On-line Command Processing
Using

CAN-Bus

for WDP3-014 and WDP3-018
and Series 300 Units

Doc. no. 222.271/DGB

Ident. no.: 00441110950

Edition: d027 05.02

Proposals
Improvements

Berger Lahr GmbH & Co. KG

Breslauer Str. 7
Postfach 1180

D-77901 Lahr

CAN-Bus

Edition: d027 05.02
Doc. no. 222.271/DGB

Sender:

Name:

Company/department:

Address:

Telephone no.:

Please inform us, using this form, if you have discovered any errors when reading this document.

We should also appreciate any new ideas and proposals.

Proposal and/or improvements:

Table of contents

1 Purpose and contents of this documentation

The BERGER LAHR WDP3-01X and Series 300 controllers can be provided with an interface for the CAN-Bus.

This documentation describes application and operation of a BERGER LAHR controller with CAN-Bus capability within a CAN-Bus network:

– Basic information on the CAN-Bus technology for BERGER LAHR controllers
– Setting up a controller in a CAN-Bus network
– Communication between a CAN-Bus station and a BERGER LAHR controller
– Functionality of a controller with CAN-Bus interface

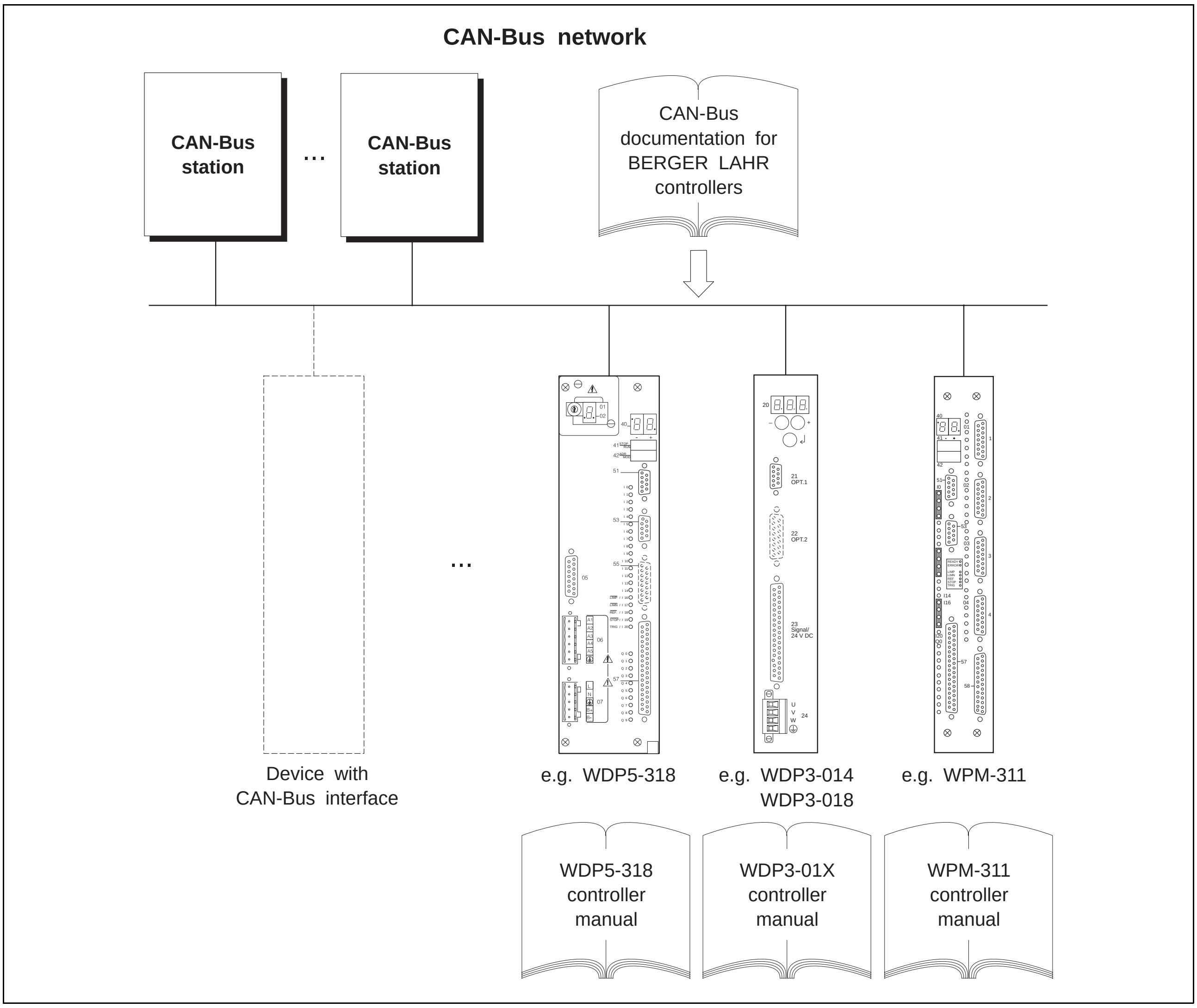


Fig. 1-1 CAN-Bus network system environment

Purpose and contents of this documentation

NOTE
This documentation applies to all BERGER LAHR controllers which are available with a CAN-Bus interface:
Series 300 units (e.g. WDP5-318, WPM-311)
as well as the WDP3-014 and WDP3-018 units
The hardware description of the controllers is included in the appropriate controller manuals.

The present documentation is structured as follows:

Basic information

Chapter 2 gives information on the essential features of CAN-Bus networks. It describes the design and the principle of data transmission in a CAN-Bus network.

Setup

Chapter 3 describes the procedure for setting up a BERGER LAHR controller in a CAN-Bus network.

Principle of communication

Chapter 4 explains the principle of communication between a CAN-Bus station and a BERGER LAHR controller.

Functionality of controllers with CAN-Bus

Chapter 5 describes the functional characteristics of BERGER LAHR controllers with CAN-Bus capability which can be utilized in a CAN-Bus network. This chapter explains concepts and relationships which must be understood in order to be able to use a BERGER LAHR controller as a drive unit in a CAN-Bus network.

