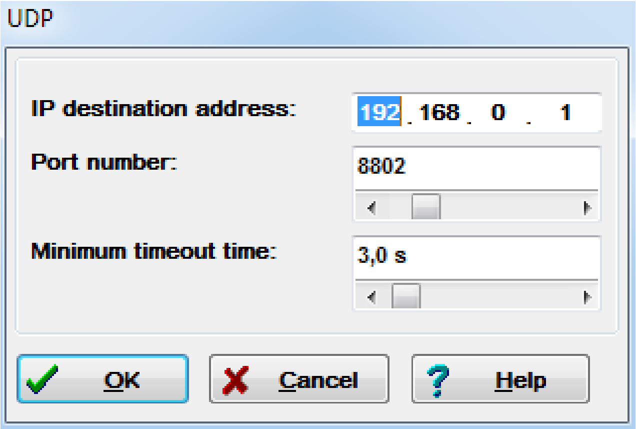
○ Activate RS232 communication

○ Activate USB communication

○ Exit program

Figure 8-3: Selection window after the loss or termination of the UDP connection

4. Enter the IP address of the servo drive to which you want to connect into the UDP window. In order to connect to the servo drive with the settings in our example in Figure 8-1, use the following setting:



UDP

IP destination address: 192 . 168 . 0 . 1

Port number: 8802

Minimum timeout time: 3,0 s

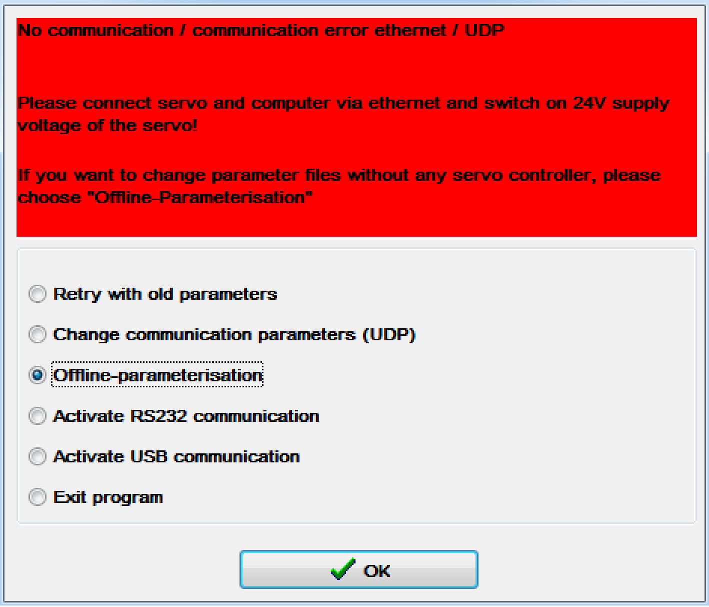
Figure 8-4: Configuration of the UDP connection

5. If the settings are correct, the Metronix ServoCommander® connects itself with the servo drive. In the event of an error, the selection window in Figure 8-3 opens. The configuration can be repeated until a connection to the servo drive is established.

8.1.1.2 The IP address of the servo drive is not known

If the IP addresses of the servo drives in the PROFINET network are unknown, the network devices can be found by way of the Metronix ServoCommander®.

1. Select the option **Offline parameterisation** in the selection window and click OK to confirm.



No communication / communication error ethernet / UDP

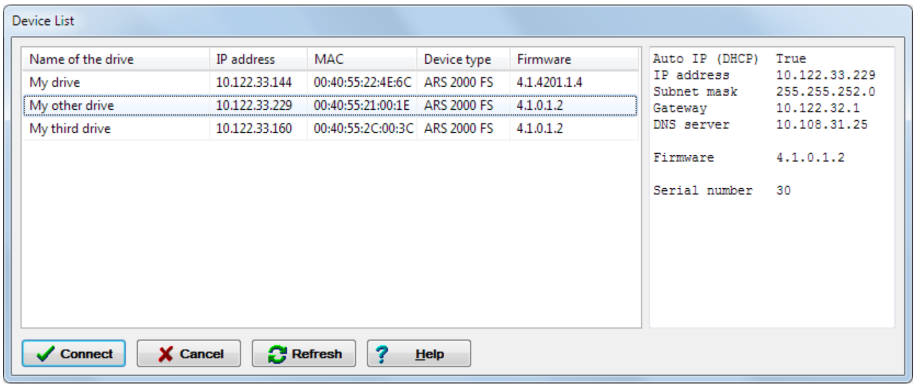
Please connect servo and computer via ethernet and switch on 24V supply voltage of the servo!

If you want to change parameter files without any servo controller, please choose "Offline-Parameterisation"

○ Retry with old parameters
○ Change communication parameters (UDP)
● Offline-parameterisation
○ Activate RS232 communication
○ Activate USB communication
○ Exit program

Figure 8-5: Offline parameterisation

2. As the next step, an offline parameter set must be loaded. Select a valid parameter set and confirm with OK.
3. The Metronix ServoCommander® opens. The servo drives that are included in the network can be found under **Options – Communication – Communication parameters UDP (Ethernet) – Search network…** They are displayed in the **device list** and can be selected by way of the mouse. When the button **Connect** is selected, the Metronix ServoCommander® establishes a connection to the selected servo drive.



Device List

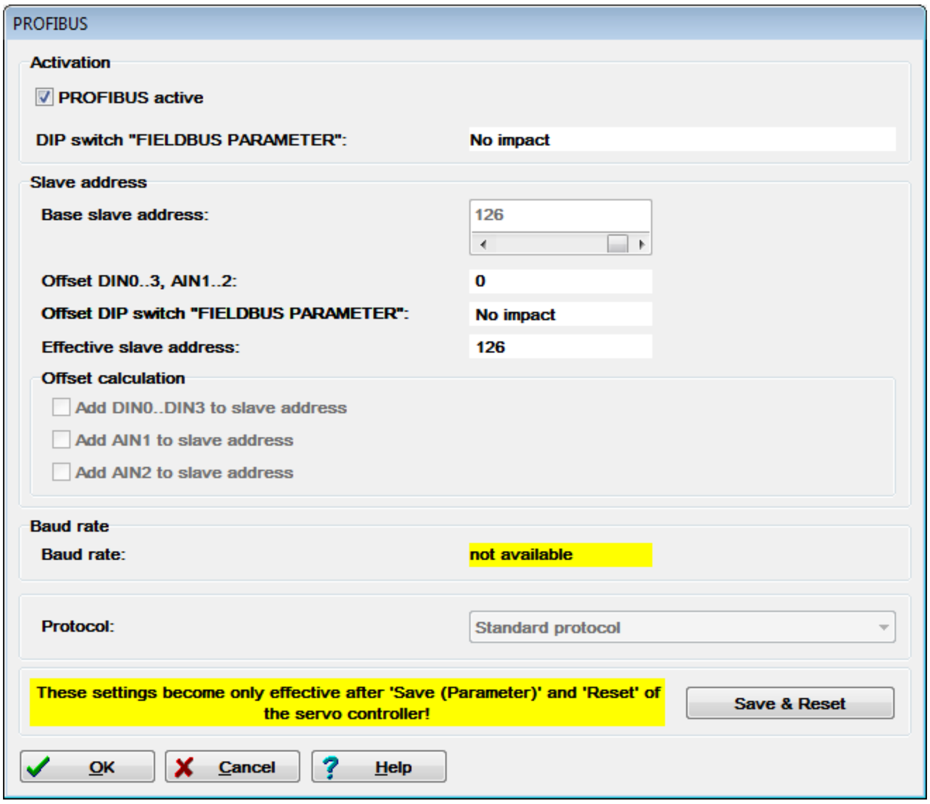
Name of the drive	IP address	MAC	Device type	Firmware
My drive	10.122.33.144	00:40:55:22:4E:6C	ARS 2000 FS	4.1.4201.1.4
My other drive	10.122.33.229	00:40:55:21:00:1E	ARS 2000 FS	4.1.0.1.2
My third drive	10.122.33.160	00:40:55:2C:00:3C	ARS 2000 FS	4.1.0.1.2

```
Auto IP (DHCP)  True
IP address      10.122.33.229
Subnet mask     255.255.252.0
Gateway         10.122.32.1
DNS server      10.108.31.25

Firmware        4.1.0.1.2

Serial number   30
```

Figure 8-6: Metronix ServoCommander® window "Device List"

8.2 PROFIBUS operating parameters

The operating parameters menu can be opened in the Metronix ServoCommander® via the menu bar under **Parameters - Field bus - PROFIBUS – Operation parameters**. Figure 8-7 shows the window of the parameterisation program used to set the operating parameters.



PROFIBUS

Activation

☑ PROFIBUS active

DIP switch "FIELDBUS PARAMETER": No impact

Slave address

Base slave address: 126

Offset DIN0..3, AIN1..2: 0

Offset DIP switch "FIELDBUS PARAMETER": No impact

Effective slave address: 126

Offset calculation

□ Add DIN0..DIN3 to slave address

□ Add AIN1 to slave address

□ Add AIN2 to slave address

Baud rate

Baud rate: not available

Protocol: Standard protocol

These settings become only effective after 'Save (Parameter)' and 'Reset' of the servo controller!

Figure 8-7: Setting of operating parameters under PROFIBUS-DP

<u>Activation:</u>

The PROFIBUS communication is activated by way of the check box **PROFIBUS active** or via the **DIP switch “FIELDBUS PARAMETER”** of the FSM module.

It must be taken into consideration that the PROFBUS communication settings will not become effective until after a Save & Reset has been performed. However, the deactivation of the communication becomes immediately effective.

The activation of the PROFIBUS communication via DIP switch **8** of the FSM module is described in the following table:

DIP switch 8 FSM module	Effect
All DIP switches = OFF	No effect. The configuration for the fieldbus activation will be adopted from the parameter set of the servo drive.
DIP switch 8 = OFF and at least one DIP switch set	Activation via DIP switch: fieldbus inactive
DIP switch 8 = ON	Activation via DIP switch: fieldbus active.

It is always displayed which fieldbus is currently activated/deactivated via the DIP switch.

The DIP switch always acts on the currently present fieldbus whereby the fieldbus modules take precedence over CANopen (onboard). This means:

➔ If DIP switch 8 = ON the currently available fieldbus will be activated.

➔ If a fieldbus module is present, this fieldbus will be activated.

➔ If there is no fieldbus module present, CANopen will be activated by default.

If DIP switch 8 = OFF and at least one other DIP switch is set, all fieldbuses will be deactivated.

Slave address:

At the servo drive end, only the slave address has to be entered to configure the communication. After this, the communication can be activated.

When the communication has been activated, the base value of the slave address can no longer be changed.

Starting at the base value, it is possible to increase the slave address using options and a suitable hardware connection. The digital inputs DIN0..3, the analogue inputs AIN1 and AIN2 and the DIP switches 1 – 7 of the FSM module are intended for this purpose. The states of the inputs are read once immediately after a reset and used to calculate the effective slave address. Later changes have no effect. The options have different values:

Option	Active at	Value
Addition of DIN0..3	+ 24 V DC	0..15
Addition of AIN1	U_on > + 5 V	0, 16
Addition of AIN2	U_on > + 5 V	0, 32

Baudrate:

The baud rate of the PROFIBUS communication is automatically recognised by the hardware used. The automatically detected baud rate is displayed in this menu if a connection between Master and Slave could have been established successfully. The following baud rates are supported by servo drives of the ARS 2000 product range:

Baud rate
9.6 kBaud
19.2 kBaud
45.45 kBaud
93.75 kBaud
187.5 kBaud
500.0 kBaud
1.5 MBaud
3.0 MBaud
6.0 MBaud
12.0 MBaud

Protocol:

Selection of the PROFIBUS protocol type. At present, only the Metronix **standard protocol** is supported.

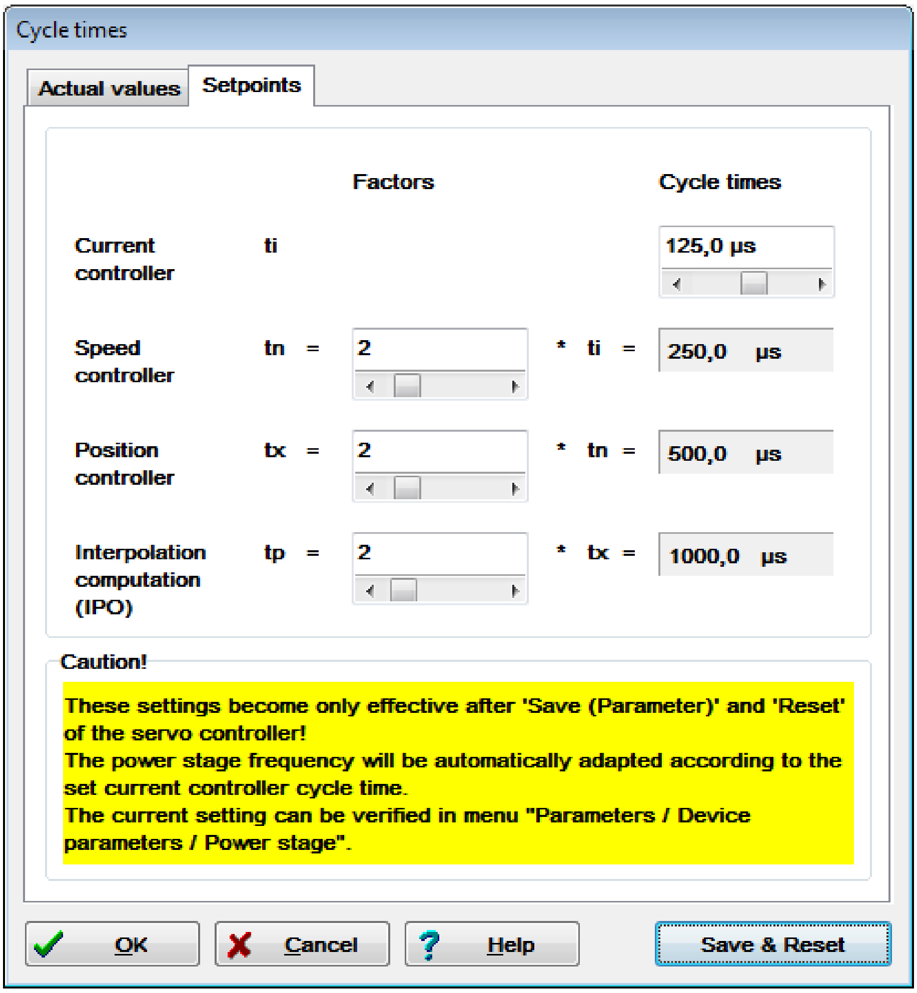
Save & Reset:

The settings of the operating parameters will not become effective until the button **Save & Reset** is selected. As a result, the settings will be saved in the parameter set, which is then followed by a reset of the servo drive.

8.2.1 Adaptation of cycle times

The cycle times of the controller structure of ARS 2000 controllers can be set in a variable manner. If the field bus communication is activated via PROFIBUS, a current controller cycle time of 125 µs is recommended. To set this parameter, open the window **Parameters - Controller parameters - Cycle times...**. Click on the **Settings** button. Now the cycle time of the current controller can be changed. Set the value as shown in Figure 8-8.

Changes concerning the cycle times or the factors do not take effect until they are saved and the system is reset.



			Factors			Cycle times
Current controller	ti					125,0 µs
Speed controller	tn	=	2	*	ti =	250,0 µs
Position controller	tx	=	2	*	tn =	500,0 µs
Interpolation computation (IPO)	tp	=	2	*	tx =	1000,0 µs

Caution!

These settings become only effective after 'Save (Parameter)' and 'Reset' of the servo controller!
The power stage frequency will be automatically adapted according to the set current controller cycle time.
The current setting can be verified in menu "Parameters / Device parameters / Power stage".

Figure 8-8: Parameterisation of the servo drive cycle times

9 Device control

9.1 Overview

To make the control of a field device (slave) independent from the various manufacturers, two data words have been specified in the PROFIdrive specification (4). **Control word 1** is used to control the major device functions by the master whereas the status of the device is read back in s**tatus word 1**. The PROFIdrive profile defines the order in which certain bits have to be set, e.g. to enable the power stage of the servo drive.

Device control of ARS 2000 servo drives follows the state diagram specified in the PROFIdrive profile. The implementation and potential manufacturer-specific deviations are described in detail in chapter 9.4 State diagram and device control.

The implementation of the control word and the status word only follows the PROFIdrive specification. Deviations from the specifications are marked. In addition, some bits have different meanings depending on the operating mode. In the following chapters, first the control word and then the status word are described. Then the device control mechanism using these two data words is explained.

9.2 Control word 1

Control Word 1 is used to control various device functions, e.g. servo drive enabling (hereinafter referred to as “controller enabling”). Some bits have corresponding meanings. The use of these bits is described in chapter 9.4 State diagram and device control. The meaning of the individual bits follows the PROFIdrive profile. In addition, some functions are a manufacturer-specific and some bits have a manufacturer-specific functionality.

PNU	**967**
Subindex	**0**
Name	Control word 1
Data type	UINT16
Access	rw
Unit	-
Value range	-
Default value	0

In received telegrams 0..2, control word 1 is bound to a fixed position. The system evaluates the control word always as the last datum. This means that new target positions are written first. A command to start a positioning run, which is transmitted simultaneously, therefore always refers to the data transmitted in the same telegram.

Some bits have different meanings depending on the operating mode. The two following tables show the meanings for the two operating modes.

Bit	Meaning	Behaviour
0	ON / OFF (OFF 1)	See the “Device control” section, chapter 9.4
1	No coast stop (no OFF 2) / coast stop (OFF 2)	
2	No quick stop (no OFF 3) / quick stop (OFF 3)	
3	Enable Operation / Disable Operation	
4*	1: Enable ramp generator 0: Reset ramp generator	1: All speed setpoints enabled 0: All speed setpoints disabled
5*	1: Unfreeze ramp generator 0: Freeze ramp generator	1: Setpoint ramp enabled 0: Ramp stopped (frozen)
6*	1: Enable setpoint 0: Disable setpoint	1: All setpoint inputs for the ramp enabled 0: All setpoint inputs for the ramp deactivated
7	Fault acknowledge (edge from 0->1)	Active errors are acknowledged if possible
8*	Jog 1 ON / Jog 1 OFF	The speed of motion of the “positive jogging” position set is preset as a speed setpoint through the ramp
9*	Jog 2 ON / Jog 2 OFF	The speed of motion of the “negative jogging” position set is preset as a speed setpoint through the ramp
10	1: Control by PLC 0: No control by PLC	1: Control word evaluated 0: Control word not evaluated
11*	Device-specific	free
12-15	Device-specific	free

*: Different meaning in another operating mode.

Table 9-1: Control word 1 for speed control mode

Meaning of/notes concerning the various bits

Bit 4: Corresponds to the hold input (hold function) of FB43 if the Metronix function blocks are used. If the input is set (logic 1 at the input of the FB → bit 4 = 0), the drive will decelerate with the set deceleration rate. If the input is reset (logic 0 at the input of the FB → bit 4 = 1), the motor will accelerate with the set acceleration rate up to the set speed value.

Bit 5: Setting bit 5 to logic 0 freezes the current setpoint setting.

Example: - Current setpoint setting 200 rpm (bit 5 = 1)
- Deletion of bit 5 (bit 5 = 0)
→ If now bit 4 is deleted, for example (→ the drive decelerates with the set deceleration rate), there will be no reaction. The drive continues to run with the last specified speed value (in our example with 200 rpm).

Bit 6: If bit 6 is reset (bit 6 = 0), the setpoint inputs for the ramps will be deactivated. This means that the drive will be stopped to a standstill as quickly as possible (if physically possible even quicker than the deceleration for an emergency stop → the current setpoint setting will be set to 0 without any ramp at all). The setting of the bit (bit 6 = 1), on the other hand, causes the motor to accelerate as quickly as possible to the set speed value (abruptly, without any ramps).
Bit 5 has no effect on the function of bit 6.

When the servo drive is disabled, the drive is stopped to a standstill with the set deceleration for an emergency stop. When the servo drive is enabled, the motor accelerates with the set acceleration rate up to the speed setpoint.

Bit	Meaning	Behaviour
0	ON / OFF (OFF 1)	See the “Device control” section, chapter 9.4
1	No coast stop (no OFF 2) / coast stop (OFF 2)	
2	No quick stop (no OFF 3) / quick stop (OFF 3)	
3	Enable Operation / Disable Operation	
4*	1: Do not reject traversing task 0: Reject traversing task	0: The current positioning run is cancelled or no positioning is started 1: No action
5*	1: No intermediate stop 0: Intermediate stop	0: No action or no positioning is started 0 ->1: Acceleration to motion speed in accordance with the current position set 1: No action 1 -> 0: Stopping with deceleration in accordance with current position set
6*	0 -> 1: Activate traversing task ***	Manufacturer-specific implementation: 0 -> 1: Start of the positioning run under the set position data set[1] if boundary conditions[2] are fulfilled
7	Fault acknowledge (edge from 0->1)	Active errors are acknowledged if possible
8*	Jog 1 ON / Jog 1 OFF	Manufacturer-specific implementation: 0 -> 1: Start of positioning in accordance with “positive jogging” position set 1 -> 0: Stop with deceleration in accordance with “positive jogging” position set
9*	Jog 2 ON / Jog 2 OFF	Manufacturer-specific implementation: 0 -> 1: Start of positioning in accordance with “negative jogging” position set 1 -> 0: Stop with deceleration in accordance with “negative jogging” position set
10	1: Control by PLC 0: No control by PLC	1: Control word evaluated 0: Control word not evaluated
11*	Start / Stop homing procedure	1: (No action) Continue homing 1 -> 0: Homing still active: Termination of homing without error Homing already terminated: No action 0: No action 0 -> 1: Start of homing[3]
12**	Relative / absolute	At positioning start: 1: Relative positioning 0: Absolute positioning
13**	Interrupt current positioning / add	At positioning start: 1: Interrupt current positioning, immediately start new positioning. 0: Add positioning to the end of the running positioning
14,15	Device-specific	

*: Different meaning in another operating mode.
**: Manufacturer-specific bit
***: Manufacturer-specific deviation from PROFIdrive specification

Table 9-2: Control word 1 for positioning mode

Notes concerning Table 9-2:

[1]: The command “Activate traversing” starts the position set selected through PNU 1002 0. If the PROFIBUS position set is started, the current options of control word 1 will be adopted. Otherwise the options of the corresponding position set will take effect.