Programming examples

Chapter 6 contains programming instructions which can be used for creating application programs for CAN-Bus stations. The procedure for creating programs is illustrated by way of programming examples.

Error handling

Chapter 7 describes potential errors and lists error codes.

Command descriptions

Chapters 8 and 9 contain summary descriptions of write and read commands, respectively.

Data structures

The appendix summarizes the command reference lists and the data structures used in command and data transmission between CAN-Bus stations and BERGER LAHR controllers.

Symbols used

The following symbols and safety notes are used in this documentation and should be observed.

ATTENTION
Special attention is drawn to potentially inappropriate use involving the risk of consequential damage.

NOTE
Important or additional information on the device or on the documentation.

This symbol identifies application examples.

2 Basic information on the CAN-Bus technology

2.1 Introduction

The CAN-Bus was originally developed for fast and economic data transmission in automotive engineering.
Today, the CAN-Bus is also used in industrial automation and development is in progress into communication between field equipment.
The CAN-Bus is a standardized open bus architecture which can be used for communication between devices, sensors and actuators of different manufacturers.
BERGER LAHR controllers of the Series 300 and the WDP3-014 and WDP3-018 controllers can be equipped with a CAN-Bus interface for integration into a CAN-Bus network.
These controllers can operate with the simple CAN-Bus protocol (7-layer ISO model, layers 1 and 2) as well as with the application-specific CAL interface (CAN Application Layer).

2.2 Structure and topology

A CAN-Bus network consists of several network stations (network nodes) which are attached to a bus cable. Communication between the network stations is effected by serial data transmission.
Field devices with CAN-Bus capability of different manufacturers can serve as network nodes. Examples of network nodes are PLC controllers, sensors, actuators and positioning controllers manufactured by BERGER LAHR.
The bus cable must be terminated on both ends with a 120 Ω terminator.

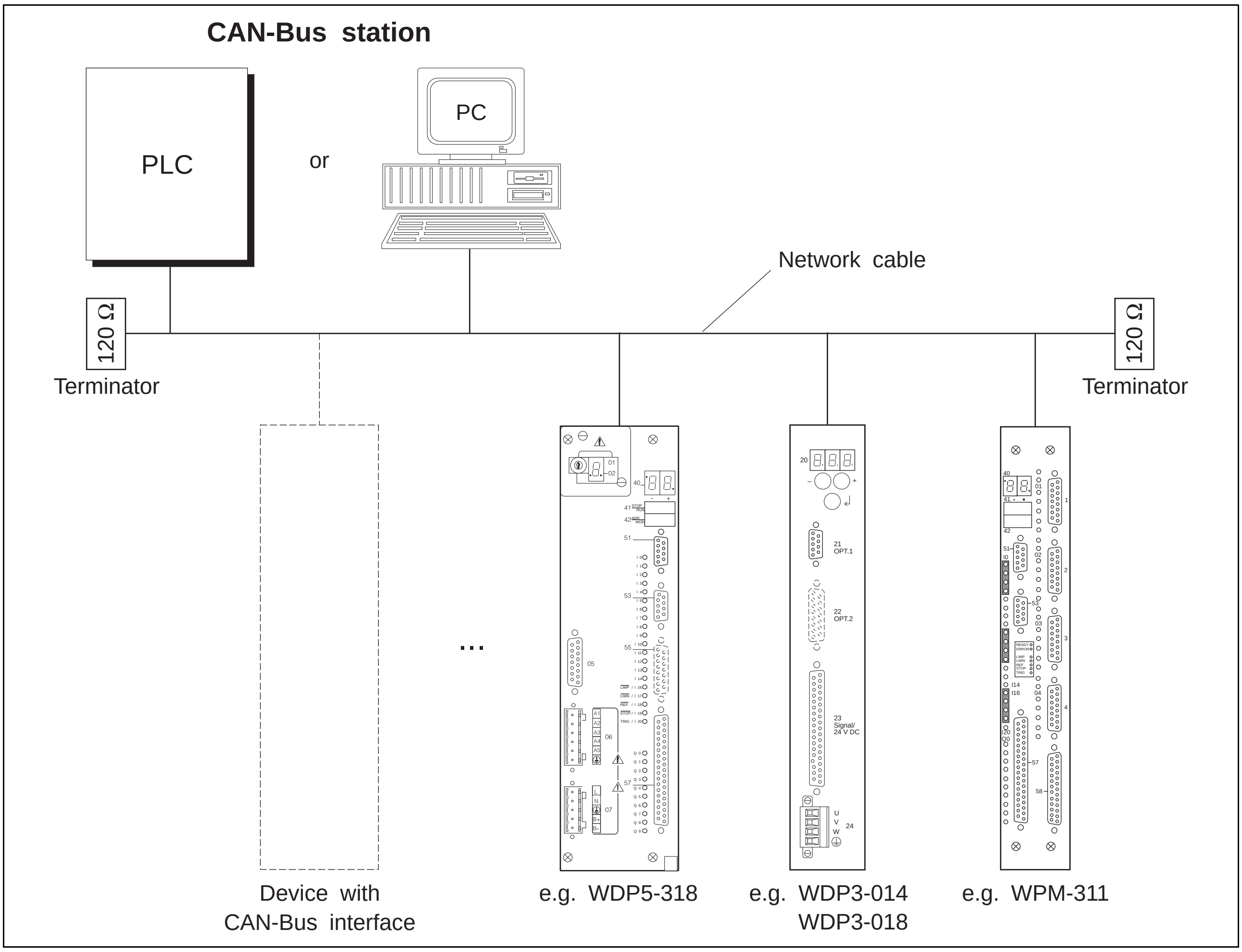


Fig. 2-1 BERGER LAHR controllers in a CAN-Bus network

2.3 Communication

In a CAN-Bus network, each network station can exchange data with any other network station. Data exchange, or communication, is effected using communication objects (COB).

Communication objects

Each communication object consists of an identifier (object name) and the actual data.

Identifier	Data (1 – 8 bytes)

Identifier The identifier is a unique name for the communication object in the CAN-Bus network. The identifier defines the destination device and the purpose of the data contained in the communication object.

Data The actual data contain application-specific data (e.g. measured values, positions, commands, etc.).

Communication objects for BERGER LAHR controllers

Two communication objects are used by BERGER LAHR controllers for communication in a CAN-Bus network:

– Command objects
– Data objects

Both objects contain exactly **8 bytes** of actual data each.