[2]: The following boundary conditions apply to the start of a positioning:

- Bit 4 = 1 (Do not reject traversing task)
- Bit 5 = 1 (No intermediate stop)
- No homing active

[3]: Parameterisable options are taken into account, e.g. “with following positioning”.

Chapter 9.4 describes the device control. The servo drive takes on various states. Defined transitions can be performed between these states. The transitions are triggered by so-called commands through bits 0..3. The commands are described in detail in chapter 9.4. The following tables shows an overview of the commands:

Command:	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**State transitions**
	0008_h	**0004_h**	**0002_h**	**0001_h**	
OFF	×	**1**	**1**	**0**	1, 5, 11
ON	×	**1**	**1**	**1**	2
Coast Stop	×	×	**0**	×	6, 7, 8
Quick Stop	×	**0**	**1**	×	9, 10, 12
Disable Operation	**0**	**1**	**1**	**1**	4
Enable Operation	**1**	**1**	**1**	**1**	3

(× = not relevant)

Table 9-3: Overview of all commands

As some state transitions take a certain time, all status changes that are triggered through **control word 1** have to be read back through **status word 1**. It is only when the requested status can also be read in **status word 1**, that another command can be written in through **control word 1**.

9.3 Status word 1

Status word 1 reflects various device states, e.g. an active controller enabling. Some bits have corresponding meanings. All this is described in chapter 9.4 State diagram and device control. The meaning of the bits follows the PROFIdrive profile. In addition, some functions are manufacturer-specific and some bits have a manufacturer-specific functionality.

PNU	**968**
Subindex	**0**
Name	Status word 1
Data type	UINT16
Access	ro
Unit	-
Value range	-
Default value	-

In response telegrams 0..2, status word 1 is bound to a fixed position.

Some bits have different meanings depending on the operating mode. The two following tables show the meaning for the two operating modes.

Bit	Meaning	Behaviour
0	1: Ready To Switch On 0: Not Ready To Switch On	See the “Device control” section, chapter 9.4
1	1: Ready To Operate 0: Not Ready To Operate	
2	1: Operation Enabled 0: Operation Disabled	
3	1: Fault Present 0: No Fault Present	1: Active faults 0: No fault active
4	1: No OFF2 0: OFF2	1: No OFF2 command active 0: OFF2 command (control word 1, power stage OFF) active
5	1: No OFF3 0: OFF3	1: No OFF3 command active 0: OFF3 command (control word 1, quick stop) active
6	1: Switching On Inhibited 0: Switching On Not Inhibited	See the “Device control” section
7	1: Warning Present 0: No Warning Present	1: Active warning and/or setpoint disabling through limit switch active in at least one direction of rotation 0: No warning active
8*	1: Speed error within tolerance range 0: Speed error out of tolerance range	1: The actual speed is within the parameterisable signal window around the target speed 0: The actual speed is out of the parameterisable signal window around the target speed
9	1: Control by PLC 0: No control by PLC	Mirroring of bit 10 of control word 1
10*	1: f or n reached 0: f or n not reached	1: Actual speed > freely parameterisable reference speed[1)] 0: Actual speed < freely parameterisable reference speed[1)]
11-13*	Device-specific	Free
14-15	Device-specific	Free

*: Different meaning in another operating mode.

Table 9-4: Status word 1 for speed control mode

[1)]: For this comparison, the sign character is always taken into consideration, i.e. the comparison is not based on the sum of the actual speed and the reference speed.

Meaning of/notes concerning the various bits

Bit 8: The parameterisable **signal window** can be configured with the Metronix ServoCommander® under **Parameters\Signals…\Speed signal**.

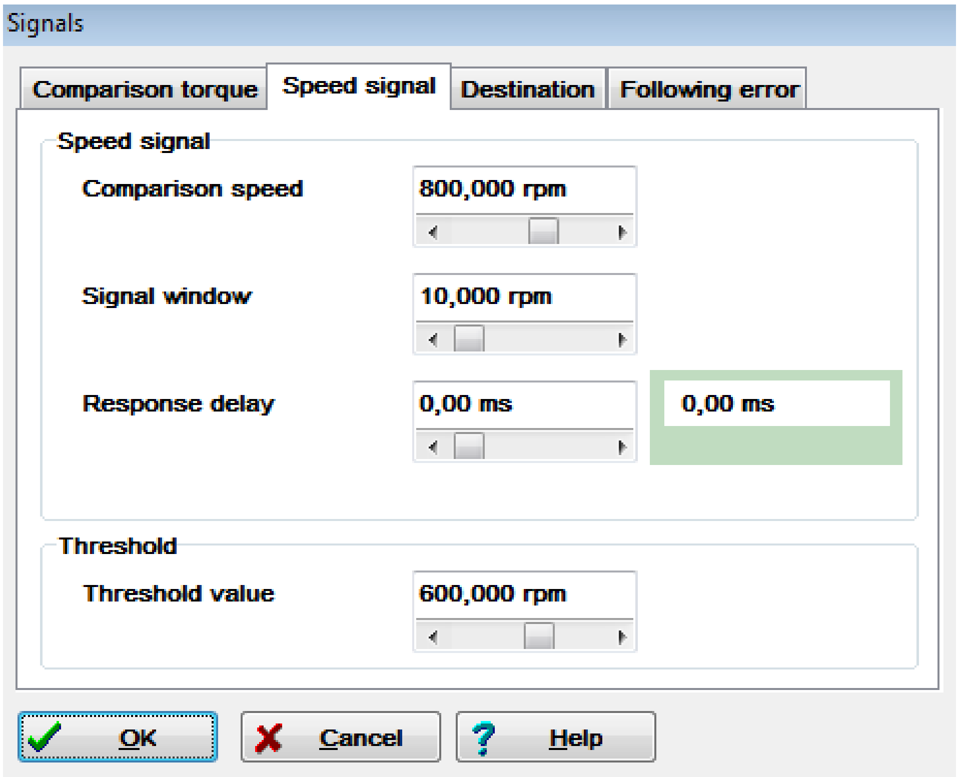
Comparison speed: This value corresponds to the set value (setpoint setting). If, for example, a speed setpoint of 800,000 rpm is specified via the PB, this setpoint also corresponds to the comparison speed (identical values).

Signal window (see Figure 9-1):

The output will be active when the current speed is in the range of (Comparison speed - Signal window …. Comparison speed + Signal window).

Note: Since the actual speed value is subject to fluctuations, this signal may fluctuate if the parameterisation of the signal window is too narrow.

Response delay: This time value defines how long the drive must be within the set signal window before bit 8 will be set. If the drive leaves the set signalling window (even very briefly), the set response delay will be restarted (and bit 8 will be reset).



Speed signal

Comparison speed 800,000 rpm

Signal window 10,000 rpm

Response delay 0,00 ms 0,00 ms

Threshold

Threshold value 600,000 rpm

Figure 9-1: Speed signal

Bit 10: The freely parameterisable threshold value can be configured with the Metronix ServoCommander® under Parameters\Signals…\Speed signal.

In addition to the comparison speed (bit 8), the user can define a second value (threshold value). Bit 10 will be set once this value is reached (or exceeded).

Bit	Meaning	Behaviour
0	1: Ready To Switch On 0: Not Ready To Switch On	See the “Device control” section, chapter 9.4
1	1: Ready To Operate 0: Not Ready To Operate	
2	1: Operation Enabled 0: Operation Disabled	1: Controller enabling active 0: Controller enabling not active
3	1: Fault Present 0: No Fault Present	1: Active faults 0: No fault active
4	1: No OFF2 0: OFF2	1: No OFF2 command active 0: OFF2 command (control word 1, power stage OFF) active
5	1: No OFF3 0: OFF3	1: No OFF3 command active 0: OFF3 command (control word 1, quick stop) active
6	1: Switching On Inhibited 0: Switching On Not Inhibited	See the device control section
7	1: Warning Present 0: No Warning Present	1: Active warning and/or setpoint disabling through limit switch active in at least one direction of rotation 0: No warning active
8*	1: Following error within tolerance range 0: Following error out of tolerance range	1: No following error 0: Following error message active
9	1: Control by PLC 0: No control by PLC	Mirroring of bit 10 of control word 1
10*	1: Target position reached 0: Not at target position	1: The current positioning run has been completed and the actual position is within the target window 0: The current positioning run is still active or the actual position is not within the target window.
11*	1: Home position valid 0: Home position not valid	1: A homing run has been completed successfully 0: No homing run has been performed or the position information is invalid due to an error.
12*	Traversing Task Acknowledge	See detailed description
13*	1: Drive stopped 0: Drive in motion	1: The actual speed is within a fixed tolerance window around 0. No active positioning or active quick stop. 0: The actual speed is out of the tolerance window around 0 or the condition above is not fulfilled.
14-15	Device-specific	Free

*: Different meaning in another operating mode.

Table 9-5: Status word 1 for positioning mode

Note: The signal "home_valid" sometimes depends on the encoder system that is used.
If an absolute encoder is used, this signal is set to logic 1 even without the execution of a homing run.

Reason: The signal "home_valid" refers to the drive system (servo drive, encoder, and motor). Thus, from the point of view of the servo drive, a homing run is not necessary, since the precise position of the drive is known.

Just like the combination of several bits of **control word 1** can trigger several state transitions, the combination of several bits of **status word 1** can indicate the state of the servo drive. The following table shows the possible states of the state diagram and the associated bit combination used to indicate the states in **status word 1**.

Status	Bit 6	Bit 2	Bit 1	Bit 0	Mask	Value
	0040_h	0004_h	0002_h	0001_h		
SWITCH_ON_INHIBITED	**1**	**0**	**0**	**0**	**0047_h**	**0040_h**
READY_FOR_SWITCHING_ON	**0**	**0**	**0**	**1**	**0047_h**	**0001_h**
SWITCHED_ON	**0**	**0**	**1**	**1**	**0047_h**	**0003_h**

Table 9-6: Device status

Bits 4 and 5 depend on the command and therefore are not included in Table 9-6.

9.4 State diagram and device control

This chapter describes how ARS 2000 servo drives are controlled with the help of the two data words "control word 1" (PNU 967) and "status word 1" (PNU 968), i.e. how the power stage is activated, for example. This control mechanism follows the specification of the PROFIdrive profile. The following terms are used for the explanation of the context:

State: Depending on whether the power stage has been activated, for example, or whether a fault has occurred, the servo drive is in different states. The states defined under PROFIdrive will be described in this chapter.

Example: **SWITCHING_ON_INHIBITED**

State transition Just like the states, also the transitions between the states are defined under PROFIdrive. Transition means getting from one state to another. The master triggers state transitions by setting bits in **control word 1**. The servo drive can also trigger a state transition internally when it detects a fault.

Command To trigger state transitions, certain combinations of bits have to be set in **control word 1**. Such a combination is referred to as a command.

Example: **Enable Operation**

State diagram The states and the state transitions form the state diagram, i.e. an overview of all states and possible transitions.

9.4.1 State diagram

The states have been taken over to a large extent from the PROFIdrive specification. PROFIdrive distinguishes between a ramp stop and a quick stop. The controller enabling is deactivated in both cases so that the simplified state diagram shown in Figure 9-2 results.

After it has been switched on, the servo drive initialises and finally reaches the **SWITCHING_ON_INHIBITED** state. The power stage is deactivated and the motor shaft can be rotated freely. Via state transitions **1**, **2** and **3**, the **OPERATION** state is reached. This state corresponds to the controller enabling through PROFIBUS. In this state, the power stage is activated and the drive is controlled as defined by the operating mode set. Make sure that the servo drive has been parameterised correctly and that an associated setpoint equals zero.

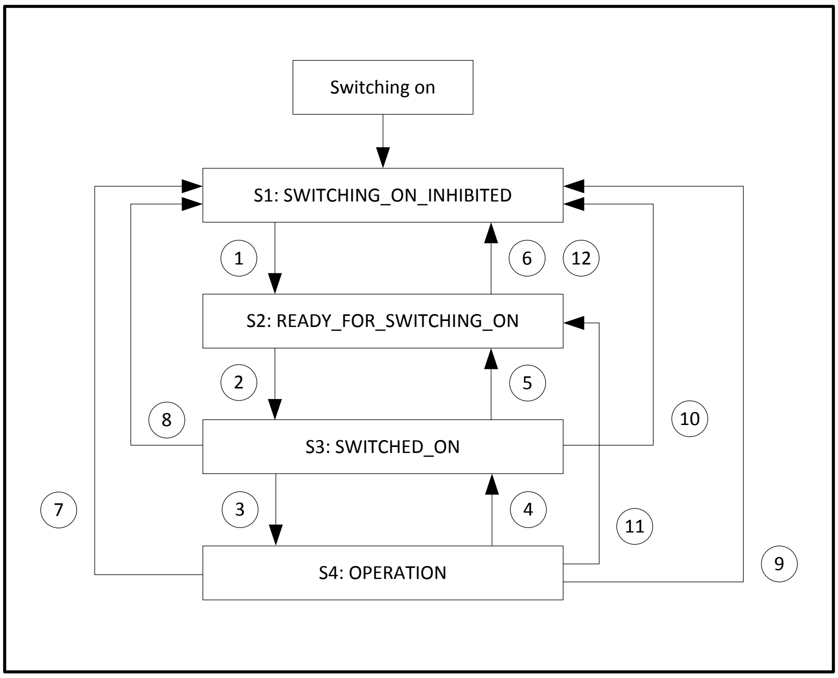


Figure 9-2: Simplified state diagram

State transition **4** corresponds to the cancellation of the controller enabling, i.e. a running motor is slowed down to standstill following a set emergency stop ramp. State transition **7** corresponds to the cancellation of the power stage enabling, i.e. a running motor would coast down in an uncontrolled manner.

If a fault occurs (no matter in which state), the system is led to the **SWITCHING_ON_INHIBITED** state. Depending on the severity of the fault, certain actions, e.g. an emergency stop, can be performed before this state is reached.

The states and their meanings are listed in the following table:

Name	Meaning
Switching on	The servo drive performs a self-test. The PROFIBUS/PROFINET communication does not work yet.
SWITCHING_ON_INHIBITED	The servo drive has completed its self-test. PROFIBUS/PROFINET communication is possible.
READY_FOR_SWITCHING_ON	The servo drive waits until the digital inputs “power stage enable" and "controller enable" are connected to 24 V. (Controller enabling logic „DIn5 and PROFIBUS“).
SWITCHED_ON	Power stage enabling is active.
OPERATION	The motor is connected to power and is controlled as defined by the operating mode.

9.4.2 Device control

To perform the state transitions described in chapter 0, certain bit combinations have to be set in **control word 1** (see below). The lowest 4 bits of **control word 1** are evaluated together to trigger a state transition. In the following section, only the most important state transitions 1, 2, 3, 4, 7 and 11 are explained. A table showing all possible states and state transitions can be found in chapter 0. The first column of the following table shows the desired state transition, and the second column the requirements which have to be fulfilled for this transition (in most cases a command coming from the host, indicated by a frame in the table). The creation of this command, i.e. the bits which have to be set in **control word 1**, is shown in the third column (x = not relevant). Bit 10 in **control word 1** has always to be set to control the servo drive. After the state transition, the new state can be recognised by evaluating the relevant bits in **status word 1.** This is shown in the last column.

No.	**Will be performed if**	**Bit combination for control word 1**					**Action**	**Status word 1**[1)]
		Bit	**3**	**2**	**1**	**0**		
1	Power stage and controller enabling present + no coast stop + no quick stop + command **OFF**	**OFF** =	**x**	**1**	**1**	**0**	None	0x0201
2	Command **ON**	**ON** =	**0**	**1**	**1**	**1**	Activation of power stage enabling	0x0203
3	Command **Enable Operation**	**Enable Operation** =	**1**	**1**	**1**	**1**	Control in accordance with operating mode set	0x0207
4	Command **Disable Operation**	**Disable Operation** =	**0**	**1**	**1**	**1**	Cancellation of controller enabling	0x0203
11	Command **OFF**	**OFF** =	**x**	**1**	**1**	**0**	Cancellation of controller enabling	0x0201
7	Command **Coast Stop**	**Coast Stop** =	**x**	**x**	**0**	**x**	Power stage is disabled. The motor coasts down and can be rotated freely.	0x0250 or 0x0270

[1)]: After completion of the state transition. The mask for the relevant bits is 0x0277
(x = not relevant)

Table 9-7: Important state transitions of the servo drive

The following example shows the enabling of the servo drive via the PROFIBUS/PROFINET fieldbus:

EXAMPLE

The servo drive is to be “enabled”, i.e. the power stage and the controller enabling are to be activated via PROFIBUS:

1.) The servo drive is in the **SWITCH_ON_INHIBITED** state

2.) The servo drive has to change over to the **OPERATION** state

3.) According to the state diagram (Figure 9-2), state transitions 1, 2 and 3 have to be performed.

4.) The following results from Table 9-7:

Transition 1:	**Control Word 1 = 0406_h** **Status word 1** **previous = 0x0240h**	**New state: READY_FOR_SWITCHING_ON *1)** **Status word 1** **expected = 0x0201h**
Transition 2:	**Control Word 1 = 0407_h** **Status word 1** **previous = 0x0201h**	**New state: SWITCHED_ON *1)** **Status word 1** **expected = 0x0203h**
Transition 3:	**Control Word 1 = $040F_h$** **Status word 1** **previous = 0x0203h**	**New state: OPERATION *1)** **Status word 1** **expected = 0x0207h**

Note:

1.) The example assumes that no other bits are set in **control word 1.** Bit 10 must be set. Otherwise, only bits 0..3 are relevant for the transitions.

*1) The master has to wait until the state can be read back in **status word 1** in the relevant bits (mask = 0x0277). This will be explained in detail later in this manual.

9.4.3 Command overview

The following table shows the commands for the state transitions described in chapter 0:

No.	Will be performed if	Bit combination for control word 1					Action	Status word 1[1)]
		Bit	3	2	1	0		
1	Power stage and controller enabling present + no Coast Stop + no Quick Stop + command **OFF**	**OFF** =	x	1	1	0	None	0x0201
2	Command **ON**	**ON** =	0	1	1	1	Activation of power stage enabling	0x0203
3	Command **Enable Operation**	**Enable Operation** =	1	1	1	1	Control in accordance with adjusted operating mode	0x0207
4	Command **Disable Operation**	**Disable Operation** =	0	1	1	1	Cancellation of controller enabling	0x0203
5	Command **OFF**	**OFF** =	x	1	1	0	Cancellation of controller enabling	0x0201
6	Command **Coast Stop**	**Coast Stop** =	x	x	0	x	None	0x0250 resp. 0x0270
7	Command **Coast Stop**	**Coast Stop** =	x	x	0	x	The power stage is disabled. The motor coasts down and can be rotated freely.	0x0250 resp. 0x0270
8	Command **Coast Stop**	**Coast Stop** =	x	x	0	x	Cancellation of power stage enabling	0x0250 resp. 0x0270
9	Command **Quick Stop**	**Quick Stop** =	x	0	1	x	Cancellation of controller enabling	0x0260
10	Command **Quick Stop**	**Quick Stop** =	x	0	1	x	Cancellation of controller enabling	0x0260
11	Command **OFF**	**OFF** =	x	1	1	0	Cancellation of controller enabling	0x0201

No.	Will be performed if	Bit combination for control word 1					Action	Status word 1[1)]
		Bit	3	2	1	0		
12	Command Quick Stop	Quick Stop =	**x**	**0**	**1**	**x**	Cancellation of controller enabling	0x0260

[1)]: After completion of the state transition. The mask for the relevant bits is 0x0277

Table 9-8: Overview of all state transitions of the servo drive

Power stage disabled...

...means that the power semiconductors (transistors) are no longer activated. **If this state is assumed while the motor is running, the motor will coast down.** A motor brake which might be present, will be put on automatically.

Caution: The signal does not guarantee that the motor is really disconnected from power.

Power stage enabled...

...means that the motor is controlled as defined by the selected operating mode. A motor brake which might be present will be released automatically. In the event of a defect or incorrect parameterisation (motor current, number of poles, encoder offset angle, etc.) the drive might react in an uncontrolled manner.

10 Manufacturer-specific parameter numbers

10.1 Overview

The following table provides an overview of currently implemented PNUs:

PNU	Sub-index	Description	Type	Access
1000	0	Position set number (read/write)	UINT16	rw
1001	-	(Position data)		
	0	Target position	INT32	rw
	1	Profile velocity	INT32	rw
	2	End velocity	INT32	rw
	3	Acceleration (positioning)	UINT32	rw
	4	Deceleration (positioning)	UINT32	rw
	5	Acceleration and deceleration (positioning)	UINT32	rw
1002	0	Start set number	UINT8	rw
1003	0	Profile type of the position sets	INT16	rw
1004	0	Thread Speed	UINT16	rw
1005	-	(Software position limits)		
	0	Lower software limit switch	INT32	rw
	1	Upper software limit switch	INT32	rw
1006	-	(Rotary Axis)		
	0	Rotary axis mode	UINT8	rw
	1	Lower rotary axis limit	INT32	rw
	2	Upper rotary axis limit	INT32	rw
1010	0	Target velocity	INT32	rw
1011	-	(Accelerations for velocity control)		
	0	Acceleration (velocity control)	UINT32	rw
	1	Deceleration (velocity control)	UINT32	rw
	2	All accelerations (velocity control)	UINT32	rw

PNU	Sub-index	Description	Type	Access
1040	-	(Jogging)		
	0	Jogging velocities (symmetrical)	INT32	rw
	1	Jogging accelerations (symmetrical)	UINT32	rw
1041	-	(Jogging positive)		
	0	Jogging velocity positive	INT32	rw
	1	Jogging acceleration positive	UINT32	rw
	2	Jogging deceleration positive	UINT32	rw
	3	Jogging accelerations positive	UINT32	rw
1042	-	(Jogging negative)		
	0	Jogging velocity negative	INT32	rw
	1	Jogging acceleration negative	UINT32	rw
	2	Jogging deceleration negative	UINT32	rw
	3	Jogging accelerations negative	UINT32	rw
1050	0	Homing method	INT8	rw
1051	0	Homing offset	INT32	rw
1060	0	Thread speed	INT32	rw
1100	0	Position actual value	INT32	ro
1101	0	Velocity actual value	INT32	ro
1102	0	Current actual value	INT32	ro
1110	-	(Sampling positions)		
	0	Sampled actual position on rising edge	INT32	ro
	1	Sampled actual position on falling edge	INT32	ro
1141	0	Digital inputs	UINT32	ro
1270	-	(Position control parameters)		
	2	Position error tolerance window	UINT32	rw
1271	-	(Position window data)		
	0	Window for „target reached“ message	UINT32	rw
1272	-	(Following error data)		
	0	Following error window	UINT32	rw
1273	-	(Position error data)		
	0	Following error limit	UINT32	rw
1500	0	Operating mode	UINT8	ro
1510	-	(CAM Control)		
	0	Control CAM disks	UINT32	rw
	1	Control axis error compensations	UINT32	rw
1600	0	Last error number	UINT16	ro
2000	0	Entry for manufacturer specific PKW access	2 UINT32	rw

PNU	Sub-index	Description	Type	Access
2010	-	(Placeholder)		
	0	8-bit placeholder (blank element)	UINT8	rw
	1	16-bit placeholder (blank element)	UINT16	rw
	2	32-bit placeholder (blank element)	UINT32	rw
2011	0	32-bit placeholder (blank element)	UINT32	rw

10.2 PNUs for positioning mode

This section describes the parameters required for the positioning mode.

10.2.1 PNU 1000: Position Set Number

This parameter is used to select the position data set into which the data transmitted through PROFIBUS/PROFINET are entered. This parameter provides access to all position data sets of the servo drive. Field bus data sets are often volatile data sets. The position data set for PROFIBUS/PROFINET, however, can be saved and parameterised with the help of the Metronix ServoCommander® parameterisation program. In this way, parameters, which do not have to be changed in an application during operation, can be defined in a fixed manner. Accelerations, for example, can be entered once and then do not have to be transmitted.

This parameter also provides access to special position data sets for homing or jogging. Due to the special structure of the data sets, parameterisation using the ServoCommander® program is recommended.

PNU	**1000**
Subindex	**0**
Name	Position Set Number
Data type	UINT16
Access	rw
Unit	-
Value range	0 .. 267 0..255: Standard position data sets 256: Homing phase 0 257: Homing phase 1 258: Homing phase 2 259: Positive jogging 260: Negative jogging 261..265: Reserved 266: PROFIBUS/PROFINET position data set 267: Reserved
Default value	266 (PROFIBUS/PROFINET position set)

10.2.2 PNU 1002: Start Set Number

This parameter is used to select the position data set which is started through control word 1 following a positioning start command. The servo drive has 256 storable standard position data sets. These data sets can be selected unambiguously on the basis of 8 bits.

Apart from these 256 sets, only the PROFIBUS/PROFINET position data set is of interest for starting. To keep the number of data to be transmitted on the bus as small as possible, the PROFIBUS/PROFINET position data set is addressed under the last index. This means that position data set number 255 cannot be started on the bus.

PNU	**1002**
Subindex	**0**
Name	Start Set Number
Data type	UINT8
Access	rw
Unit	-
Value range	0 .. 255 0..254: Standard position data sets 255: PROFIBUS/PROFINET position data set
Default value	255 (PROFIBUS/PROFINET position set)

10.2.3 PNU 1001: Position Data

This parameter number is used to address parameters of the selected position data set. PNU 1000 is used for selection. The following parameters are available:

- Target position
- Profile velocity
- End velocity
- Acceleration and deceleration, either separately or combined for both accelerations

The data are interpreted in the way they were set as physical units. For this purpose, the Metronix ServoCommander® parameterisation software opens a special window (see chapter 7 Physical units).

This PNU can also be used to parameterise parameters for jogging, for example. To do so, the position data set number has to be set accordingly. Then the jogging speed can be set with the help of the profile velocity.

PNU	**1001**
Name	Position Data

Subindex	**0**
Name	Target Position
Data type	INT32
Access	rw
Unit	Physical unit of position
Value range	-
Default value	0

Subindex	**1**
Name	Profile Velocity
Data type	INT32
Access	rw
Unit	Physical unit of velocity

Value range	-
Default value	1000 rpm

Subindex	**2**
Name	End Velocity
Data type	INT32
Access	rw
Unit	Physical unit of velocity
Value range	-
Default value	0

Subindex	**3**
Name	Acceleration Positioning
Data type	UINT32
Access	rw
Unit	Physical unit of acceleration
Value range	-
Default value	10,000 (rpm)/s

Subindex	**4**
Name	Deceleration Positioning
Data type	UINT32
Access	rw
Unit	Physical unit of acceleration
Value range	-
Default value	10,000 (rpm)/s

The parameter **All Accelerations Positioning** provides access to the acceleration and deceleration ramp. If the same value is intended for both parameters, only one data value has to be transmitted. This value is then used internally for both acceleration types. It has to be taken into consideration that during a read operation only the current value of the deceleration ramp is read. The user has to make sure that reading a value is sufficient. This can be ensured, for example, by reading a value once and then writing it back.

Subindex	**5**
Name	All Accelerations Positioning
Data type	UINT32
Access	rw
Unit	Physical unit of acceleration
Value range	-
Default value	10,000 (rpm)/s

10.2.4 PNU 1003: Position Profile Type

This parameter allows switching the jerk limitation of position sets between 0 and an automatic determination. For automatic determination the filter time for the jerk limitation will always be calculated in dependence of the acceleration and the profile velocity when calling the position set. By this, the jerk free filter time is also actualised if one of the parameters acceleration or profile velocity is modified. The value does not need to be calculated by the PLC. The “call” of a position set differs from the “start” of a position set. The real start of a position set may be delayed due to the appropriate option, e.g. “wait until the end of the current positioning process”. The “call” refers to the moment when the start command is transmit via fieldbus.

This parameter has different values. Depending on the state of the firmware development one or more position sets may be influenced.

PNU	**1003**
Subindex	**0**
Name	Position Profile Type
Data type	UINT16
Access	rw
Unit	-
Value range	0 .. 1 0: Jerk free filter time for position set = 0. This operation is executed once after Reset or during writing of this parameter. If the jerk free filter time is subsequently modified (e.g. by the parameterisation program) the entered filter time is still effective until a repetitive writing of this parameter or a Reset. 1: Automatic determination of the jerk free filter time of the position set when this position set is called.
Default value	0

10.2.5 PNU 1004: Override Factor

This parameter allows changing the profile velocity of a positioning cycle at any time. For example the profile velocity is divided by two when changing the override to 50 %. After Reset this value is always 100 %. It cannot be saved non-volatile by saving the parameter set.

Changing this parameter does not affect the acceleration. It remains unchanged. So changes of the override during a deceleration phase have no effect on the actual positioning cycle.

PNU	**1004**
Subindex	**0**
Name	Override Factor
Data type	UINT16
Access	rw
Unit	Per mill (1000 corresponds to * 100%)
Value range	0 .. 2000 (corresponds to 0 .. 200%)
Default value	1000 (corresponds to 100%)

Note: The specification of the override factor among the safety parameters must be taken into consideration. This means that 100% of the master specification via the PB only corresponds to 100% if the override factor has been set to 100% among the safety parameters. The input of the override factor via the ServoCommander® will not be stored in the servo drive. After Power on and Reset, this value is always 100%.

Example: Specification of an override factor of 100% among the safety parameters.
If the master transmits an override factor of 30% (corresponds to the value 1000), this corresponds to 30 mm/s and not to 100 mm/s if a speed of 100 mm/s has been specified.

10.2.6 PNU 1005: Software Position Limits

This parameter number sets and reads the software position limits. They have the function of software limit switches. They are only valid in the operating mode positioning. If the target position of a position cycle is beyond these limits, the positioning does not start. If parameterised adequately, a message may be generated.

PNU	**1005**
Name	Software Position Limits

Subindex	**0**
Name	Lower software position limit switch
Data type	INT32
Access	rw
Unit	Physical unit of position
Value range	-
Default value	Maximum negative position

Subindex	**1**
Name	Lower software position limit switch
Data type	INT32
Access	rw
Unit	Physical unit of position
Value range	-
Default value	Maximum positive position

10.2.7 PNU 1006: Rotary Axis

These parameter numbers are intended to set the rotary axis mode and its limits. In rotary axis mode, the position setpoint and the position actual value is limited to the rotary axis limit values. The upper and lower value coincide, i.e. for a rotary axis range of one revolution the lower limit is e.g. 0.0 R and the corresponding upper limit is 1.0 R.

The mode has effect on the setpoint generation in the operating mode positioning only. The mode “Direction always positive” e.g. does not avoid any movement in negative direction at all. The position controller still outputs position setpoints which cause a movement in negative direction. The setpoint generation in any other operating mode is not affected.

PNU	**1006**
Name	Rotary Axis

Subindex	**0**
Name	Rotary axis mode
Data type	UINT8
Access	rw
Unit	-
Value range	0: Off 1: Shortest Distance 2: Direction from position set 3: Direction always positive 4: Direction always negative
Default value	0

Subindex	**1**
Name	Lower rotary axis limit
Data type	INT32
Access	rw
Unit	Physical unit of position
Value range	-
Default value	Maximum negative position

Subindex	**2**
Name	Upper rotary axis limit
Data type	INT32
Access	rw
Unit	Physical unit of position
Value range	-
Default value	Maximum positive position

10.2.8 PNU 1050: Homing Method

Set the homing method by this parameter number. Please refer to the Software Manual Servo drives ARS 2000 for a detailed description of the homing methods.

PNU	**1050**
Subindex	**0**
Name	Homing Method
Data type	INT8
Access	rw
Unit	-
Value range	see Software Manual Servo drives ARS 2000
Default value	1

10.2.9 PNU 1051: Home Offset

Set the distance between the home position and the zero position of a homing process by this parameter number. Positive values move the zero position into positive direction referring to the home position. The following figure illustrates this by an example for the target „limit switch“ and the zero pulse as zero position. The shown direction of the arrow displays the effective direction of the offset.

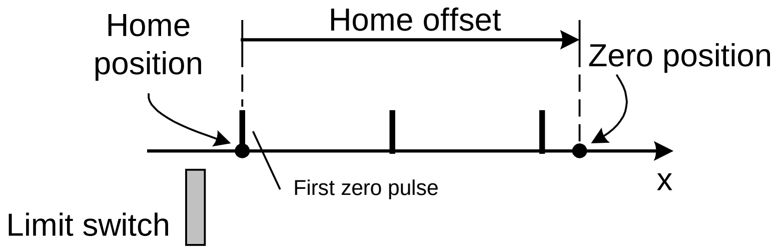


Figure 10-1: Interpretation of the home offset

The data are interpreted in the physical unit position. The physical unit can be set by an appropriate menu of the parameterisation program Metronix ServoCommander®.

PNU	**1051**
Subindex	**0**
Name	Home Offset
Data type	INT32
Access	rw
Unit	Physical unit position
Value range	-
Default value	0

Note:

The determination of the zero position resp. an additional shift of the zero position by the home offset may require saving the corresponding data in the EEPROM of the angle encoder. This is for example useful when applying Multiturn absolute encoders. Please also refer to the Software Manual Servo drives ARS 2000 for further documentation.

10.2.10 PNU 1060: Thread Speed

This parameter modifies the thread speed. The setup mode is activated or deactivated by a corresponding digital input. During active setup mode reduced speed limitations are valid in the operating modes speed control and positioning.

Note:
The thread speed is directly written in velocity units.
Using the parameterisation program Metronix ServoCommander® this value is set in proportional dependence on the speed limit.
Accordingly, the parameterisation has a mutual dependence: Writing the thread speed affects the proportional indication of the thread speed in the parameterisation program Metronix ServoCommander® and vice versa.

The speed limit and the thread speed in % are not available.

PNU	**1060**
Subindex	**0**
Name	Thread Speed
Data type	INT32
Access	rw
Unit	Physical unit of velocity
Value range	-
Default value	3276 rpm

10.2.11 PNU 1270: Position Control Parameters

Use this parameter number to modify the settings of the position controller. The position error tolerance window defines a range of the controller error in which the position controller does not generate any output (speed Setpoint). This may be advantageous for drives with backlash.

PNU	**1270**
Name	Position Control Parameters
Subindex	**2**
Name	Position error tolerance window
Data type	UINT32
Access	rw
Unit	Physical unit position
Value range	0,001 R .. 1 R
Default value	0,01°

10.2.12 PNU 1271: Position Window Data

Use this parameter number for settings concerning the “target reached” message. The Target window defines a range in which the “target reached” message can be generated.

PNU	**1271**
Name	Position Window Data
Subindex	**0**
Name	Target Window
Data type	UINT32
Access	rw
Unit	Physical unit position
Value range	-
Default value	10°

10.2.13 PNU 1272: Following Error Data

Use this parameter number for settings concerning the range for the following error message. The Following error window defines a range outside of which e.g. a warning (following error) is generated (depending on the parameterised reaction).

PNU	1272
Name	Following Error Data

Subindex	0
Name	Following Error Window
Data type	UINT32
Access	rw
Unit	Physical unit position
Value range	0 R .. 101 R
Default value	50°

10.2.14 PNU 1273: Position Error Data

Use this parameter number for settings concerning a position range outside of which the servo drive executes a reaction according to the parameterisation. Besides the following error window here another position window may be defined that may cause the servo drive to switch off while generating an error message (depending on the parameterised reaction).

PNU	1273
Name	Position Error Data

Subindex	0
Name	Position Error Limit
Data type	UINT32
Access	rw
Unit	Physical unit position
Value range	0 .. 2^{31}-1
Default value	180°

10.3 PNUs for speed control mode

This section describes the parameters required for the speed control mode.

10.3.1 PNU 1010: Target Velocity

This parameter is used to set the speed setpoint. The fixed setpoint 1 is intended for these setpoints. In speed control mode, this setpoint is also selected automatically.

Normally, the field bus setpoint 1 can be stored as a fixed setpoint in the parameter set. When the PROFIBUS communication is active in the parameter set, the field bus setpoint is always set to zero. As a result, the value stored in the parameter set is overwritten.

The value 0 for the PROFIdrive control word 1 has the consequence that the field bus setpoint is not led to the ramp (setpoint not enabled). The setpoint selector for speed control has to be set beforehand when the PROFIBUS/PROFINET communication is inactive. Further information can be found in chapter 11 Operating modes.

PNU	**1010**
Subindex	**0**
Name	Target Velocity
Data type	INT32
Access	rw
Unit	Physical unit of velocity
Value range	-
Default value	0

10.3.2 PNU 1011: Accelerations for Velocity Control

Under this parameter number, the acceleration values for the speed control mode can be parameterised. The servo drive defines 4 different acceleration ramps. As in most applications several ramps are parameterised in an identical manner, the following selections are possible:

- Acceleration, combined for positive and negative direction of rotation
- Deceleration, combined for positive and negative direction of rotation
- Combination of acceleration and deceleration for positive and negative direction of rotation

The data are interpreted in the way they were set as physical units. For this purpose, the Metronix ServoCommander® parameterisation software opens a special window (see chapter 7 Physical units).

The parameter with subindex 0 and 1 provides access to the acceleration for both directions of rotation. Internally, it is always written to the accelerations for both directions of rotation. It has to be taken into consideration, that during a read operation only the current value of the acceleration ramp for a positive direction of rotation is read. The user has to make sure that reading a value is sufficient. This can be ensured, for example, by reading a value once and then writing it back.

PNU	**1011**
Name	Accelerations for Velocity Control

Subindex	**0**
Name	Acceleration Velocity Control
Data type	UINT32
Access	rw
Unit	Physical unit of acceleration
Value range	-
Default value	14,100 (rpm)/s

Subindex	**1**
Name	Deceleration Velocity Control
Data type	UINT32
Access	rw
Unit	Physical unit of acceleration
Value range	-
Default value	14,100 (rpm)/s

The parameter **All Accelerations Velocity Control** provides access to the acceleration and deceleration ramp for both directions of rotation. If the same value is intended for all 4 parameters, only one data value has to be transmitted. This value is then used internally for all four acceleration types. It has to be taken into consideration that during a read operation only the current value of the deceleration ramp for the positive direction of rotation is read. The user has to make sure that reading a value is sufficient. This can be ensured, for example, by reading a value once and then writing it back.

Subindex	**2**
Name	All Accelerations Velocity Control
Data type	UINT32
Access	rw
Unit	Physical unit of acceleration
Value range	-
Default value	14,100 (rpm)/s

10.4 PNUs for Different Operating Modes

This section describes parameters which are not especially assigned to one single operating mode.

10.4.1 PNU 1040: Jogging

This parameter allows the simplified access of both velocities and all four acceleration values for jogging. By this, less data has to be transmit in the cyclic data telegram. Internally, all parameters are written.

The parameter of the jogging velocity has the property that it becomes immediately active. A modification has also effect if the jogging mode is active yet.

PNU	**1040**
Name	Jogging

Subindex	**0**
Name	Symmetrical Jogging Velocity
Data type	INT32
Access	rw The read access returns the value of the velocity in positive direction.
Unit	Physical unit velocity
Value range	-
Default value	100 rpm

Subindex	**1**
Name	Symmetrical Jogging Accelerations
Data type	UINT32
Access	rw The read access returns the value of the acceleration in positive direction.
Unit	Physical unit acceleration
Value range	-
Default value	1.000 (rpm)/s

10.4.2 PNU 1041: Jogging Positive

This parameter number sets the parameters for jogging in positive direction (TIPP0) more detailed. By this, depending on the application also a single parameter can be modified.

The parameter of the jogging velocity has the property that it becomes immediately active. A modification has also effect if the jogging in positive direction is active yet.

PNU	**1041**
Name	Jogging Positive

Subindex	**0**
Name	Jogging Velocity Positive
Data type	INT32
Access	rw
Unit	Physical unit velocity
Value range	-
Default value	100 rpm

Subindex	**1**
Name	Jogging Acceleration Positive
Data type	UINT32
Access	rw
Unit	Physical unit acceleration
Value range	-
Default value	1.000 (rpm)/s

Subindex	**2**
Name	Jogging Deceleration Positive
Data type	UINT32
Access	rw
Unit	Physical unit acceleration
Value range	-
Default value	1.000 (rpm)/s

Subindex	**3**
Name	Symmetrical Jogging Accelerations Positive
Data type	UINT32
Access	rw The read access returns the value of the acceleration
Unit	Physical unit acceleration
Value range	-
Default value	1.000 (rpm)/s

10.4.3 PNU 1042: Jogging Negative

This parameter number sets the parameters for jogging in negative direction (TIPP1) more detailed. By this, depending on the application also a single parameter can be modified.

The parameter of the jogging velocity has the property that it becomes immediately active. A modification has also effect if the jogging in negative direction is active yet.

PNU	**1042**
Name	Jogging Negative

Subindex	**0**
Name	Jogging Velocity Negative
Data type	INT32
Access	rw
Unit	Physical unit velocity
Value range	-
Default value	100 rpm

Subindex	**1**
Name	Jogging Acceleration Negative
Data type	UINT32
Access	rw
Unit	Physical unit acceleration
Value range	-
Default value	1.000 (rpm)/s

Subindex	**2**
Name	Jogging Deceleration Negative
Data type	UINT32
Access	rw
Unit	Physical unit acceleration
Value range	-
Default value	1.000 (rpm)/s

Subindex	**3**
Name	Symmetrical Jogging Accelerations Negative
Data type	UINT32
Access	rw The read access returns the value of the acceleration
Unit	Physical unit acceleration
Value range	-
Default value	1.000 (rpm)/s

10.5 Actual values

This section provides an overview of actual values which can be read with the help of parameter numbers.

10.5.1 PNU 1100: Position Actual Value

This parameter is used to return the actual position value. This value is scaled with the physical unit set for PROFIBUS/PROFINET. The calculation of the actual position value may lead to errors as the internal position of the servo drive has a greater value range than the range that can be transmitted. This, however, depends on the physical units set, on the gear ratio and on the feed constant. Please contact the technical support team if you have any queries.

PNU	**1100**
Subindex	**0**
Name	Position Actual Value
Data type	INT32
Access	rw
Unit	Physical unit of position
Value range	-
Default value	-

10.5.2 PNU 1101: Velocity Actual Value

This parameter is used to return the actual speed value. This value is scaled with the physical unit set for PROFIBUS/PROFINET.

PNU	**1101**
Subindex	**0**
Name	Velocity Actual Value
Data type	INT32
Access	ro
Unit	Physical unit of velocity
Value range	-
Default value	-

10.5.3 PNU 1102: Current Actual Value

This parameter is used to read the actual current value. When this value is returned, it is referred to the rated motor current.

PNU	**1102**
Subindex	**0**
Name	Current Actual Value
Data type	INT32
Access	ro
Unit	Parts per thousand with regard to the rated motor current
Value range	-
Default value	-

10.5.4 PNU 1110: Sampling Positions

These parameter numbers return the positions sampled on the rising resp. falling edge of the so called sampling input. For this purpose select the desired input DIN8 or DIN9 in the menu **Parameters - I/Os - Digital inputs** of the parameterisation program Metronix ServoCommander®.

PNU	**1110**
Name	Sampling Positions

Subindex	**0**
Name	Sampling Position Rising Edge
Data type	INT32
Access	ro
Unit	Physical unit position
Value range	-
Default value	-

Subindex	**1**
Name	Sampling Position Falling Edge
Data type	INT32
Access	ro
Unit	Physical unit position
Value range	-
Default value	-

10.5.5 PNU 1141: Digital Inputs

The parameter is used to read the state of the digital inputs. The available digital inputs depend on the parameterisation of the servo drive and on optional technological modules.