Command object

The command object contains commands. A BERGER LAHR controller can receive command objects from another network station and execute the commands contained therein.
A command consists of the command number and the command data. A distinction is made between write commands and read commands.

Write commands
Read commands

Write commands initiate certain functions on the controller. Read commands request data from a controller.

Examples of commands are: Axis positioning (write command) or determining the current position of an axis (read command).

The command object consists of:

Identifier	Command number (2 bytes)	Command data (6 bytes)

→

Command object to the BERGER LAHR controller

Data object

The data object contains the data of a BERGER LAHR controller for monitoring and error handling by the network station which sent a command object to the controller.

Standard data
Read data

Data always consist of the standard data (axis signals and axis status) and the read data. Read data are requested from a controller using a read command. Read data are always transmitted in response to the previously sent read command. This is still valid when a write command has been sent.

The data object consists of:

Identifier	Standard data	Read data

←

Data object from the BERGER LAHR controller

Communication principle

A BERGER LAHR controller responds to each command (command object) by sending the corresponding data (data object).
A controller is only ready for receiving a command when it has acknowledged the previous command by sending a data object.

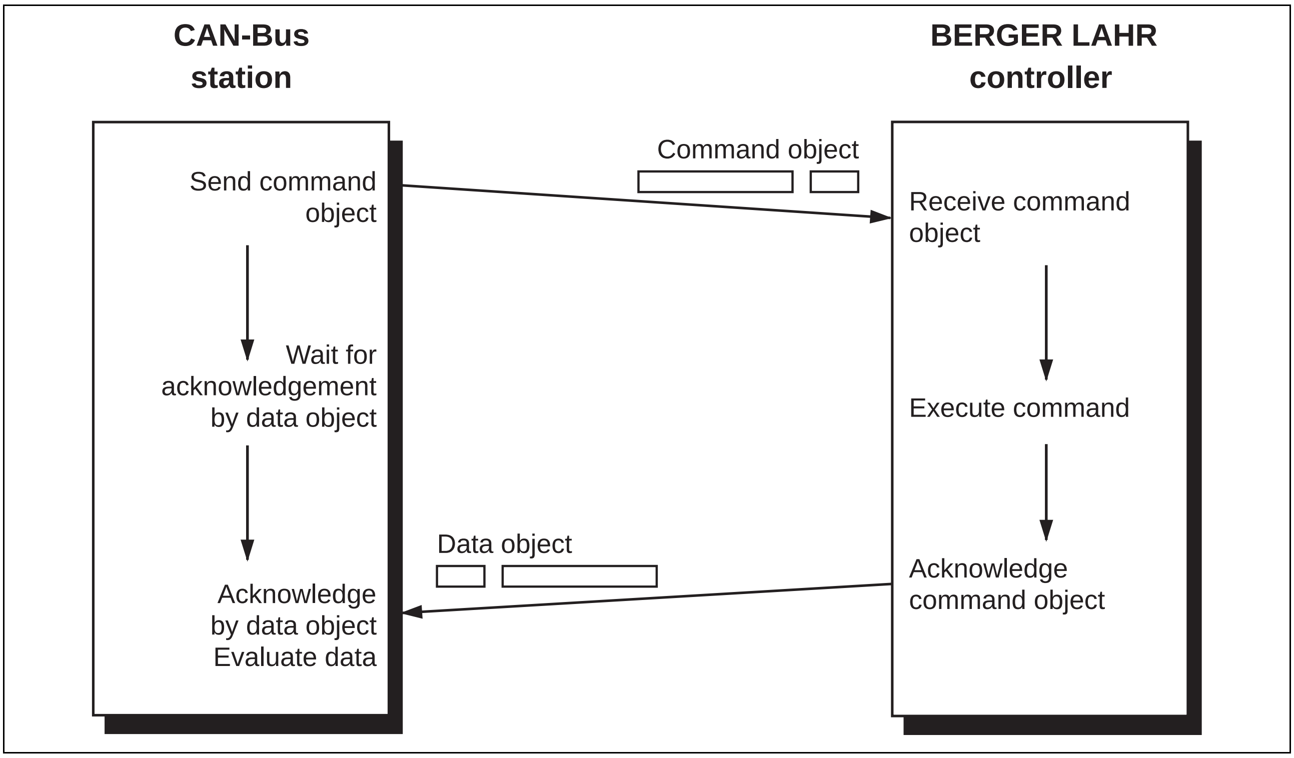


Fig. 2-2 Principle of communication

Identifier

One command object and one data object exist for each BERGER LAHR controller in a CAN-Bus network.
Command objects and data objects are distinguished by the identifier or object name.

The identifier assignment is based on the device address of the controller and/or the CAN-Bus protocol used (see chapter 2.4).

2.4 CAN-Bus operating modes

BERGER LAHR controllers with a CAN-Bus interface can work in two different CAN-Bus operating modes (protocols):

1. Simple CAN-Bus protocol
2. CAL protocol

NOTE
The operating mode can be selected on the controller front panel; see chapter 3.

2.4.1 Simple CAN-Bus protocol

In this operating mode, the controller uses the communication protocols specified in the ISO/DIS 11-898 and ISO/DIS 11-519-1 standards. These are the protocols of layers 1 and 2 of the ISO 7-layer model.

NOTE
The simple CAN-Bus protocol does not implement any network management functions. The network is not monitored, i.e. failure of a network station cannot be detected automatically. This is only possible by activating bus monitoring with the TIMEOUT command. With the CAN-Bus protocol, the identifiers of the communication objects are predefined.

Communication objects

Two communication objects are available for communication with a BERGER LAHR controller:

– Command objects
– Data objects

Identifier

The identifiers of the two objects must be determined from the device address of the controller to which the objects are sent.

$\textbf{Identifier}_{\textbf{Command object}}$ = Controller device address x 16

$\textbf{Identifier}_{\textbf{Data object}}$ = $\text{Identifier}_{\text{Command object}}$ + 8

Example 1:
A network station is to send a command to the controller with the device address 3:

$\text{Identifier}_{\text{Command object}}$ = 3 x 16 = 48
$\text{Identifier}_{\text{Data object}}$ = 48 + 8 = 56

The network station must send the command object to the controller using the identifier 48.
The network station can read the data with which the controller responds to the command from the data object with the identifier 56.

Example 2:
Three BERGER LAHR controllers are to be used in a CAN-Bus network. Refer to the following table for the identifiers of the command and data objects.

Object	**Device address**	**Identifier**
Command object for controller 1	1	16
Command object for controller 2	3	48
Command object for controller 3	7	112
Data object for controller 1	1	24
Data object for controller 2	3	56
Data object for controller 3	7	120

2.4.2 CAL protocol

CAL: CAN Application Layer

The application-specific CAL interface offers important network management functions for communication with a BERGER LAHR controller in a CAN-Bus network.
The network management performs the following tasks:

– Network configuration
– Device login/logout
– Link monitoring
– Automatic identifier assignment

The following data elements of the BERGER LAHR implementation of the CAN-Bus are relevant for the CAL interface:

CAL classification

CMS CAN Message Specification
NMT Network management class 2
DBT Fully implemented
LMT Not implemented

Node name

The node name of a BERGER LAHR controller in a CAN-Bus network is determined from the controller’s device address.

“BL_ _XXX”
“XXX” is a three-digit number, where:
“XXX” = Device address

Example:
Node name of a BERGER LAHR controller with device address 4:
“XXX” = 4
Node name: “BL_ _004”

Communication objects

The object names of the two communication objects (command object and data object) are also determined from the device address of the controller to which the objects are to be sent:

Command object: “#BL_KOM_XXX”
Data object: “#BL_DAT_XXX”
The three-digit number “XXX” is determined as described in chapter 2.4.1.

Example:
Object names for a BERGER LAHR controller with device address 5:
“XXX” = 5
Name of command object: “#BL_KOM_005”
Name of data object: “#BL_DAT_”

The following table shows the CAL-specific object description of the two BERGER LAHR objects.

Object name	#BL_KOM_XXX	#BL_DAT_XXX
Data type	ARRAY[8] of BYTE	ARRAY[8] of BYTE
User type	Client	Server
Access	write only	write only

Identifier

Object-specific identifiers are automatically generated from the names of the command objects and data objects of the bus stations and communicated to the latter during network initialization.