PNU	**1141**
Subindex	**0**
Name	Digital Inputs
Data type	UINT32
Access	ro
Unit	-
Value range	Bit assignment: Bit 0: Reserved (= 0) Bit 1: DIN 0 Bit 2: DIN 1 Bit 3: DIN 2 Bit 4: DIN 3 Bit 5: DIN 4 (digital power stage enabling) Bit 6: DIN 5 (digital controller enabling) Bit 7: DIN 6 (limit switch 0 left = negative direction of rotation) Bit 8: DIN 7 (limit switch 1 right = positive direction of rotation) Bit 9: DIN 8 (default: start input) Bit 10: DIN 9 (default: sample input) Bit 11: DIN 10 (optional digital input instead of DOUT 2) Bit 12: DIN 11 (optional digital input instead of DOUT 3) Bit 13..20: DIN 0..7 from optional technological module EA88 in port 1 Bit 21: DIN_AIN 1 (optional digital input instead of analogue input 1) Bit 22: DIN_AIN 2 (optional digital input instead of analogue input 2) Bit 23..30: DIN 0..7 from optional technological module EA88 in port 2 Bit 31: Reserved
Default value	-

10.6 Parameters for the telegram structure

Due to certain technical requirements, some parameters cannot be stored under any random address in the storage area of the master. In addition to that, the length of the telegrams may differ although the same number of data is transmitted. This is why certain parameters have been defined to fill gaps, for example.

10.6.1 PNU 2000: PKW Access

Register this parameter into a telegram in order to variably access different parameter at runtime of the process. There is only one access at maximum available for the receive and response telegrams 1..3. Parameterise this access accordingly. Ensure that this access is registered in all receive and response telegrams.

The term PKW results from Parameter-Kennung-Wert (parameter - identification - value) stated in a former version of the PROFIdrive standard. There, the value for the identification was limited. Here the access is enlarged to higher parameter numbers.

PNU	**2000**	
Subindex	**0**	
Name	PKW Access (manufacturer specific)	
Data type	2 * UINT32 (8 Byte)	
Access	rw	
Unit	-	
Value range	Byte 0:	Access type: 0x00: No access 0x41: Read access 0x42: Write access Other values: reserved
	Byte 1..2:	Parameter number or CAN objects (attention: consider the CAN factor group and enter the CAN object in hexadecimal format)
	Byte 3:	Subindex
	Byte 4..7:	Data
Default value	0	

10.6.2 PNU 2010: Placeholders

These parameters are used to fill up parameters. As a result, data areas (e.g. data blocks) can be structured such that parameters with a length of 2 or 4 bytes are located at even storage addresses.

PNU	**2010**
Name	Placeholder

Subindex	**0**
Name	8 Bit
Data type	UINT8
Access	rw
Unit	-
Value range	-
Default value	0

Subindex	**1**
Name	16 Bit
Data type	UINT16
Access	rw
Unit	-
Value range	-
Default value	0

Subindex	**2**
Name	32 Bit
Data type	UINT32
Access	rw
Unit	-
Value range	-
Default value	0

10.6.3 PNU 2011: Element 0

This parameter has basically the same purpose as the parameter PNU 2010.2. The only difference is that it is not shown in the telegram editor of the ServoCommander® program, if it is entered at the end of a telegram. The number of entries in a telegram is limited to 10. Unused entries therefore contain this PNU.

PNU	**2011**
Subindex	**0**
Name	Element 0
Data type	UNT32
Access	rw
Unit	-
Value range	-
Default value	0

10.7 Parameter for Various Intents

This section contains parameters which are not explicitly assigned to a special function group.

10.7.1 PNU 1600: Last Error Code

This parameter number returns the last generated error.

PNU	**1600**	
Subindex	**0**	
Name	Last Error Code	
Data type	UINT16	
Access	ro	
Unit	-	
Value range	Bit 0..3:	Subindex error number (0..9)
	Bit 4..15:	Main index error number (1..96)
Default value	- The PNU returns “0” if no error is active. Then, the value is invalid.	

10.7.2 PNU 1510: CAM Control

These parameters are used to activate or deactivate CAM disks or an axis error compensation. The appropriate CAM or axis error compensation tables have to be transferred to the servo drive by the parameterisation program Metronix ServoCommander® before.

PNU	**1510**
Name	CAM Control

Subindex	**0**
Name	Control CAM disks
Data type	UINT32
Access	rw
Unit	-
Value range	Bit 0..7: Number of the selected CAM disk Bit 16 : Activate (1) or deactivate (0) the CAM disk
Default value	0

Subindex	**1**
Name	Control axis error compensation

Data type	UINT32
Access	rw
Unit	-
Value range	Bit 0..7: Number of the selected axis error compensation Bit 16 : Activate (1) or deactivate (0) the axis error compensation
Default value	0

11 Operating modes

11.1 Overview

The servo drives of the ARS 2000 product range offer three basic operating modes:

- Torque control
- Speed control
- Positioning

Within the operating modes, different behaviour is the result of setpoint selectors which can be parameterised differently. The positioning mode, for example, offers additional modes, e.g. for point-to-point positioning or clock-synchronised operation.

PROFIdrive (see chapter 1.2) defines so-called application classes. These classes can be set using corresponding parameter numbers. To simplify this, the operating mode is linked to the cyclic receive telegrams. Following the PROFIdrive specification, the following operating modes and the associated receive telegrams are supported at present:

Operating mode	Receive telegram	Identifier
Positioning	0	0xE0
Speed control	1	0xE1

Other operating modes cannot be selected at present.

11.2 Parameters

The operating mode is continuously monitored or selected with the help of receive telegrams. The process of changing the operating mode requires several cycles of an internal function. This is why a manufacturer-specific parameter number has been defined to be able to read the current operating mode.

11.2.1 PNU 1500: Operating Mode

This manufacturer-specific parameter is used to set/read the operating mode. The operating mode is served regardless of the setpoint selectors. Other special features can be found in the associated chapters.

PNU	**1500**
Subindex	**0**
Name	Operating Mode

Data type	UINT8
Access	rw
Unit	-
Value range	0x08: Speed control 0x10: Positioning
Default value	-

11.3 Speed control mode

PROFIdrive defines special features for handling the setpoint. For this purpose, the meanings of the corresponding bits in control word 1 have been defined. The setpoint can be deactivated, for example, or the setpoint ramp can be stopped (“frozen”). The following points have to be fulfilled to meet this requirement:

- When the PROFIBUS communication is activated, the fixed setpoint 1 is activated for the adder. If the controller enabling logic has been set to DIN5 and PROFIBUS, this selector is also called PROFIBUS in Metronix ServoCommander®.
- If the setpoint for the setpoint ramp has been disabled by the PROFIdrive control word 1, no setpoint is activated in the adder (no check box is marked in Metronix ServoCommander®).

11.4 Positioning mode

PROFIdrive defines special features concerning the behaviour in the positioning mode. For this purpose, the meanings of the corresponding bits in control word 1 have been defined. Homing controlled by the slave, for example, is started by a certain bit. Some features are now described in detail:

- All global options for homing are also valid when homing is started through the PROFIdrive control word 1. An optional following positioning to zero position is performed.
- Due to a manufacturer-specific implementation, positioning is started only on a rising edge of the associated bit in control word 1.
- Positioning is started even if no successful homing has been performed beforehand.

Other manufacturer-specific bits are defined in control word 1 in order to be able to perform optional relative or absolute positioning tasks.

- Distinction between absolute and relative positioning
- Definition as to whether at the start of a positioning run a running position has to be interrupted or whether the positioning run to the started has to be added to the running positioning.

Some applications require a sequence of motion tasks to be executed without interruption (see Figure 11-1). This can be realised in two different ways:

1. Interruption of the running positioning
2. Start of a following positioning with the end velocity being the profile velocity for the first motion task.

The second case has to be applied if the second positioning run has to start at a specific position.

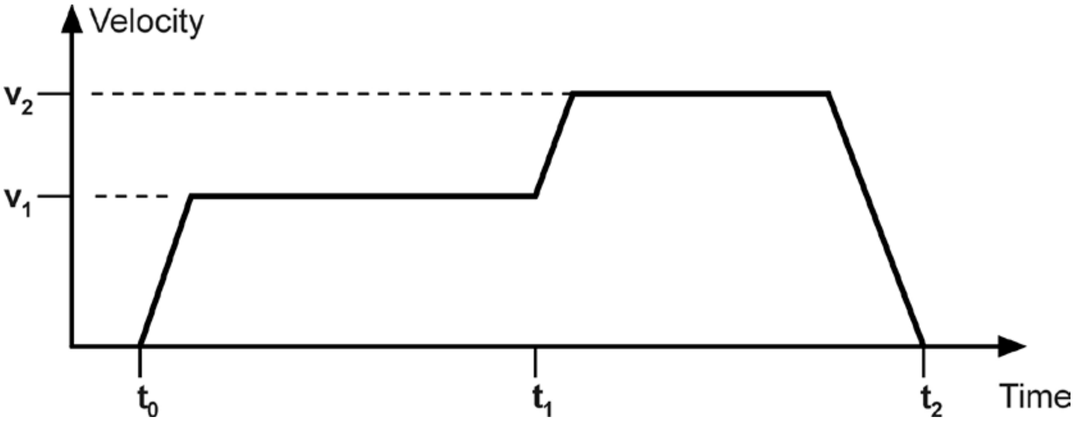


Figure 11-1: Uninterrupted sequence of motion tasks

12 Profile-specific parameter numbers

This chapter provides an overview of the parameter numbers (PNU) which were implemented following PROFIdrive.

12.1 Overview

The following table provides an overview of the implemented PNUs.

PNU	Sub-index	Description	Type	Access
918	0	Slave address [1)]	UINT16	ro
963	0	Automatically detected baud rate [1)]	UINT16	ro
964	-	(Device identification)		
	0	Manufacturer ID	UINT16	ro
	1	Device type	UINT16	ro
	2	Device version	UINT16	ro
	3	Firmware date (year)	UINT16	ro
	4	Firmware date (day/month)	UINT16	ro
967	0	Control word 1	V2	rw
968	0	Status word 1	V2	ro
971	0	Save parameters	UINT16	rw

[1)]: Parameter is only relevant for PROFIBUS.

Note: Data type V2 is defined as a bit sequence with a length of 2 bytes.

12.2 PNU description

The PNUs 967 and 968 are described in detail in chapter 9 Device control. The other PNUs are described in the following sections.

12.2.1 PNU 918: Node address

This parameter returns the slave address of the servo drive. The addresses always start at 0. The addresses up to 2, for example, are occupied by the PROFIBUS master or other control devices. Thus the lowest address is 3.

The PROFIdrive specification defines 126 as a fixed default value. However, it has to be taken into consideration that the exchange of useful data is prohibited under this address.

PNU	**918**
Subindex	**0**
Name	Node address
Data type	UINT16
Access	ro
Unit	-
Value range	3 .. 126
Default value	126

The Parameter number 918 is only relevant for the fieldbus technology PROFIBUS.

12.2.2 PNU 963: Actual baud rate

This parameter supplies the baud rate of the PROFIBUS communication. The baud rate is automatically detected by the hardware used.

PNU	**963**
Subindex	**0**
Name	Actual baud rate
Data type	UINT16
Access	ro
Unit	-

Value range	0: 9.6 kBaud 1: 19.2 kBaud 2: 93.75 kBaud 3: 187.5 kBaud 4: 500 kBaud 5: No baud rate detected or specified 6: 1500 kBaud 7: 3000 kBaud 8: 6000 kBaud 9: 12000 kBaud 11: 45.45 kBaud
Default value	5

The Parameter number 963is only relevant for the fieldbus technology PROFIBUS.

12.2.3 PNU 964: Device Identification

The device identification parameter provides information on the connected device under several subindexes.

PNU	**964**
Name	Device identification

Subindex	**0**
Name	Content: Manufacturer
Data type	UINT16
Access	ro
Unit	-
Value range	277 (0x0115)
Default value	277 (0x0115)

Subindex	**1**
Name	Content: Device Type
Data type	UINT16
Access	ro
Unit	-

Value range	0x2000 Unknown device type 0x2005 ARS 2102 0x2006 ARS 2105 0x2009 ARS 2302 0x200A ARS 2305 0x200B ARS 2310 0x200C ARS 2320 0x2008 ARS 2320W 0x200D ARS 2340 0x200E ARS 2360W 0x2045 ARS 2102 FS 0x2046 ARS 2105 FS 0x2050 ARS 2108 FS 0x2049 ARS 2302 FS 0x204A ARS 2305 FS 0x204B ARS 2310 FS
Default value	-

Subindex	**2**
Name	Content: Version
Data type	UINT16
Access	ro
Unit	-
Value range	hhss: hh: Main revision ss: Subrevision
Default value	-

The PNUs with subindexes 3 and 4 indicate the date of creation of the firmware. This date of creation may be the same in different product levels or revisions if the basic implementation has not been changed.

Subindex	**3**
Name	Content: Firmware date (date of creation, year)
Data type	UINT16
Access	ro
Unit	Year
Value range	-
Default value	-

Subindex	**4**
Name	Content: Firmware date (date of creation, day / month)
Data type	UINT16
Access	ro
Unit	-
Value range	ttmm: tt: Day mm: Month
Default value	-

12.2.4 PNU 971: Transfer into a Non-Volatile Memory

Active different saving processes of parameters inside the servo drive by this parameter number.

PNU	**971**	
Subindex	**0**	
Name	Transfer into a non-volatile memory	
Data type	UINT16	
Access	rw	
Unit	-	
Value range	0:	No action
	1:	Save the actual parameter set and all position data sets
	2..255:	reserved
	256:	Save all position data sets without the standard parameters
	257:	Save parameters into the angle encoder
	258:	Save the new actual position (e.g. after a homing process) into the angle encoder. The other parameter of the EEPROM are also saved with their actual values.
	259..65535:	reserved
Default value	0	

Some saving processes allow the analysis if the process is still running or completed with an error. The actual state of the implementation:

Reading returns “1” as long as saving of the complete parameter set including all position data sets is active. The process is activated by writing the value “1”. In other cases reading always returns “0”.

13 PROFIBUS function and data blocks for SIEMENS S7

13.1 Overview

Special function blocks have been written for ARS 2000 servo drives and Siemens PLC systems (SIMATIC-S7 control systems). These functions blocks facilitate the integration of the servo drives into a PLC program with PROFIBUS functionality. The various function and data blocks (FD, DB) are assigned to one of the operating modes.

To make the use of the function blocks clearer, example programs were written. These programs complement the Application Note.

This chapter is intended to familiarise the user quickly with the functions of the inputs and outputs and the use of the FBs and DBs within the S7 environment. **However, this chapter does not replace the previous chapters of the PROFIBUS/PROFINET Manual.**

The available example programs are only examples and explain the fundamental methods for using the function and data blocks.

When the example programs are used in customer-specific applications, the user has to check whether all function- and safety-relevant requirements have been fulfilled.

Information

Specific function and data blocks have been developed for the ARS 2000 servo drives. These function and data blocks are integrated in example programs and can be downloaded free of charge under **www.metronix.de**.

If one switches between the operating modes, one also has to switch between two function blocks. If the controller enabling shall remain active during switching between two function blocks, it might be necessary to modify the function blocks accordingly. Please contact the technical support department.

Attention

To switch from one operating mode to another, the current FB has to be deactivated and an FB of another operating mode has to be activated. As a result, the controller enabling might be deactivated.

13.2 Function and data blocks for ARS 2000 servo drives

13.2.1 Function block (FB) overview

The function blocks (FBs) are used to control the state diagram implemented in the servo drive.

Status word 1 in byte 2/3 of the DB40 actual_value data block symbolises the actual state of the servo drive. Control word 1, which is contained in bytes 2/3 of every mode-specific telegram format and described by the FB, is used to control the servo drive.

The master connection provides the PLC system in defined I/O areas with the input and output data of the servo drives. These data are read out of the slave through SFC14 and written to the slave through SFC15. SFCs 14 and 15 are system functions which **must** be used for consistent reading and writing of telegrams with a length > 4 bytes.

The data are stored in the data blocks defined for the operating mode in question through the corresponding SFCs. The SFCs are not integrated in the function blocks and **must** be loaded as S7 program elements.

In practice, the DBs are used as I/O modules to control the states of the servo drive and to transmit data which have to be sent to the servo drive.

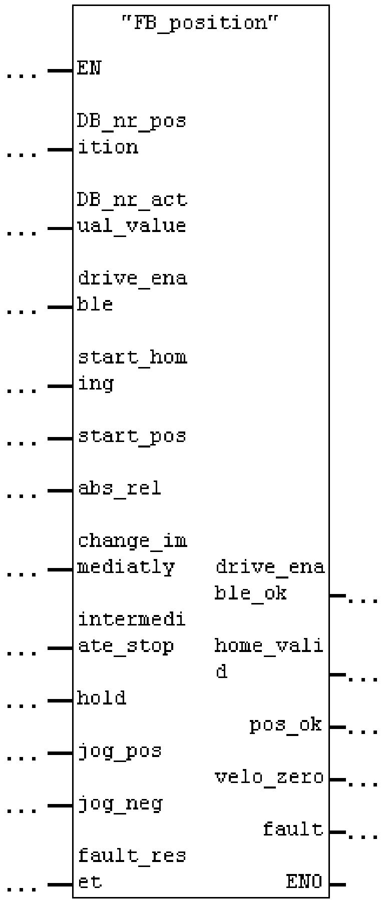
Every operating mode and every servo drive requires separate function blocks. If several servo drives are used in different operating modes, this might lead to bottlenecks concerning the storage utilisation of the application.

Information

The order of the data in the data blocks corresponds to the order defined in the telegram format of the associated operating mode (FB).

13.2.2 FBs of ARS 2000 servo drives

13.2.2.1 FB_position (FB 41 positioning mode)



- **Parameters:**
  - **DB_nr_position:**

    Number of the data block containing the data for the positioning mode (INT).
  - **DB_nr_actual_value:**

    Number of the data block containing the actual value data (INT).
- **Inputs:**
  - **drive_enable:**

    This input is used to activate the controller enabling in the positioning mode. The drive is held in its position.
  - **start_homing:**

    This input is used to start a homing run. The motor is started. This requires an active controller enabling, i.e. the output drive_enable_ok must be set. If the start_homing input is reset during homing, the homing run is aborted without an error.

- **start_pos:**

  A rising edge signals that a new motion task has to be accepted. A falling edge has no consequence.
  During homing, this input has no effect.

- **abs_rel:**

  If this input is set with a rising edge at the start_pos input, the positioning run is executed relatively to the current target position. If this input is not set with a rising edge at start_pos, an absolute positioning run is executed.

- **change_immediatly:**

  If this input is set with a rising edge at the start_pos input, a running positioning run is aborted immediately and replaced with a new motion task. If this input is not set with a rising edge at start_pos, the new motion task is added to the end of the running positioning run. In this case, the pos_ok output is not set at the end of the current positioning run, but at the end of the added positioning run.

- **intermediate_stop**

  If this input is not set, a started positioning run is completed. If the input is set during a running positioning run, the drive is stopped and remains in position control mode. The current positioning run is **not** terminated. It will be continued when the intermediate_stop input is reset.
  During homing, this input has no effect.

- **hold:**

  If this input is set, the current positioning run is stopped. The system is slowed down with the deceleration valid for this positioning run. After the process has been completed, the pos_ok output is not set. A reset of the input has no effect.
  During homing, this input has not effect.

- **jog_pos:**

  If the input is set, the drive accelerates with the acceleration value set in position set JOG 0 until it reaches the speed also parameterised in this position set.
  If there is a falling edge at this input, the drive slows down to standstill with the deceleration set in position set JOG 0.
  During homing, this input has no effect.

- **jog_neg:**

  If the input is set, the drive accelerates with the acceleration set in position set JOG 1 until it reaches the speed also parameterised in this position set.
  If there is a falling edge at this input, the drive slows down to standstill with the deceleration set in position set JOG 1.
  During homing, this input has no effect.

- **fault_reset:**

  If there is a rising edge at this input, pending fault messages are acknowledged.

- **Outputs:**
  - **drive_enable_ok:**

    The controller enabling of the servo drive is active.
  - **home_valid:**

    This output is set, if there is a valid home position. The output is not set during a running homing run. It is not set or reset until after a homing run has been completed successfully.
  - **pos_ok:**

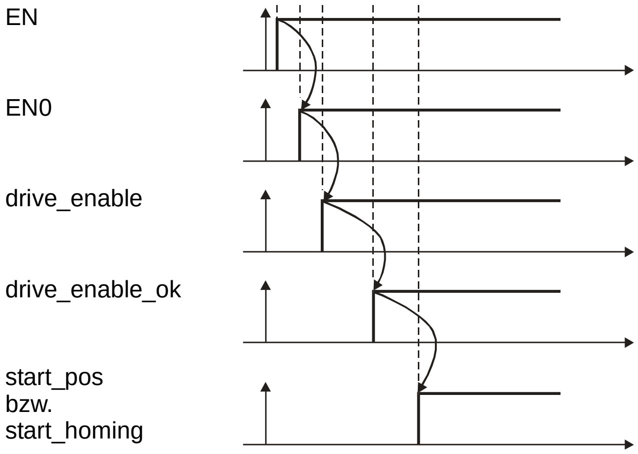
    This output is set, when the current position is in the target window after the positioning run has been completed.
  - **velo_zero:**

    If this output is set, the drive stands still.
  - **fault:**

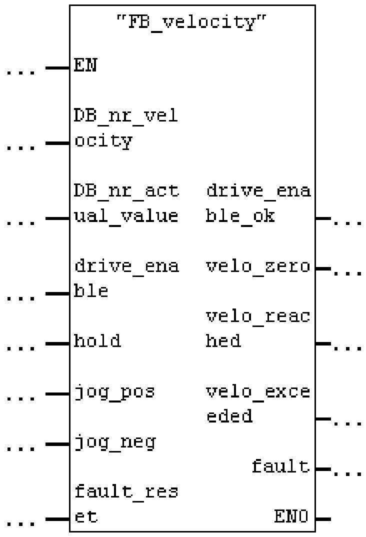
    If this output is set, there is a fault. The controller enabling is cancelled.

    The output is reset after the fault has been acknowledged.

Please note the timing shown in the diagram when using the function blocks.



13.2.2.2 FB_velocity (FB 43 speed control)



- **Parameters:**
  - **DB_nr_velocity:**

    The number of the data block containing the data for speed control mode (INT)
  - **DB_nr_actual_value:**

    The number of the data block containing the actual value data (INT).