3 Setting up a controller

To set up a BERGER LAHR controller in a CAN-Bus network, the following steps are required:

1. Connect the controller to the CAN-Bus network cable.
2. Switch on the controller's voltage supply.
3. Set the device address on the controller front panel.
4. Set the baud rate on the controller front panel.
5. Set the CAN-Bus operating mode on the controller front panel.

The device address, the baud rate and the CAN-Bus operating mode must be set by parameters on the controller front panel. The parameters to be used depend on the controller type (see table).

NOTE
On Series 300 units, the CAN-Bus interface can be installed either in adapter slot 51 or in adapter slot 53 (see controller manual). The parameters 61, 62, 63 are used for the settings related to slot 51, the parameters 71, 72, 73 for the settings related to slot 53.

Parameter	Series 300 units	WDP3-01X
Device address	61*/71**	P60
Baud rate	62*/72**	P61
CAN-Bus operating mode	63*/73**	P62

* Interface in slot 51
** Interface in slot 53

NOTE
The setting procedure for the device parameters via front panel is described in the appropriate controller manual.

1. **Connect the controller to the CAN-Bus network cable.**

DANGER
The supply voltage for the devices must be disconnected whenever wiring work is carried out.

ATTENTION
Wiring work may only be carried out in accordance with VDE 0105 by trained personnel.

ATTENTION
Free, unassigned pins must not be wired.

The connection to the network cable is established via a 9-pin CAN-Bus interface for each BERGER LAHR controller with CAN-Bus capability. This interface is a modified RS 485 interface.

NOTE
On Series 300 units, the CAN-Bus interface can be installed either in adapter slot 51 or in adapter slot 53 (see controller manual).

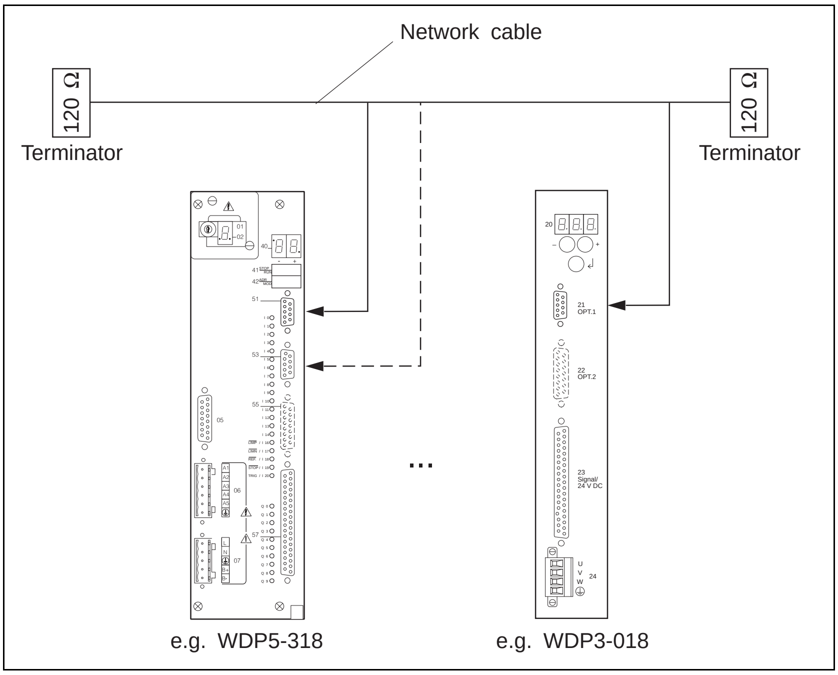


Fig. 3-1 Connection diagram

NOTE
The maximum bus cable length depends on the number of stations, the internal and external signal transmission times and the baud rate.
The following applies: The higher the baud rate, the shorter the required bus cable.

Pin	Signal	Meaning
1	–	–
2	CAN_LOW	Inverted data line
3	GND	Ground
4	–	–
5	–	–
6	GND	Ground
7	CAN_HIGH	Data line
8	–	–
9	–	–

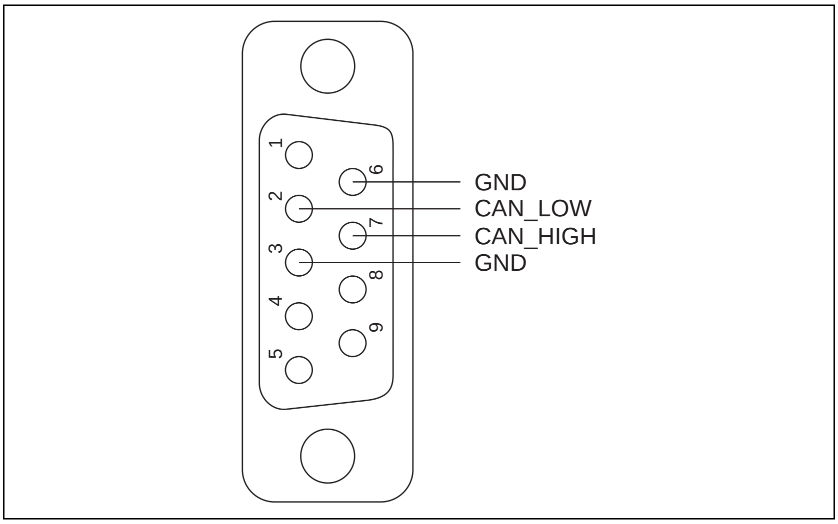


Fig. 3-2 Interface plug connector on the device

2. Switch on the controller's voltage supply.

The following steps require that the controller supply voltage be connected.
Connect the voltage supply (see controller manual).

3. Set the device address on the controller front panel.

In order to identify the controller in a network, the device address must be set on the controller front panel.

NOTE
The identifiers and object names for communication in the CAN-Bus network are derived from the device address.

Parameter	Device address	Setting
P60	Device address for operating a WDP3-01X controller	0 to 126*
61	Device address for operating a Series 300 controller via interface adapter slot 51	
71	Device address for operating a Series 300 controller via interface adapter slot 53	

* Default

4. Set the baud rate on the controller front panel.

Parameter	Baud rate	Setting
P61	Baud rate for operating a WDP3-01X controller	01 = 500 kbauds 02 = 250 kbauds 03 = 125 kbauds* 04 = 100 kbauds 05 = 50 kbauds 06 = 20 kbauds 07 = 10 kbauds
62	Baud rate for operating a Series 300 controller via interface adapter slot 51	
72	Baud rate for operating a Series 300 controller via interface adapter slot 53	

* Default

5. Set the CAN-Bus operating mode on the controller front panel.

Parameter	CAN-Bus operating mode	Setting
P62	For operating a WDP3-01X controller Simple CAN-Bus protocol CAL protocol	 01* 02
63	For operating a Series 300 controller via adapter slot 51 Simple CAN-Bus protocol CAL protocol	 01* 02
73	For operating a Series 300 controller via adapter slot 53 Simple CAN-Bus protocol CAL protocol	 01* 02

* Default

4 Communication with a controller with CAN-Bus capability

4.1 Contents of this chapter

Data exchange

This chapter describes the data exchange between a CAN-Bus station and a BERGER LAHR controller in a CAN-Bus network.

This information is required in order to be able to integrate a controller into a CAN-Bus network and program the application software for the CAN-Bus station accordingly.

Functional scope

The CAN-Bus interface has been defined as a standardized interface for all BERGER LAHR controllers with CAN-Bus capability. The difference between the controllers consists in their scope of functions rather than the way they exchange data and commands.

The functional scope of a controller and the valid commands are described in the appropriate controller manuals and in the command summaries in chapters 8 and 9.

Controller

The term controller denotes a BERGER LAHR controller with CAN-Bus interface here.

CAN-Bus station

The term CAN-Bus station denotes any device in a CAN-Bus network. Every station can send commands to or receive commands from any other station in the CAN-Bus network. This means that any station can operate as a master or slave if it has been configured as a master or slave.

4.2 Communication principle

The communication between CAN-Bus stations is effected with commands and data.

The station sends a command to the controller, and the controller responds to a command (acknowledges it) by sending its data. The controller must first acknowledge (respond to) a command before the station is allowed to send another command to the controller.

Write commands
Read commands

The commands either initiate the execution of functions on a controller (write commands) or request the controller to send certain data to the station (read commands).

Standard data

Standard data contain information which can be used for monitoring status conditions (axis status, axis signals) and error handling (error codes).

Read data

With read commands, read data selected by the read command are transmitted in addition to the standard data (fig. 4-1).