- **Inputs:**
  - **drive_enable:**

    Activation of the controller enabling in speed control mode. The speed setpoint takes effect depending on the other inputs of the function block.
  - **hold:**

    If this input is set, the speed setpoint is blocked. The drive slows down to standstill as quickly as possible. When the input is reset, the motor accelerates as quickly as possible to reach the adjusted speed setpoint.
  - **jog_pos:**

    If this input is set, the drive accelerates with the set acceleration of the setpoint ramp until it reaches the speed parameterised in position set JOG 0.
    If there is a falling edge at this input, the drives slows down to standstill with the set deceleration of the setpoint ramp.
  - **jog_neg:**

    If this input is set, the drive accelerates with the set acceleration of the setpoint ramp until it reaches the speed parameterised in position set JOG 1.
    If there is a falling edge at this input, the drives slows down to standstill with the adjusted deceleration of the setpoint ramp.
  - **fault_reset:**

    If there is a rising edge at this input, pending fault messages are acknowledged.

- **Outputs:**
  - **drive_enable_ok:**

    The controller enabling of the servo drive is active.
  - **velo_zero:**

    This output is set, when the actual speed is zero (± 10 rpm).
  - **velo_reached:**

    When this output is set, the actual speed is in the parameterised tolerance window of the target speed (reference speed).
  - **velo_exceeded:**

    If this output is set, the actual speed is greater than the free reference speed.
  - **fault:**

    If this output is set, there is a fault. The controller enabling is cancelled.

    The output is reset after the fault has been acknowledged.

13.2.3 Integration of FBs and DBs

As shown in the overview, the function and data blocks developed for the ARS 2000 servo drives can be downloaded free of charge.

Information

The function and data blocks can be downloaded under **www.metronix.de**

The blocks are packaged into two example projects in the file **FB_DB_ARS_2000_xpx.zip**. **xpx** stands for the current version number. The file can be opened using commercially available programs, such as, for example, WINZIP. Example projects:

Operating mode	Example project
Positioning	ARS2_POS.zip
Speed control	ARS2_VEL.zip

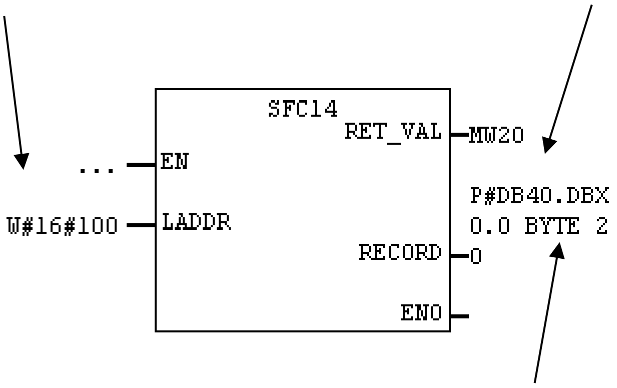
The unpacked example projects can be de-archived under STEP7. The relevant FBs, DBS and SFCs can be copied from these de-archived example objects.

To ensure consistent data exchange between the PROFIBUS master and the PROFIBUS slave, the system functions SFC14 (consistent reading) and SFC15 (consistent writing) must be used. The following parameters have to be used for the blocks:

Reading data using SFC14:

Input data area
256dez = 100hex

Data block number into which
the data of the slave are written



Initial address within the DB (0.0) and number of bytes received (20 bytes)

Parameter	Declaration	Data type	Storage area	Description
LADDR	INPUT	WORD	E, A, M, D, L, const.	Projected initial address of the E-area of the module which has to be read.
RET_VAL	OUTPUT	INT	E, A, M, D, L	If a fault occurs while the function is processed, the return value contains a fault code.
RECORD	OUTPUT	ANY	E, A, M, D, L	Target area for the useful data read. This area has to be as long as has been projected for the selected module with STEP 7. Only the data type BYTE is admissible.

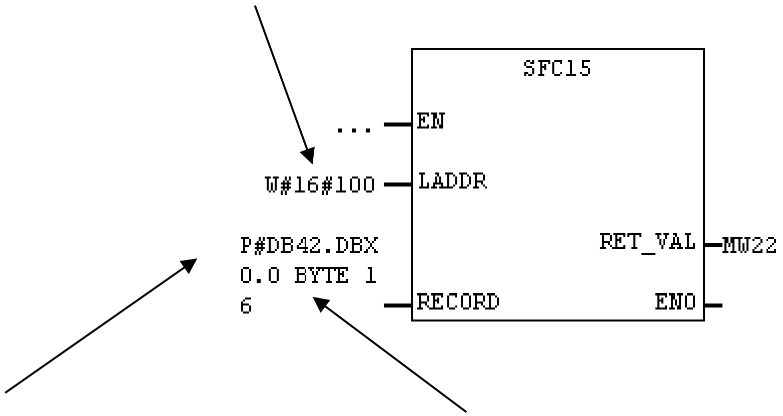
In both example projects (positioning example project and speed control example project), the same actual value data are returned. As a consequence, the response telegram is configured in an identical manner in both example projects.

Writing data using SFC15:

Depending on the operating mode (positioning or speed control), different data are transmitted to the PROFIBUS slave. This leads to different telegram lengths affecting the parameterisation of SFC15.

Writing data for the positioning mode:

Output data area 256dez = 100hex

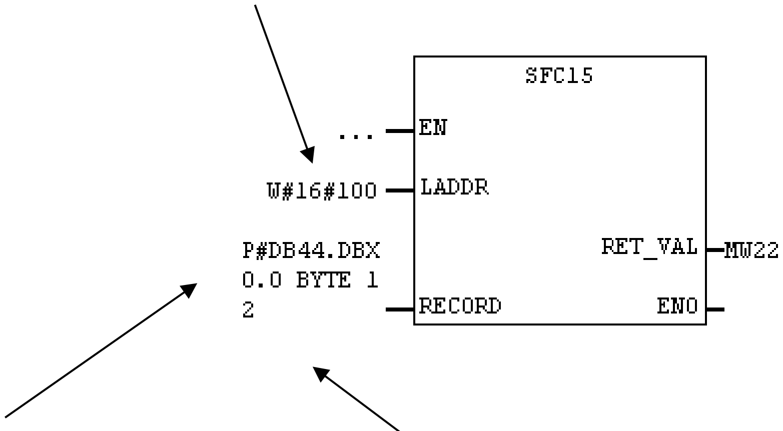


Number of the data block
of which data are transmitted
to the slave.

Initial address within the DB (0.0) and
number of bytes to be transmitted (16 bytes)

Writing data for the speed control mode:

Output data area 256dez = 100hex



Number of the data block
of which data are transmitted
to the slave.

Initial address within the DB (0.0) and
number of bytes to be transmitted (12 bytes)

Parameter	Declaration	Data type	Storage area	Description
LADDR	INPUT	WORD	E, A, M, D, L, const.	Projected initial address of the A-area of the module which has to be written to.
RECORD	INPUT	ANY	E, A, M, D, L	Source area of the useful data to be written. It has to be as long as has been projected for the selected module with STEP 7. Only the data type BYTE is admissible.
RET_VAL	OUTPUT	INT	E, A, M, D, L	If a fault occurs while the function is processed, the return value contains a fault code.

13.2.4 PROFIBUS connection of ARS 2000 servo drive

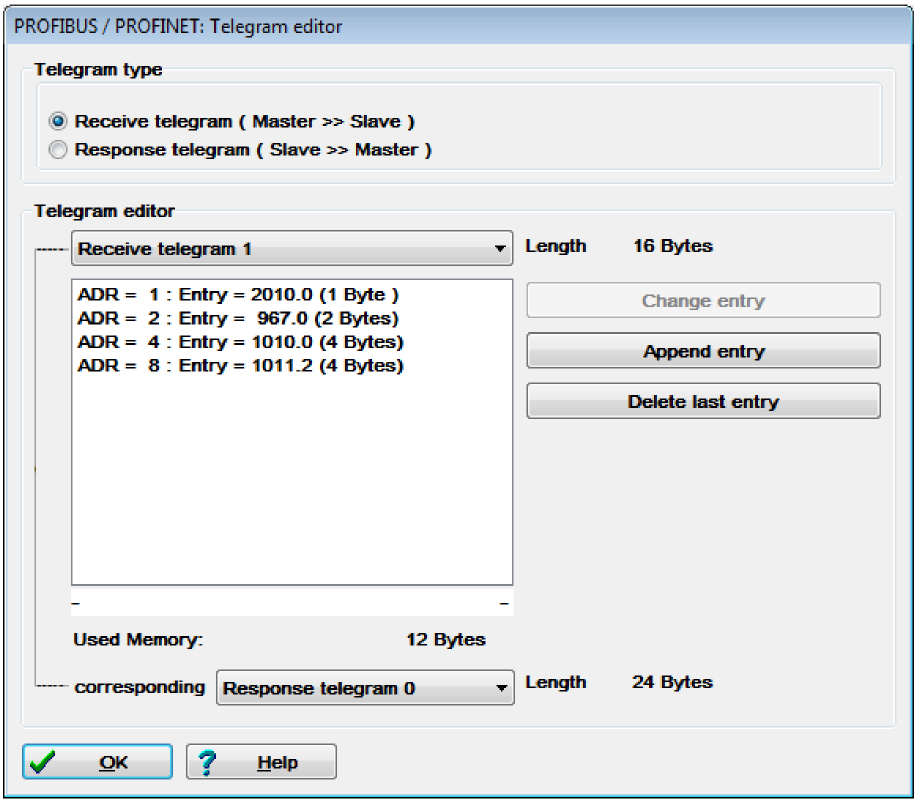
Several steps have to be performed to set up a functional PROFIBUS connection. Some settings should or have to be made before the PROFIBUS communication can be activated.

The transmission of data using the DP-V0 service requires a definition of the number of data to be transmitted and of the meaning of these data, both at the master and at the slave end. If the example projects described above are used, the master and the slave have to be configured accordingly.

When the example projects are loaded, the data blocks are already structured as required. The telegrams have to be structured as required by the example projects using the Metronix ServoCommander® parameterisation program. Information concerning the use of the telegram editor and further information on the parameters can be found in the previous chapters of the PROFIBUS product manual.

13.2.4.1 Receive telegram of the speed control example project

In this case, receive telegram 1 at the servo drive end has to be configured. The following screenshot shows the necessary parameterisation of the receive telegram for the example project. An additional diagnostic information is displayed above the button „Change entry“ if a communication could have been established between Master and Slave. The really configured length of the telegram from Master to Slave is displayed.



PROFIBUS / PROFINET: Telegram editor

Telegram type

● Receive telegram (Master >> Slave)
○ Response telegram (Slave >> Master)

Telegram editor

Receive telegram 1 Length 16 Bytes

ADR = 1 : Entry = 2010.0 (1 Byte)
ADR = 2 : Entry = 967.0 (2 Bytes)
ADR = 4 : Entry = 1010.0 (4 Bytes)
ADR = 8 : Entry = 1011.2 (4 Bytes)

Used Memory: 12 Bytes

corresponding Response telegram 0 Length 24 Bytes

The following table describes the telegram structure:

Address	Content (parameter number)	Description
0	Identifier (= 0xE1)	Fixed identifier
1	8-bit placeholder (PNU 2010.0)	Free
2	Control word 1 (PNU 967.0)	Control word for device control, has to be under this address
4	Target velocity (PNU 1010.0)	Speed setpoint, indication with the physical unit set for PROFIBUS
8	Acceleration (PNU 1011.2)	Combination of the values for acceleration and deceleration, indication with the physical unit set for PROFIBUS

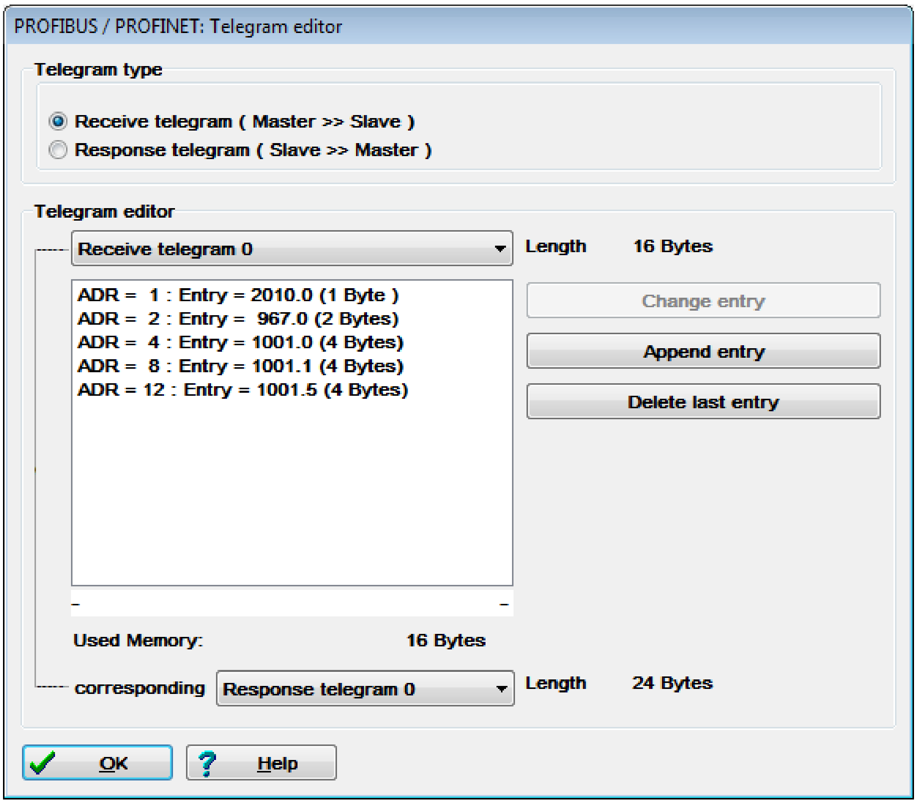
The total length of this telegram is 12 bytes. When the slave is integrated into the hardware configuration of the master, the module:

6 Words Output

has to be selected for the output data and connected to one of the ports.

13.2.4.2 Receive telegram of the positioning example project

In this case, receive telegram 0 at the servo drive end has to be configured. The following screenshot shows the necessary parameterisation of the receive telegram for the example project. An additional diagnostic information is displayed above the button „Change entry" if a communication could have been established between Master and Slave. The really configured length of the telegram from Master to Slave is displayed.


PROFIBUS / PROFINET: Telegram editor
Telegram type
● Receive telegram (Master >> Slave)
○ Response telegram (Slave >> Master)
Telegram editor
Receive telegram 0
Length 16 Bytes
Change entry
Append entry
Delete last entry
OK
Help

```
ADR =  1 : Entry = 2010.0 (1 Byte )
ADR =  2 : Entry =  967.0 (2 Bytes)
ADR =  4 : Entry = 1001.0 (4 Bytes)
ADR =  8 : Entry = 1001.1 (4 Bytes)
ADR = 12 : Entry = 1001.5 (4 Bytes)
```

Used Memory: 16 Bytes

corresponding Response telegram 0 Length 24 Bytes

The following table describes the telegram structure:

Address	Content (parameter number)	Description
0	Identifier (= 0xE0)	Fixed identifier
1	8-bit placeholder (PNU 2010.0)	Free
2	Control word 1 (PNU 967.0)	Control word for device control, has to be under this address
4	Target position (PNU 1001.0)	Target position, indication with the physical unit set for PROFIBUS
8	Profile velocity (PNU 1001.1)	Profile velocity during positioning, indication with the physical speed unit set for PROFIBUS
12	Accelerations (PNU 1001.5)	Combination of the values for acceleration and deceleration, indication with the physical unit set for PROFIBUS

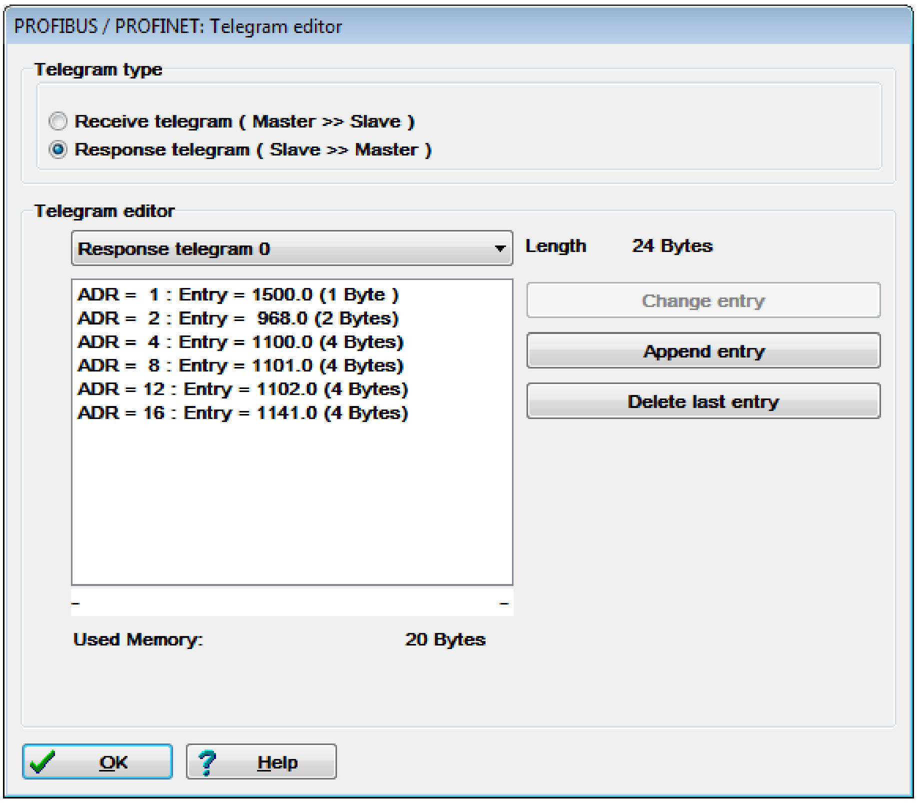
The total length of this telegram is 16 bytes. When the slave is integrated into the hardware configuration of the master, the module:

8 Words Output

has to be selected for the output data and connected to one of the ports.

13.2.4.3 Response telegram for the example projects

In both example projects, the same actual value data are transmitted from the slave to the master. Thus, the same response telegram 0 has to be configured for both receive telegrams (0 and 1) at the servo drive end. The following screenshot shows the necessary parameterisation for the example project. An additional diagnostic information is displayed above the button „Change entry“ if a communication could have been established between Master and Slave. The really configured length of the telegram from Master to Slave is displayed.



PROFIBUS / PROFINET: Telegram editor

Telegram type

○ Receive telegram (Master >> Slave)
● Response telegram (Slave >> Master)

Telegram editor

Response telegram 0

Length 24 Bytes

```
ADR =  1 : Entry = 1500.0 (1 Byte )
ADR =  2 : Entry =  968.0 (2 Bytes)
ADR =  4 : Entry = 1100.0 (4 Bytes)
ADR =  8 : Entry = 1101.0 (4 Bytes)
ADR = 12 : Entry = 1102.0 (4 Bytes)
ADR = 16 : Entry = 1141.0 (4 Bytes)
```

Used Memory: 20 Bytes

The following table describes the telegram structure:

Address	Content (parameter number)	Description
0	Identifier (= 0xF0)	Fixed identifier
1	Operating mode (PNU 1500.0)	Current operating mode of the servo drive
2	Status word 1 (PNU 968.0)	Status word for device control, has to be under this address
4	Position Actual Value (PNU 1100.0)	Actual position, indication with the physical position unit set for PROFIBUS
8	Velocity Actual Value (PNU 1101.0)	Actual speed value, indication with the physical speed unit set for PROFIBUS
12	Current actual value (PNU 1102.0)	Actual value of the current, indicated per thousand referring to the nominal motor current
16	Digital inputs (PNU 1141.0)	Current status of the digital inputs, see for the meaning of the bits in the detailed description of the PNU.

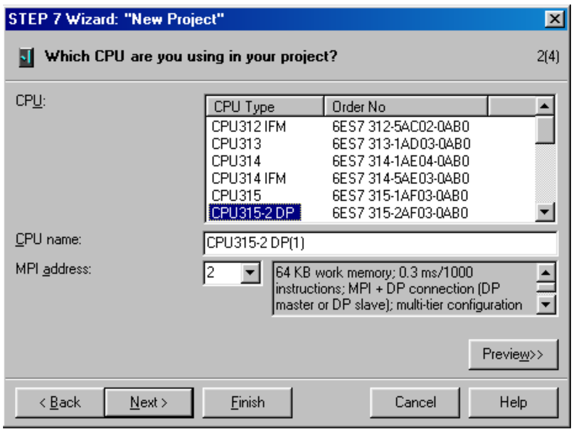
The total length of this telegram is 20 bytes. When the slave is integrated into the hardware configuration of the master, the module:

10 Words Input

has to be selected for the input data and connected to one of the ports.

13.2.5 Creating an S7 program

First start the STEP 7 software. After the start of the SIMATIC Manager, the STEP7 Wizard is opened. Follow the instructions of the STEP 7 Wizard. The STEP 7 Wizard leads you step by step through the creation of an S7 project.



STEP 7 Wizard: "New Project"

Which CPU are you using in your project? 2(4)

CPU:

CPU Type	Order No
CPU312 IFM	6ES7 312-5AC02-0AB0
CPU313	6ES7 313-1AD03-0AB0
CPU314	6ES7 314-1AE04-0AB0
CPU314 IFM	6ES7 314-5AE03-0AB0
CPU315	6ES7 315-1AF03-0AB0
CPU315-2 DP	6ES7 315-2AF03-0AB0

CPU name: CPU315-2 DP(1)

MPI address: 2

64 KB work memory; 0.3 ms/1000 instructions; MPI + DP connection (DP master or DP slave); multi-tier configuration

Then select the organisation module **OB1**. **OB1** is located at the top program level and organises all the other modules.

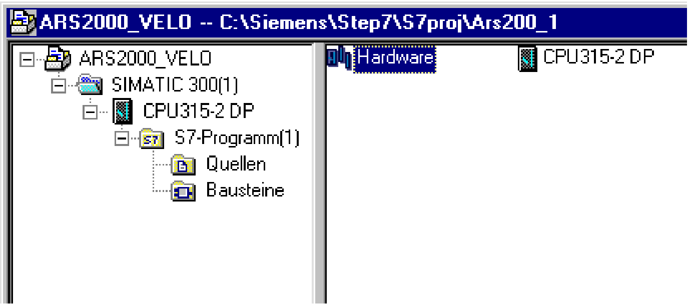
Additional modules can be added later.

After that, select the preferred programming language. This language can be converted into other programming languages later.

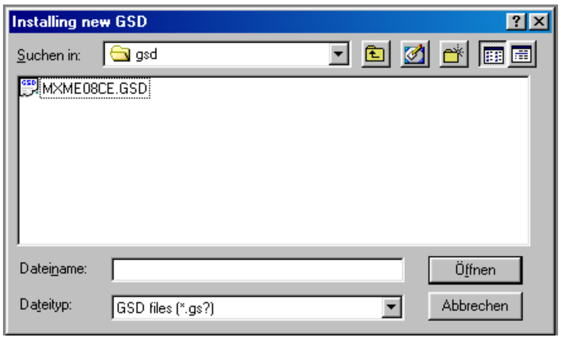
13.2.6 Integrating the GSD file

As a next step, the hardware of the PLC system used must be configured in the project created. To do so, select **SIMATIC 300** in the left pane. Then double-click on **Hardware** in the right pane.

Close the station by clicking on **Station** -> **Close**, without quitting the hardware configuration program.

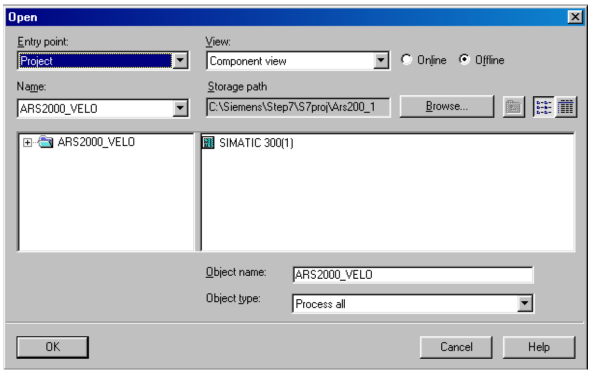


The command **Options->Install new GSD...** adds the GSD file MXME08CE.gsd, which is required for adding the ARS2000 to the hardware catalog.



13.2.7 Hardware configuration

The S7 project must then be opened using the command **Station** -> **Open...** .



Entry point: Project

View: Component view ○ Online ● Offline

Name: ARS2000_VELO

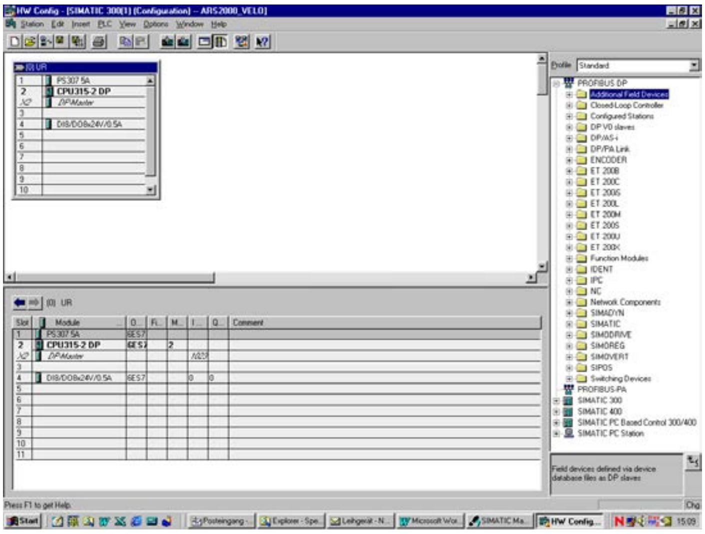
Storage path: C:\Siemens\Step7\S7proj\Ars200_1

Object name: ARS2000_VELO

Object type: Process all

Select the project name in the **Name** field of the **Open** window. Then select **SIMATIC 300(1)** in the right window pane. Then click on the **OK** button to open the hardware configuration of your project.

The following screenshot shows the hardware catalog from which the hardware components can be selected. If the hardware catalog is not shown, it can be called up by clicking on **View** -> **Catalog**.



HW Config - [SIMATIC 300(1) (Configuration) -- ARS2000_VELO]

(0) UR

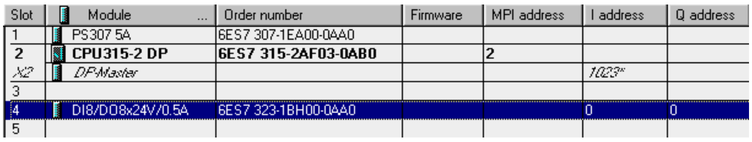
Slot	Module	O...	Fi...	M...	I...	Q...	Comment
1	PS307 5A	6ES7					
2	CPU315-2 DP	6ES7		2			
X2	DP-Master				1023		
3							
4	DI8/DO8x24V/0.5A	6ES7			0	0	
5							
6							
7							
8							
9							
10							
11							

Profile: Standard

Field devices defined via device database files as DP slaves

Press F1 to get Help.

Use the mouse to select the desired hardware components from the **Catalog** and drag them into the station window.



Slot	Module	Order number	Firmware	MPI address	I address	Q address
1	PS307 5A	6ES7 307-1EA00-0AA0				
2	CPU315-2 DP	6ES7 315-2AF03-0AB0		2		
X2	DP-Master				1023*	
3						
4	DI8/DO8x24V/0.5A	6ES7 323-1BH00-0AA0			0	0
5						

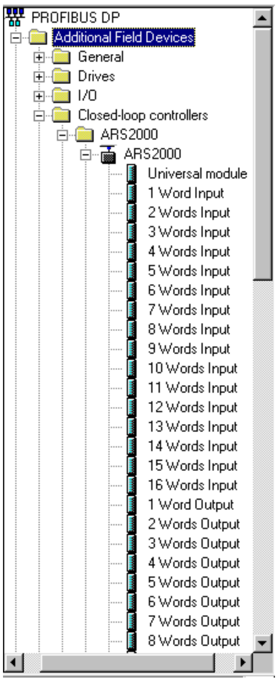
Information

The order numbers in the project must match the order numbers of the hardware (shown on the front). Port 3 is not used as the project does not use a controller board.

13.2.8 Integrating the servo drive

The servo drive is integrated by selecting the line **X2 DP-Master** in the station window. Then add the PROFIBUS with the help of the command **Insert** -> **Master System**.

Then add the desired module to the PROFIBUS using drag&drop from the directory **PROFIBUS-DP** -> **Additional Field Devices** -> **Closed-loop controllers-> ARS2000** of the hardware catalog.



```
PROFIBUS DP
  Additional Field Devices
    General
    Drives
    I/O
    Closed-loop controllers
      ARS2000
        ARS2000
          Universal module
          1 Word Input
          2 Words Input
          3 Words Input
          4 Words Input
          5 Words Input
          6 Words Input
          7 Words Input
          8 Words Input
          9 Words Input
          10 Words Input
          11 Words Input
          12 Words Input
          13 Words Input
          14 Words Input
          15 Words Input
          16 Words Input
          1 Word Output
          2 Words Output
          3 Words Output
          4 Words Output
          5 Words Output
          6 Words Output
          7 Words Output
          8 Words Output
```

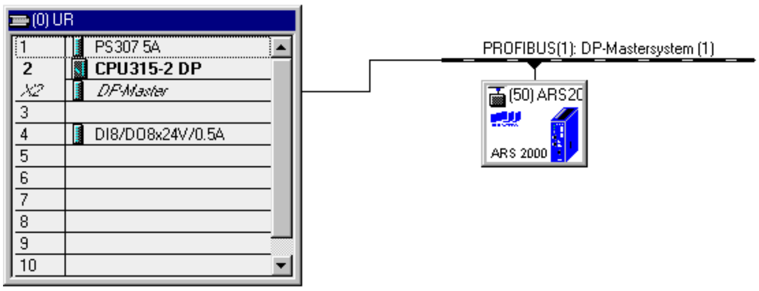
The telegram length has to be taken into consideration when selecting the modules. This means:

- Select the module **8 Words Output** for the positioning mode setpoint telegram and the module **6 Words Output** for the speed control setpoint telegram.
- Select the module **10 Words Input** for the actual value telegram.

It is also possible to use both function blocks in one project. In this case, the module **8 Words Output** has to be used.

The PROFIBUS address of the servo drive has to be set in the **Properties** window. The address has to be the same as the address previously set with the Metronix ServoCommander® program. The servo drive is now integrated.

The hardware configuration should now look like that:



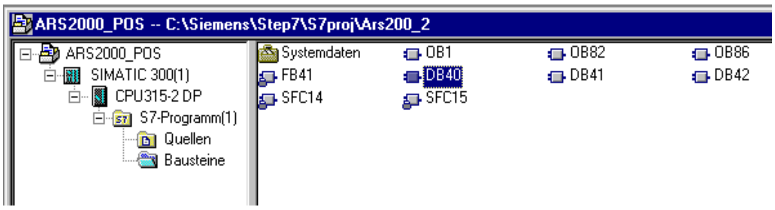
(0) UR

1	PS307 5A
2	**CPU315-2 DP**
X2	*DP-Master*
3	
4	DI8/DO8x24V/0.5A
5	
6	
7	
8	
9	
10	

The S7 project is now prepared and programming can be started.

13.2.9 Overview of function and data blocks:

Overview of positioning mode example project

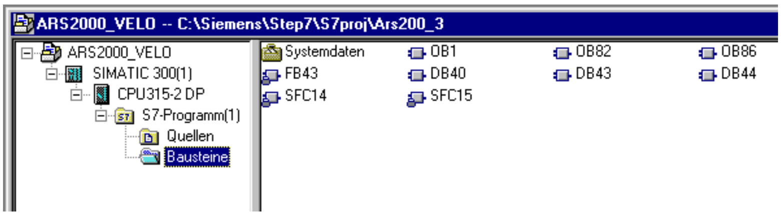


ARS2000_POS -- C:\Siemens\Step7\S7proj\Ars200_2

- ARS2000_POS
  - SIMATIC 300(1)
    - CPU315-2 DP
      - S7-Programm(1)
        - Quellen
        - Bausteine

Systemdaten	OB1	OB82	OB86
FB41	DB40	DB41	DB42
SFC14	SFC15		

Overview of speed control example project



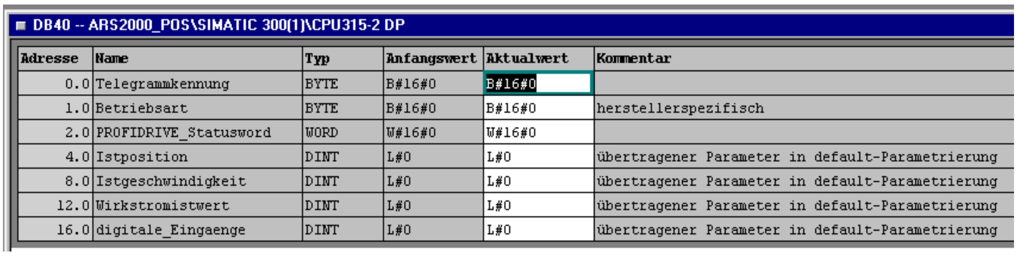
ARS2000_VELO -- C:\Siemens\Step7\S7proj\Ars200_3

- ARS2000_VELO
  - SIMATIC 300(1)
    - CPU315-2 DP
      - S7-Programm(1)
        - Quellen
        - Bausteine

Systemdaten	OB1	OB82	OB86
FB43	DB40	DB43	DB44
SFC14	SFC15		

Actual values

DB40 actual_value Actual values of the servo drive (for both example projects)



DB40 -- ARS2000_POS\SIMATIC 300(1)\CPU315-2 DP

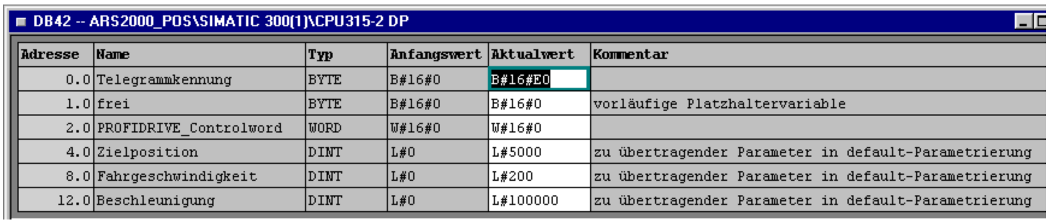
Adresse	Name	Typ	Anfangswert	Aktualwert	Kommentar
0.0	Telegrammkennung	BYTE	B#16#0	B#16#0	
1.0	Betriebsart	BYTE	B#16#0	B#16#0	herstellerspezifisch
2.0	PROFIDRIVE_Statusword	WORD	W#16#0	W#16#0	
4.0	Istposition	DINT	L#0	L#0	übertragener Parameter in default-Parametrierung
8.0	Istgeschwindigkeit	DINT	L#0	L#0	übertragener Parameter in default-Parametrierung
12.0	Wirkstromistwert	DINT	L#0	L#0	übertragener Parameter in default-Parametrierung
16.0	digitale_Eingaenge	DINT	L#0	L#0	übertragener Parameter in default-Parametrierung

Positioning

FB41 FB_position Function block for the positioning mode

DB41 Instance DB for FB41 The data block of internal variables belonging to the function block

DB42 DB_position Positioning data

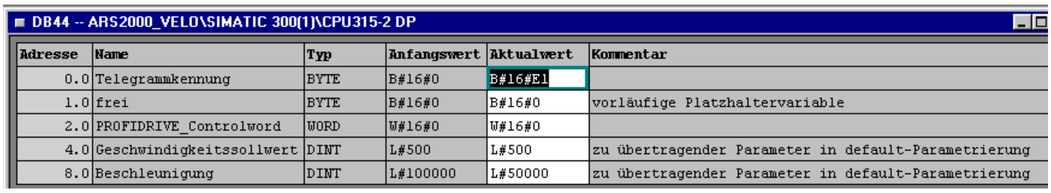


DB42 -- ARS2000_POS\SIMATIC 300(1)\CPU315-2 DP

Adresse	Name	Typ	Anfangswert	Aktualwert	Kommentar
0.0	Telegrammkennung	BYTE	B#16#0	B#16#E0	
1.0	frei	BYTE	B#16#0	B#16#0	vorläufige Platzhaltervariable
2.0	PROFIDRIVE_Controlword	WORD	W#16#0	W#16#0	
4.0	Zielposition	DINT	L#0	L#5000	zu übertragender Parameter in default-Parametrierung
8.0	Fahrgeschwindigkeit	DINT	L#0	L#200	zu übertragender Parameter in default-Parametrierung
12.0	Beschleunigung	DINT	L#0	L#100000	zu übertragender Parameter in default-Parametrierung

Speed control

FB43	FB_velocity	Function block for speed control mode
DB43	Instance DB for FB43	Data block of internal variables belonging to the function block
DB44	DB_velocity	Data for speed control mode



DB44 -- ARS2000_VELO\SIMATIC 300(1)\CPU315-2 DP

Adresse	Name	Typ	Anfangswert	Aktualwert	Kommentar
0.0	Telegrammkennung	BYTE	B#16#0	B#16#E1	
1.0	frei	BYTE	B#16#0	B#16#0	vorläufige Platzhaltervariable
2.0	PROFIDRIVE_Controlword	WORD	W#16#0	W#16#0	
4.0	Geschwindigkeitssollwert	DINT	L#500	L#500	zu übertragender Parameter in default-Parametrierung
8.0	Beschleunigung	DINT	L#100000	L#50000	zu übertragender Parameter in default-Parametrierung

14 PROFINET function blocks and data blocks for SIEMENS S7

14.1 Overview

For the ARS 2000 servo drives and Siemens PLC systems (SIMATIC S7 control systems), special function blocks have been written. They support the easy integration of a servo drive into a PLC program with PROFINET functionality. The function blocks and data blocks (FB, DB) are assigned to a specific operating mode.

Example programs have been written in order to explain and demonstrate the handling of the function blocks.

The aim of this chapter is to provide the user with a quick introduction into the function of the inputs and outputs and also into the handling of the FBs and DBs in the S7 environment. **This chapter does not replace the preceding chapters of the PROFIBUS/PROFINET manual.**

As their name indicates, the example programs are only examples and they are provided in order to explain the basic handling of the various function blocks and data blocks.

If the example programs are used in customer-specific applications, the user must check whether all of the function- and safety-relevant conditions are fulfilled.

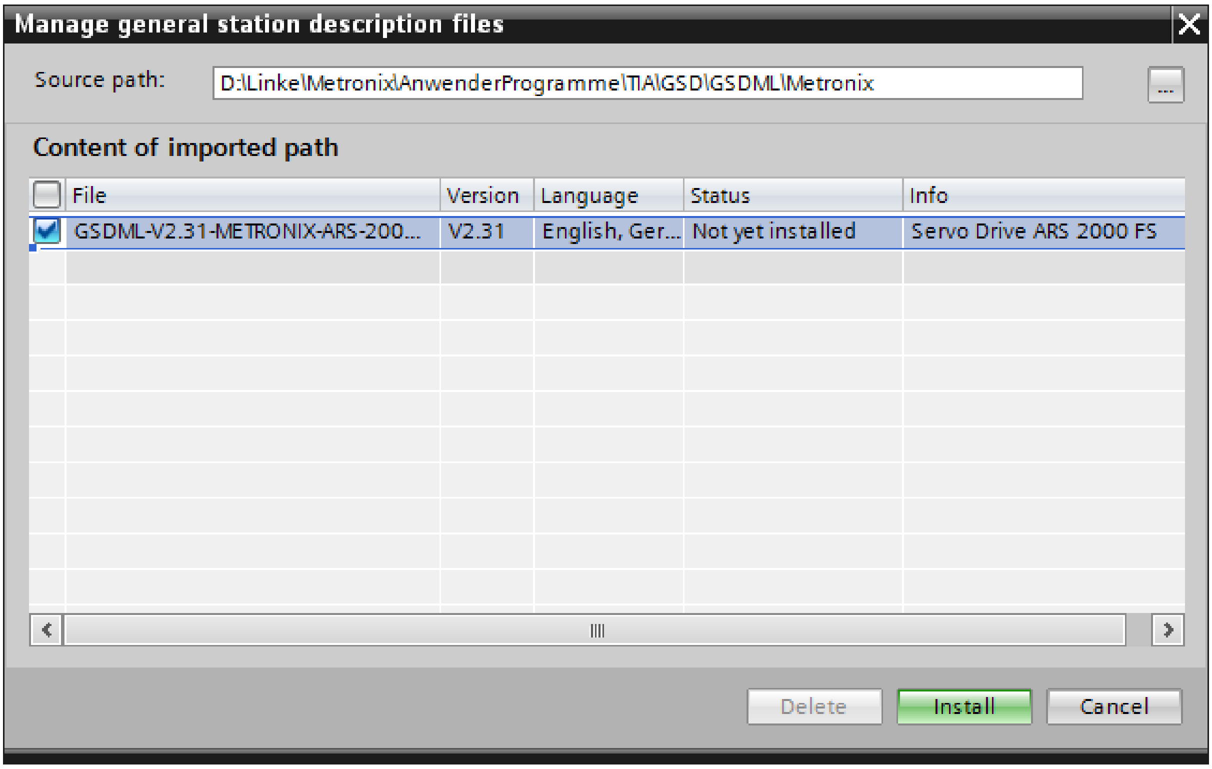
The following chapters explain the fundamental hardware configuration and the use of the function blocks in an example project.

14.2 Integration of the hardware in the TIA portal

14.2.1 Installation of the GSDML file

In the TIA (Totally Integrated Automation) portal, the menu item **Manage general station description files (GSD)** must be selected on the menu bar under **Options**. Select the path to the GSDML file (i.e. **GSDML-V2.31-METRONIX-ARS-2000-FS-20160512.xml**) in the field **Source path** and install the file.

The respective current version of the GSDML file can be found on our homepage for download: http://www.metronix.de



Source path: D:\Linke\Metronix\AnwenderProgramme\TIA\GSD\GSDML\Metronix

☐ File	Version	Language	Status	Info
☑ GSDML-V2.31-METRONIX-ARS-200...	V2.31	English, Ger...	Not yet installed	Servo Drive ARS 2000 FS

Figure 14-1: Menu for the integration of the GSDML file

After the installation and closing of the menu window, the hardware catalog is updated. As a result, the ARS 2000 servo drive is then available for selection in the hardware catalog (see Figure 14-2).

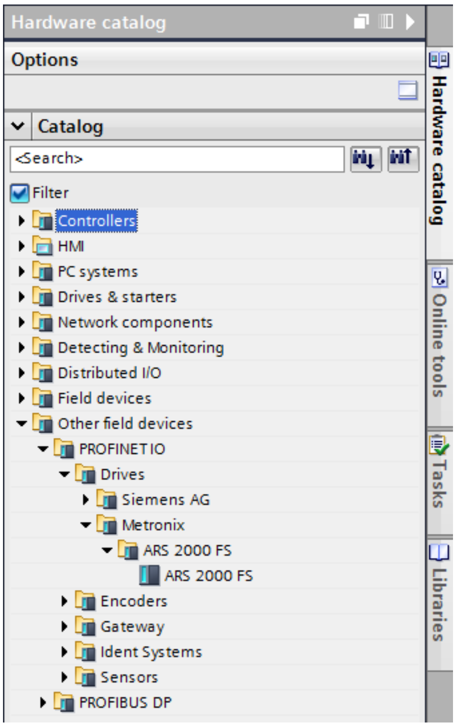


Figure 14-2: Hardware catalog overview

14.2.2 Hardware configuration

The hardware components can be selected from the hardware catalog. Select the desired hardware components in the **hardware catalog** by way of the mouse and drag and drop them into the **topology view** of the TIA portal.

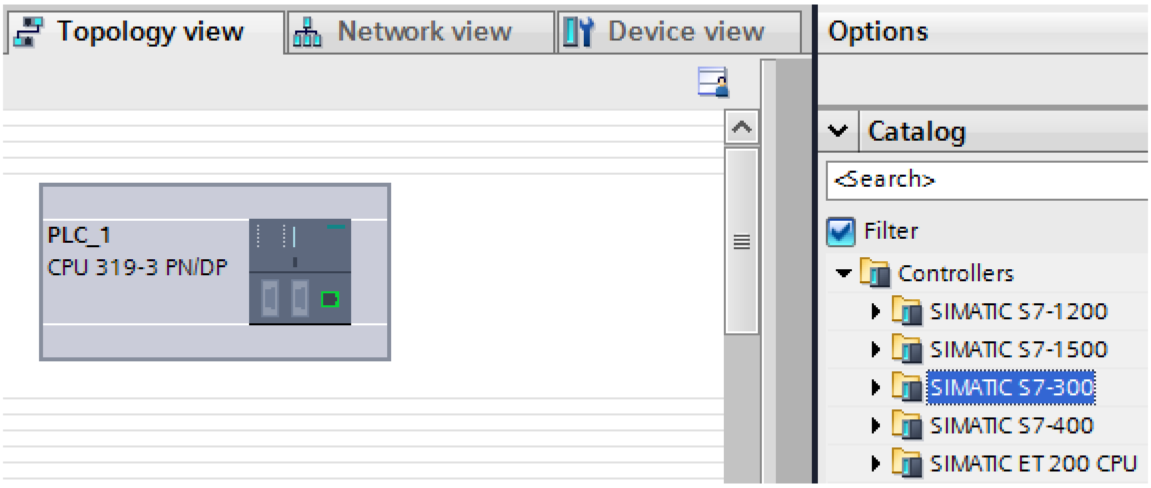


Figure 14-3: Example: use of S7-300

The order numbers in the project must correspond to the order numbers of the hardware (printed on the front).

14.2.3 Integration of the servo drive

In order to integrate the servo drive into the PROFINET network, the servo drive must be selected in the hardware catalog. The ARS 2000 head module must be dragged and dropped into the **topology view** of the device configuration. Dragging the mouse from one PROFINET port to another PROFINET port connects the integrated devices. After the selection of the network view, the servo drive must be assigned to an I/O controller via the link "Not assigned" (see Figure 14-4).

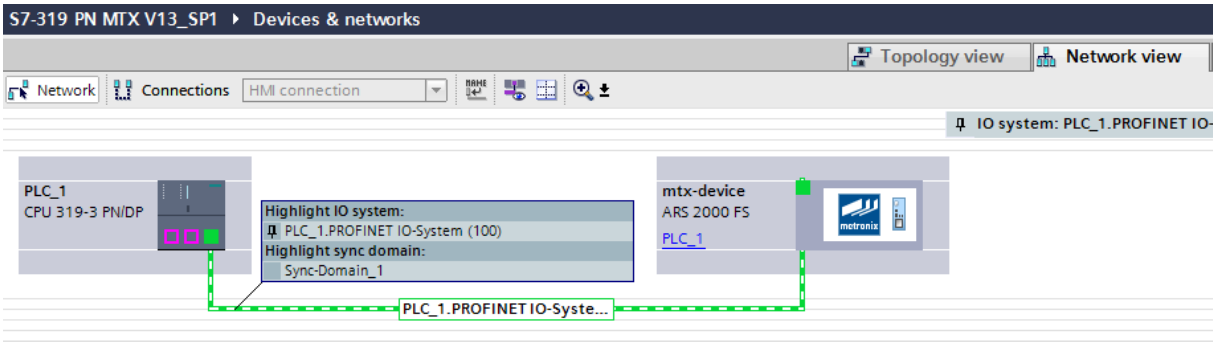
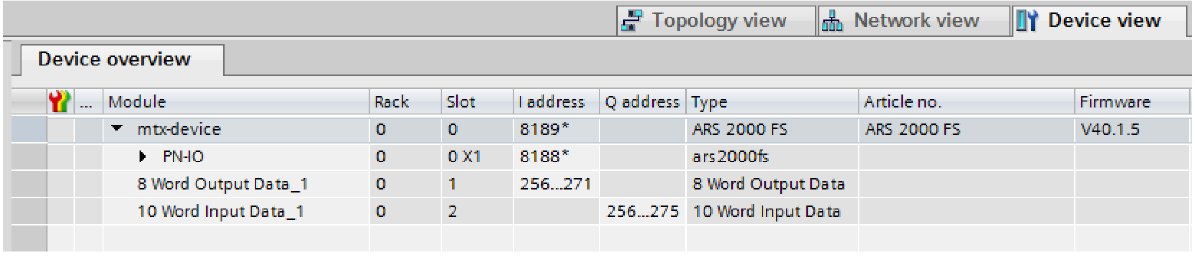


Figure 14-4: IO controller assignment

The **device name** is automatically assigned to the servo drive. After clicking the Metronix logo of the servo drive, the **device name** can be changed under the menu item **General** in the **Properties** window.

This is followed by the configuration of the telegram data. The IN and OUT modules are transferred from the hardware catalog into the blank lines of the **device view** by drag & drop. The input and output addresses are automatically assigned, but it is possible to change them manually.



	...	Module	Rack	Slot	I address	Q address	Type	Article no.	Firmware
		mtx-device	0	0	8189*		ARS 2000 FS	ARS 2000 FS	V40.1.5
		PN-IO	0	0 X1	8188*		ars2000fs		
		8 Word Output Data_1	0	1	256...271		8 Word Output Data		
		10 Word Input Data_1	0	2		256...275	10 Word Input Data		

Figure 14-5: Configuration of the telegram data

When selecting the modules, the respective telegram length must be taken into consideration. This means:

- The module **8 Words Output** must be selected for the setpoint telegram for the position control mode or speed control mode.
- The module **10 Words Input** must be selected for the actual value telegram.

The integration of the servo drive is now complete and the S7 project has been prepared to such an extent that the programming of the control system can commence.

14.3 Function blocks for ARS 2000/ARS 2000 FS servo drives

14.3.1 Download of the function blocks/example project

The following downloads for servo drives of the ARS 2000/ ARS 2000 FS type are available for download at http://www.metronix.de:

- Function blocks for position and speed control. The blocks are provided in the format of the TIA library, which means that they can be particularly easily integrated into an existing TIA project (see also chapter 14.3.2). The function blocks are included in the following ZIP file and they can be opened by way of a standard program, e.g. WINZIP:
  Metronix_ARS2000_Funktionsbausteine.zip
- An example project in the TIA format for position and speed control. This project already enables the control of a servo drive. Only the hardware environment must be configured beforehand (see the information provided from chapter 14.2 onward). The example project is included in the following ZIP file and it can be opened by way of a standard program, e.g. WINZIP:
  Metronix_ARS2000_Beispielprojekt.zip

Switching between the two operating modes also requires switching between two function blocks. If the disabling of the servo drive is to be avoided during the change-over between two function blocks, it may be necessary to modify the function blocks accordingly. Please contact the technical support team in this case.

Attention

When switching from one operating mode to another, the current FB must be deactivated and the FB of the other operating mode must be used instead. As a consequence, the servo drive may be disabled.

14.3.2 Overview of the function blocks

The function blocks (FB) are responsible for controlling the state diagram that is implemented in the servo drive. In the TIA, the function blocks are imported via the menu item **Global libraries - Create new global library**. Select the global library that you want to open. The library file has the extension ".al[version number]". This means that global libraries, which have been saved with the current product version of the TIA portal, have the file extension ".al13". If you have selected a global library from an earlier version of the TIA portal, the function blocks must be migrated. Further information can be found in the TIA portal under **Help**.

When the process is complete, the function blocks are available under **Global libraries** (see Figure 14-6).

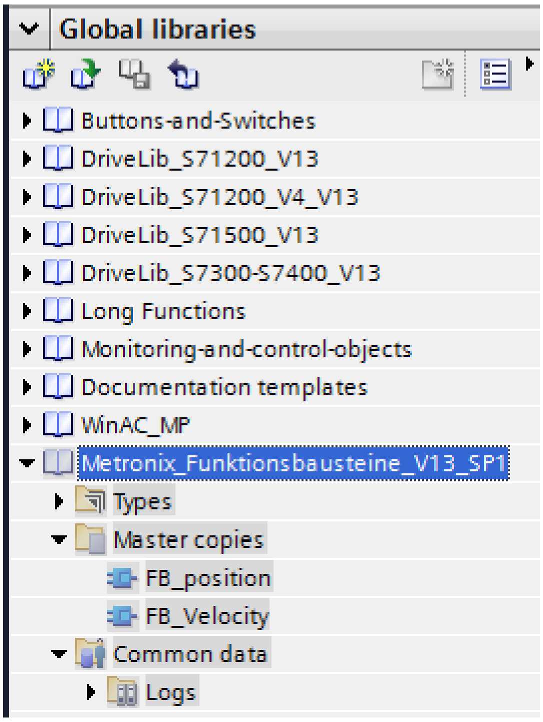
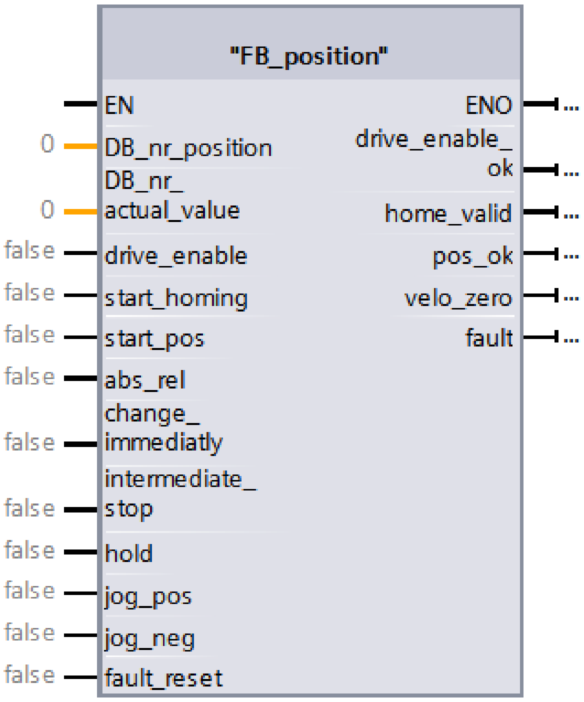


Figure 14-6: Metronix function blocks

Every servo drive requires separate function blocks for each of the operating modes

14.3.2.1 FB_position (position control mode)



- **Parameters:**
  - **DB_nr_position:**

    Number of the data block containing the data for the position control mode (INT).
  - **DB_nr_actual_value:**

    Number of the data block containing the actual value data (INT).
- **Inputs:**
  - **drive_enable:**

    This input is used to enable the servo drive in the position control mode. The drive will be held in its position and remain in the position control mode.
  - **start_homing:**

    This input is used to start the homing run. The motor starts. A requirement for this is that the servo drive has been enabled, i.e. the output drive_enable_ok must be set. If the input start_homing is reset during a homing run, the homing run will be aborted without generating an error.
  - **start_pos:**

    A rising edge indicates that a new motion task is to be started. A falling edge has no effect.
    This input does not have any effect during a homing run.

- **abs_rel:**

  If there is a rising edge at the input start_pos and this input is set, the positioning run will be performed relative to the current position setpoint. If there is a rising edge at start_pos, but this input is not set, an absolute positioning run will be performed.

- **change_immediatly:**

  If there is a rising edge at the input start_pos and this input is set, a running positioning process will be aborted immediately and replaced with the new motion task. If there is a rising edge at start_pos, but this input is not set, the new motion task will be added to the end of the current positioning run. In this case, the output pos_ok will not be set at the end of the current positioning run, but at the end of the added positioning run.

- **intermediate_stop**

  If this input is not set, a started positioning run will be completed. If the input is set during a positioning run, the drive will stop and remain in position control mode. The current positioning run is **not** complete. It will be continued when the input intermediate_stop is reset.
  This input does not have any effect during a homing run.

- **hold:**

  If this input is set, the current positioning run will be stopped. The drive will decelerate with the deceleration rate that is applicable for this positioning run. After the process has been completed, the output pos_ok will not be set. A reset of the input has no effect.
  This input does not have any effect during a homing run.

- **jog_pos:**

  If the input is set, the drive accelerates up to the speed of motion that is parameterised in this position set. It does so with the acceleration rate that is set in the position set TIPP 0.
  If there is a falling edge at this input, the drive decelerates to a standstill with the deceleration rate that is set in the position set TIPP 0.
  This input does not have any effect during a homing run.

- **jog_neg:**

  If the input is set, the drive accelerates up to the speed of motion that is parameterised in this position set. It does so with the acceleration rate that is set in the position set TIPP 1.
  If there is a falling edge at this input, the drive decelerates to a standstill with the deceleration rate that is set in the position set TIPP 1.
  This input does not have any effect during a homing run.

- **fault_reset:**

  If there is a rising edge at this input, any pending fault messages will be acknowledged.

- **Outputs:**

  - **drive_enable_ok:**

    The controller enable signal of the servo drive is active.

  - **home_valid:**

This output is set if there is a valid home position. The output is not set during a running homing run. It is only after the successful completion of a homing run that it is set for the first time or set again.

- **pos_ok:**

  This output is set if the current position is in the target window after the positioning run has been completed.

- **velo_zero:**

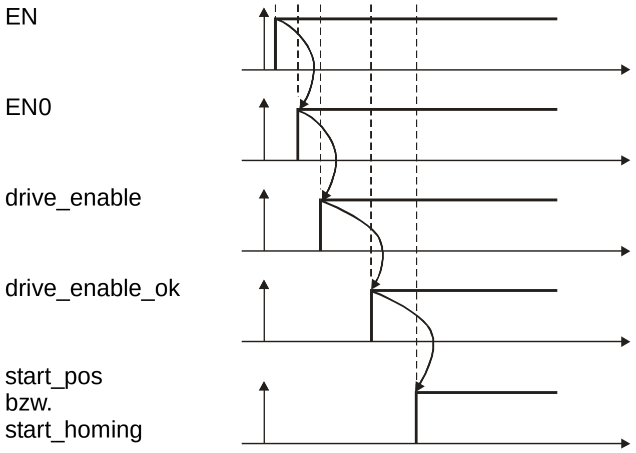
  If this output is set, the drive stands still.

- **fault:**

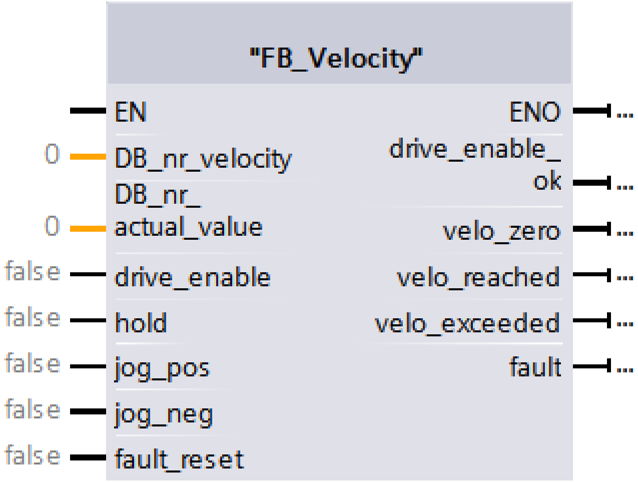
  If this output is set, there is a fault. The servo drive is disabled.

  The output is reset after the fault has been acknowledged.

Please note the timing that is shown in the diagram when using the function blocks.



14.3.2.2 FB_velocity (speed control)



- **Parameters:**
  - **DB_nr_velocity:**

    Number of the data block containing the data for the speed control mode (INT).
  - **DB_nr_actual_value:**

    Number of the data block containing the actual value data (INT).
- **Inputs:**
  - **drive_enable:**

    This input is used to enable the servo drive in the speed control mode. The speed setpoint becomes effective depending on the other inputs of the function block.
  - **hold:**

    If the input is set, the speed setpoint value is blocked. The drive decelerates as quickly as possible until it stops. When the input is reset, the motor accelerates as quickly as possible to reach the set speed setpoint.
  - **jog_pos:**

    If this input is set, the drive accelerates with the set acceleration rate of the setpoint ramp until it reaches the speed that is parameterised in the position set TIPP 0.
    If there is a falling edge at this input, the drive decelerates to a standstill with the set deceleration rate of the setpoint ramp.
  - **jog_neg:**

    If this input is set, the drive accelerates with the set acceleration rate of the setpoint ramp until it reaches the speed that is parameterised in the position set TIPP 1.
    If there is a falling edge at this input, the drive decelerates to a standstill with the set deceleration rate of the setpoint ramp.
  - **fault_reset:**

    If there is a rising edge at this input, any pending fault messages will be acknowledged.

- **Outputs:**
  - **drive_enable_ok:**

    The controller enable signal of the servo drive is active.
  - **velo_zero:**

    This output is set if the actual speed value is zero (± 10 rpm).
  - **velo_reached:**

    If this output is set, the actual speed is within the parameterised tolerance window of the set speed (comparison speed).
  - **velo_exceeded:**

    If this output is set, the actual speed is higher than the free comparison speed.
  - **fault:**

    If this output is set, there is a fault. The servo drive is disabled.
    The output is reset after the fault has been acknowledged.

14.3.2.3 Integration of the function blocks

For the consistent data exchange between the PROFINET master and PROFINET slave, the system functions DPRD_DAT (SFC14) and DPWR_DAT (SFC15) must be used for the consistent reading and writing of the telegrams. The system functions can be found in the menu **Extended instructions - Distributed I/O – Others.** The parameters of the blocks must be configured as follows:

Reading of the data by way of DPRD_DAT:

The input data are available as of the address 100hex = 256dez. This is why this address is configured as the start address of the block (LADDR). The output RECORD is used to state the start address within data block 1 (DB1) and the number of bytes received (20). DB1 is then used to store the data from the slave.

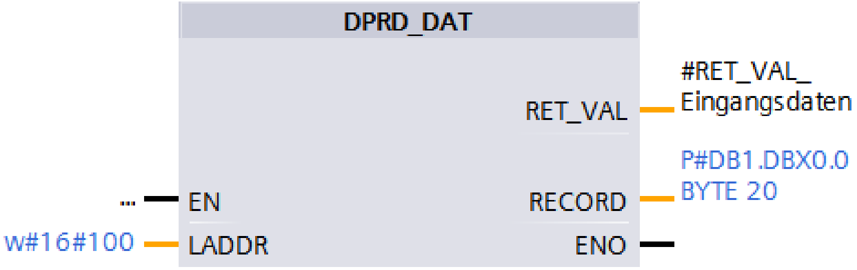


Figure 14-7: System function DPRD_DAT

Parameter	Declaration	Data type	Memory area	Description
LADDR	INPUT	WORD	E, A, M, D, L, const.	Projected start address from the E-area of the module from which data are to be read.
RET_VAL	OUTPUT	INT	E, A, M, D, L	If an error occurs during the execution of the function, the return value will include an error code.
RECORD	OUTPUT	ANY	E, A, M, D, L	Destination area for the read payload. It must have the same length as the length that has been configured for the selected module by way of STEP 7. Only the data type BYTE is permissible.

Writing of the data by way of DPWR_DAT:

Depending on the operating mode (position control or speed control), different data are transferred to the servo drive. The data length of DPWR_DAT must correspond to the configured telegram length since, otherwise, the error code 0x80B1 will be output at the output RET_VAL.

Writing of the data for the position control mode:

The output data are available as of the address 100hex = 256dez. This is why this address is configured as the start address of the block (LADDR). The input RECORD is used to state the start address within data block 2 (DB2) and the number of bytes transferred (16). DB2 is then used to transfer the data to the slave.

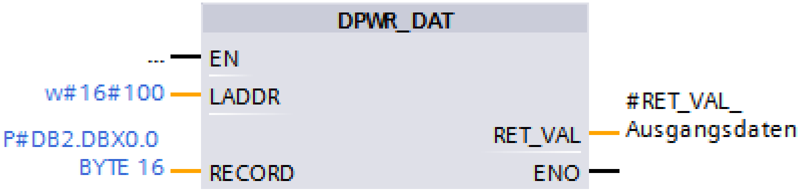


Figure 14-8: System function DPWR_DAT for the position control mode

Writing of the data for the speed control mode:

The output data are available as of the address 100hex = 256dez. This is why this address is configured as the start address of the block (LADDR). The input RECORD is used to state the start address within data block 3 (DB3) and the number of bytes transferred (16). DB3 is then used to transfer the data to the slave.

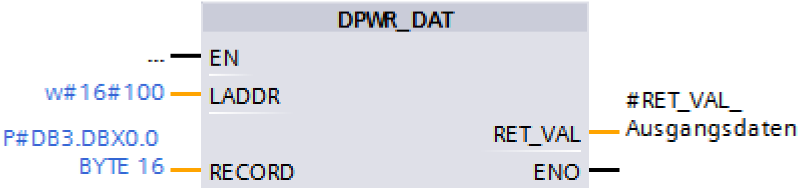
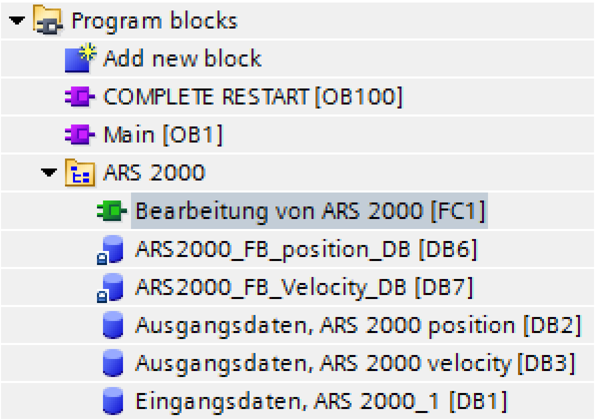


Figure 14-9: System function DPWR_DAT for the speed control mode

Parameter	Declaration	Data type	Memory area	Description
LADDR	INPUT	WORD	E, A, M, D, L, const.	Projected start address from the A-area of the module to which data are to be written.
RECORD	INPUT	ANY	E, A, M, D, L	Source area of the payload to be written. It must have the same length as the length that has been configured for the selected module by way of STEP 7. Only the data type BYTE is permissible.
RET_VAL	OUTPUT	INT	E, A, M, D, L	If an error occurs during the execution of the function, the return value will include an error code.

14.3.3 Example project (overview)

After the example project has been successfully loaded into the TIA portal, the program blocks shown in Figure 14-10 are listed in the **Project tree** area.

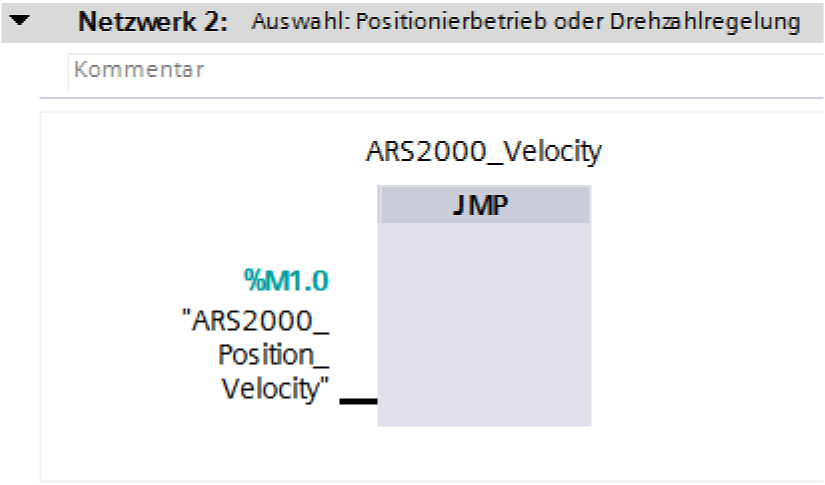


```
Program blocks
    Add new block
    COMPLETE RESTART [OB100]
    Main [OB1]
    ARS 2000
        Bearbeitung von ARS 2000 [FC1]
        ARS2000_FB_position_DB [DB6]
        ARS2000_FB_Velocity_DB [DB7]
        Ausgangsdaten, ARS 2000 position [DB2]
        Ausgangsdaten, ARS 2000 velocity [DB3]
        Eingangsdaten, ARS 2000_1 [DB1]
```

Figure 14-10: Overview of the program blocks of the example project

The actual state of the servo drive is symbolised by the "status word 1" in the input data (DB1). The control of the servo drive is ensured by the control word 1, which is contained in the operating-mode-specific output data (DB2, DB3) and described by the function (FC1) "Bearbeitung von ARS 2000".

The master interface provides the PLC with the input and output data of the servo drive in defined I/O areas. These data are read out of the slave via DPRD_DAT (SFC14) and written into the slave by way of DPWR_DAT (SFC15). Depending on the selected operating mode, the data are stored in the data blocks that are defined for the respective operating mode. The operating mode can be configured by setting the flag "ARS2000_Position_Velocity". The flag is contained in network 2 of the FC1 "Bearbeitung von ARS 2000"; see also the following Figure 14-11.



Netzwerk 2: Auswahl: Positionierbetrieb oder Drehzahlregelung

Kommentar

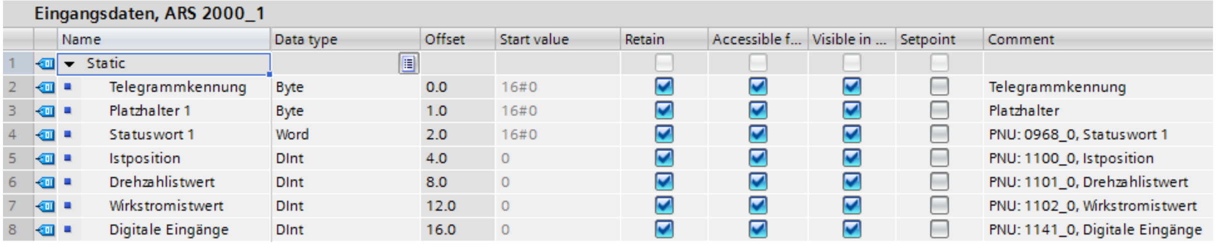
Figure 14-11: Selection of the position control mode or speed control mode

The DBs are used as I/O modules for the actuation of the states of the servo drives and for the transfer of the data that are to be transferred to the servo drive.

14.3.3.1 Function blocks and data blocks

Input data:

DB1 Input data Actual values of the servo drive (for both operating modes)



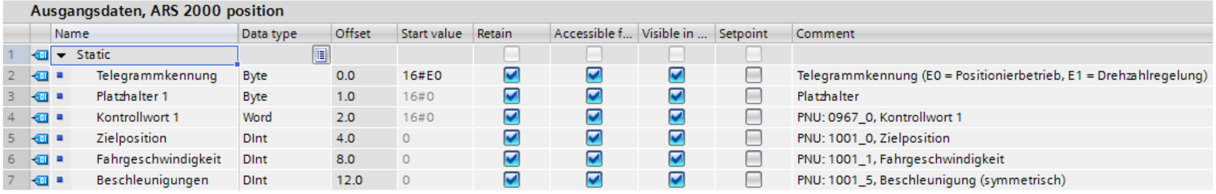
Eingangsdaten, ARS 2000_1

	Name	Data type	Offset	Start value	Retain	Accessible f...	Visible in ...	Setpoint	Comment
1	Static				□	□	□	□	
2	Telegrammkennung	Byte	0.0	16#0	☑	☑	☑	□	Telegrammkennung
3	Platzhalter 1	Byte	1.0	16#0	☑	☑	☑	□	Platzhalter
4	Statuswort 1	Word	2.0	16#0	☑	☑	☑	□	PNU: 0968_0, Statuswort 1
5	Istposition	DInt	4.0	0	☑	☑	☑	□	PNU: 1100_0, Istposition
6	Drehzahlistwert	DInt	8.0	0	☑	☑	☑	□	PNU: 1101_0, Drehzahlistwert
7	Wirkstromistwert	DInt	12.0	0	☑	☑	☑	□	PNU: 1102_0, Wirkstromistwert
8	Digitale Eingänge	DInt	16.0	0	☑	☑	☑	□	PNU: 1141_0, Digitale Eingänge

Figure 14-12: Input data

Position control:

FB41	FB_position	Function block for the position control mode
DB6	Instance DB for FB41	The data block of the internal variable, which is part of the function block
DB2	Output data	Data for the position control mode



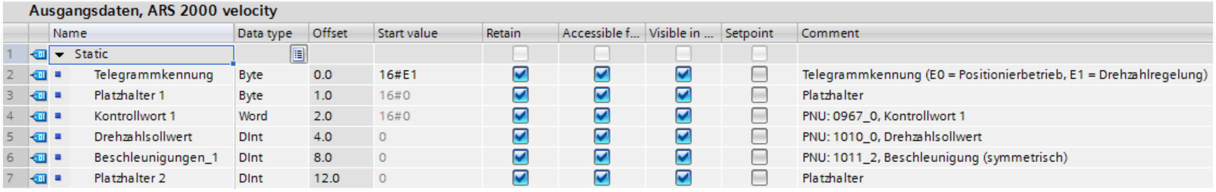
Ausgangsdaten, ARS 2000 position

	Name	Data type	Offset	Start value	Retain	Accessible f...	Visible in ...	Setpoint	Comment
1	Static				□	□	□	□	
2	Telegrammkennung	Byte	0.0	16#E0	☑	☑	☑	□	Telegrammkennung (E0 = Positionierbetrieb, E1 = Drehzahlregelung)
3	Platzhalter 1	Byte	1.0	16#0	☑	☑	☑	□	Platzhalter
4	Kontrollwort 1	Word	2.0	16#0	☑	☑	☑	□	PNU: 0967_0, Kontrollwort 1
5	Zielposition	DInt	4.0	0	☑	☑	☑	□	PNU: 1001_0, Zielposition
6	Fahrgeschwindigkeit	DInt	8.0	0	☑	☑	☑	□	PNU: 1001_1, Fahrgeschwindigkeit
7	Beschleunigungen	DInt	12.0	0	☑	☑	☑	□	PNU: 1001_5, Beschleunigung (symmetrisch)

Figure 14-13: Position control mode output data

<u>Speed control:</u>

FB43	FB_velocity	Function block for the speed control mode
DB43	Instance DB for FB43	The data block of the internal variable, which is part of the function block
DB3	Output data	Data for the speed control mode



Ausgangsdaten, ARS 2000 velocity

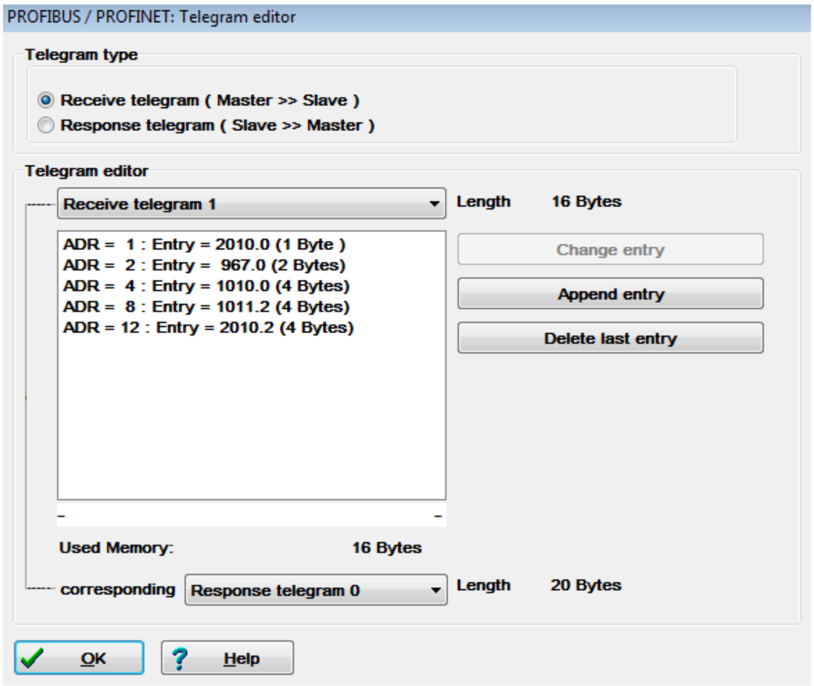
	Name	Data type	Offset	Start value	Retain	Accessible f...	Visible in ...	Setpoint	Comment
1	Static				□	□	□	□	
2	Telegrammkennung	Byte	0.0	16#E1	☑	☑	☑	□	Telegrammkennung (E0 = Positionierbetrieb, E1 = Drehzahlregelung)
3	Platzhalter 1	Byte	1.0	16#0	☑	☑	☑	□	Platzhalter
4	Kontrollwort 1	Word	2.0	16#0	☑	☑	☑	□	PNU: 0967_0, Kontrollwort 1
5	Drehzahlsollwert	DInt	4.0	0	☑	☑	☑	□	PNU: 1010_0, Drehzahlsollwert
6	Beschleunigungen_1	DInt	8.0	0	☑	☑	☑	□	PNU: 1011_2, Beschleunigung (symmetrisch)
7	Platzhalter 2	DInt	12.0	0	☑	☑	☑	□	Platzhalter

Figure 14-14: Speed control output data

14.3.4 Connection of the ARS 2000 servo drive

14.3.4.1 Receive telegram in the case of a speed control example project

In this case, the receive telegram 1 must be configured accordingly on the servo drive side. The following illustration in Figure 14-15 shows the parameterisation of the receive telegram that is required for the example project. If a communication has been established between the master and slave, additional diagnostics information can be displayed by way of the button "Change entry". The system will display the actual length of the telegram from the master to the slave, which has been configured by the master.



PROFIBUS / PROFINET: Telegram editor

Telegram type

● Receive telegram (Master >> Slave)
○ Response telegram (Slave >> Master)

Telegram editor

Receive telegram 1 — Length 16 Bytes

```
ADR =  1 : Entry = 2010.0 (1 Byte )
ADR =  2 : Entry =  967.0 (2 Bytes)
ADR =  4 : Entry = 1010.0 (4 Bytes)
ADR =  8 : Entry = 1011.2 (4 Bytes)
ADR = 12 : Entry = 2010.2 (4 Bytes)
```

Used Memory: 16 Bytes

corresponding Response telegram 0 — Length 20 Bytes

Figure 14-15: Telegram structure for the speed control mode

The following table describes the telegram structure:

Addres s	Content (parameter number)	Description
0	Identifier (= 0xE1)	Fixed identifier
1	8-bit placeholder (PNU 2010.0)	free
2	Control word 1 (PNU 967.0)	Control word for the device control; it must be located at this address
4	Speed setpoint (PNU 1010.0)	Speed setpoint; specified in the physical speed unit that has been set for PROFINET
8	Acceleration (PNU 1011.2)	Combination of the values for acceleration and deceleration; specified in the physical acceleration unit that has been set for PROFINET.
12	Placeholder (PNU 2010.2)	The placeholder ensures an identical telegram length for both operating modes. This enables the switching between the operating modes without the need for a reconfiguration.

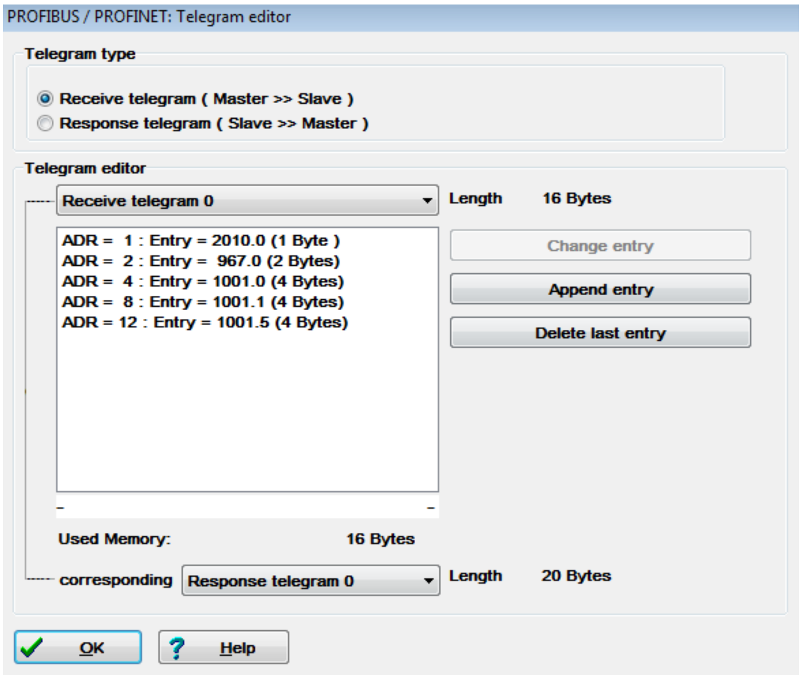
The total length of this telegram is 16 bytes. When the slave is integrated into the hardware configuration of the master, the module

8 Words Output

must be selected for the output data and connected to one of the ports.

14.3.4.2 Receive telegram in the case of a position control example project

In this case, the receive telegram 0 must be configured accordingly on the servo drive side. The following illustration in Figure 14-16 shows parameterisation of the receive telegram that is required for the example project. If a communication has been established between the master and slave, additional diagnostics information can be displayed by way of the button "Change entry". The system will display the actual length of the telegram from the master to the slave, which has been configured by the master.



PROFIBUS / PROFINET: Telegram editor

Telegram type

● Receive telegram (Master >> Slave)
○ Response telegram (Slave >> Master)

Telegram editor

Receive telegram 0 — Length 16 Bytes

```
ADR =  1 : Entry = 2010.0 (1 Byte )
ADR =  2 : Entry =  967.0 (2 Bytes)
ADR =  4 : Entry = 1001.0 (4 Bytes)
ADR =  8 : Entry = 1001.1 (4 Bytes)
ADR = 12 : Entry = 1001.5 (4 Bytes)
```

Used Memory: 16 Bytes

corresponding Response telegram 0 — Length 20 Bytes

Figure 14-16: Telegram structure for the position control mode

The following table describes the telegram structure:

Address	Content (parameter number)	Description
0	Identifier (= 0xE0)	Fixed identifier
1	8-bit placeholder (PNU 2010.0)	free
2	Control word 1 (PNU 967.0)	Control word for the device control; it must be located at this address
4	Target position (PNU 1001.0)	Target position; specified in the physical position unit that has been set for PROFINET
8	Speed of movement (PNU 1001.1)	Speed of movement during the positioning process; specified in the physical speed unit that has been set for PROFINET
12	Acceleration (PNU 1001.5)	Combination of the values for acceleration and deceleration; specified in the physical acceleration unit that has been set for PROFINET.

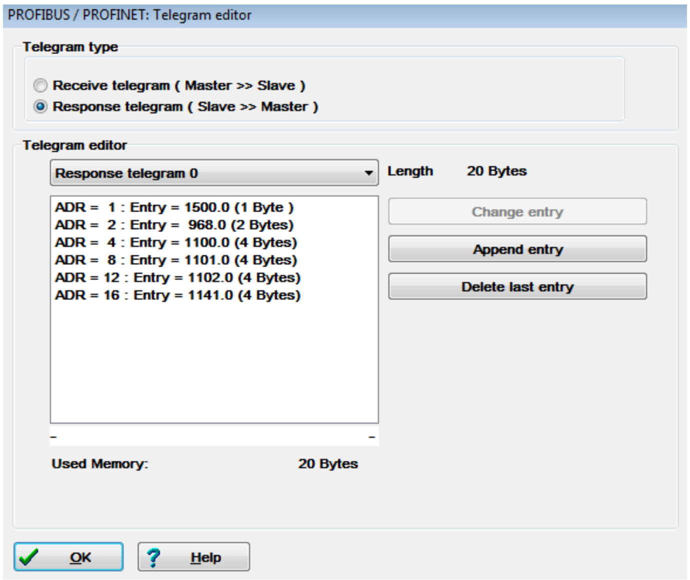
The total length of this telegram is 16 bytes. When the slave is integrated into the hardware configuration of the master, the module

8 Words Output

must be selected for the output data and connected to one of the ports.

14.3.4.3 Response telegram for the example projects

The example project uses the same actual value data that are transferred from the slave to the master. This is why the same response telegram 0 is configured on the servo drive side for both receive telegrams 0 and 1. The following illustration shows the parameterisation that is required for the example project. If a communication has been established between the master and slave, additional diagnostics information can be displayed by way of the button "Change entry". The system will display the actual length of the telegram from the slave to the master, which has been configured by the master.



PROFIBUS / PROFINET: Telegram editor

Telegram type

○ Receive telegram (Master >> Slave)
● Response telegram (Slave >> Master)

Telegram editor

Response telegram 0

Length 20 Bytes

ADR = 1 : Entry = 1500.0 (1 Byte)
ADR = 2 : Entry = 968.0 (2 Bytes)
ADR = 4 : Entry = 1100.0 (4 Bytes)
ADR = 8 : Entry = 1101.0 (4 Bytes)
ADR = 12 : Entry = 1102.0 (4 Bytes)
ADR = 16 : Entry = 1141.0 (4 Bytes)

Used Memory: 20 Bytes

Figure 14-17: Response telegram structure

The following table describes the telegram structure:

Address	Content (parameter number)	Description
0	Identifier (= 0xF0)	Fixed identifier
1	Operating mode (PNU 1500.0)	Current operating mode of the servo drive
2	Status word 1 (PNU 968.0)	Status word for the device control; it must be located at this address
4	Actual position (PNU 1100.0)	Current actual position; specified in the physical position unit that has been set for PROFINET
8	Actual speed (PNU 1101.0)	Current actual speed; specified in the physical speed unit that has been set for PROFINET
12	Actual active current (PNU 1102.0)	Current active current in per mill referred to the nominal motor current
16	Status of the digital inputs (PNU 1141.0)	Current status of the digital inputs; for information about the bit assignment see the description of the PNU

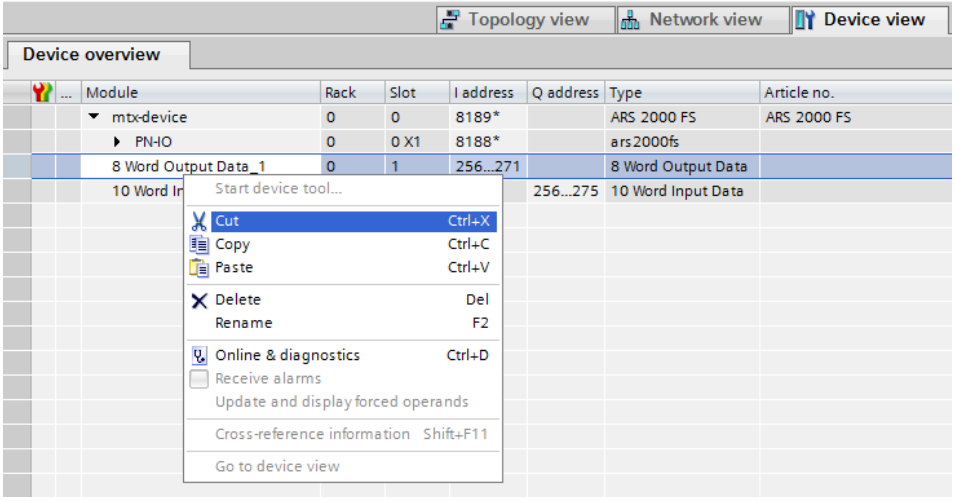
The total length of this telegram is 20 bytes. When the slave is integrated into the hardware configuration of the master, the module

10 Words Input

must be selected for the input data and connected to one of the ports.

14.3.5 Diagnostics

If there is a fault of the servo drive, it can be read out in plain text. To do so, select the **Device view**, right-click the wrench icon, and select the menu item **Online & Diagnostics**. All of the pending faults are listed under the menu item **Channel diagnostics** in the diagnostics window.



...	Module	Rack	Slot	I address	Q address	Type	Article no.
	mtx-device	0	0	8189*		ARS 2000 FS	ARS 2000 FS
	PN-IO	0	0 X1	8188*		ars2000fs	
	8 Word Output Data_1	0	1	256...271		8 Word Output Data	
	10 Word Ir				256...275	10 Word Input Data	

Figure 14-18: Activating the PROFINET diagnostics function