Acknowledgement information is transmitted in the standard data.

When a read command has been transmitted, the read data of the previous read command are sent after any subsequent write command.

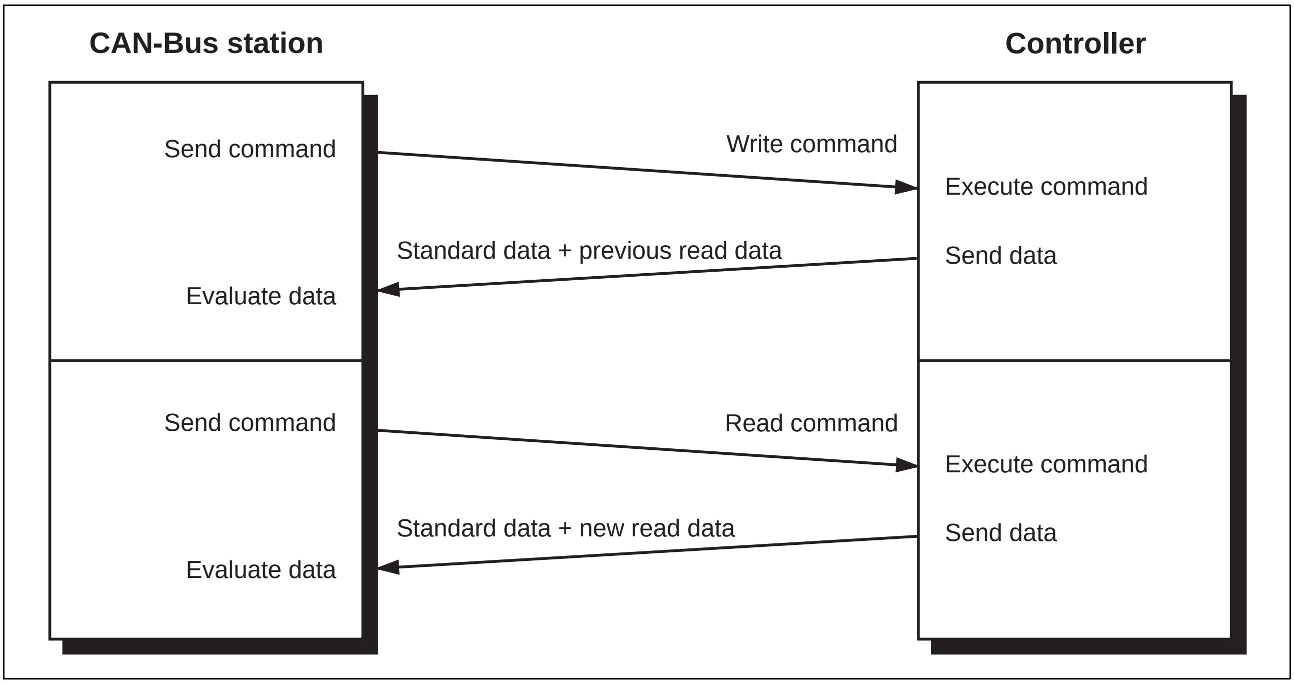


Fig. 4-1 Principle of communication

4.3 Data transmission format

The data and commands are transmitted in an 8-byte data structure.

The following illustration shows the data transmission format used for transmitting data and commands between a station and a controller.

Low address | High address

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
Word 1		Word 2		Word 3		Word 4	
Double word 1				Double word 2			

4.4 Commands

The functions of a controller are accessed using commands. Commands are sent from the station to the controller and interpreted and executed by the latter.

Two types of commands are used for BERGER LAHR controllers with CAN-Bus capability:

- Write commands
- Read commands

A command structure into which the encoded command and the associated parameters are loaded is used for transmitting commands.

4.4.1 Write commands

A write command initiates a function in a controller. A write command can also be used to transfer data to the controller as parameters of the command.

Example:
The *POS x1, 2000* command positions axis 1 of a controller to position 2000.

4.4.2 Read commands

A read command instructs the controller to transmit specific data to the station. These data are called read data.

Example:
The *GETPOS x1, actual* command requests the current position of axis 1 of a controller (fig. 4-2).

The read data are transmitted repeatedly after any subsequent write command until a new read command is sent.

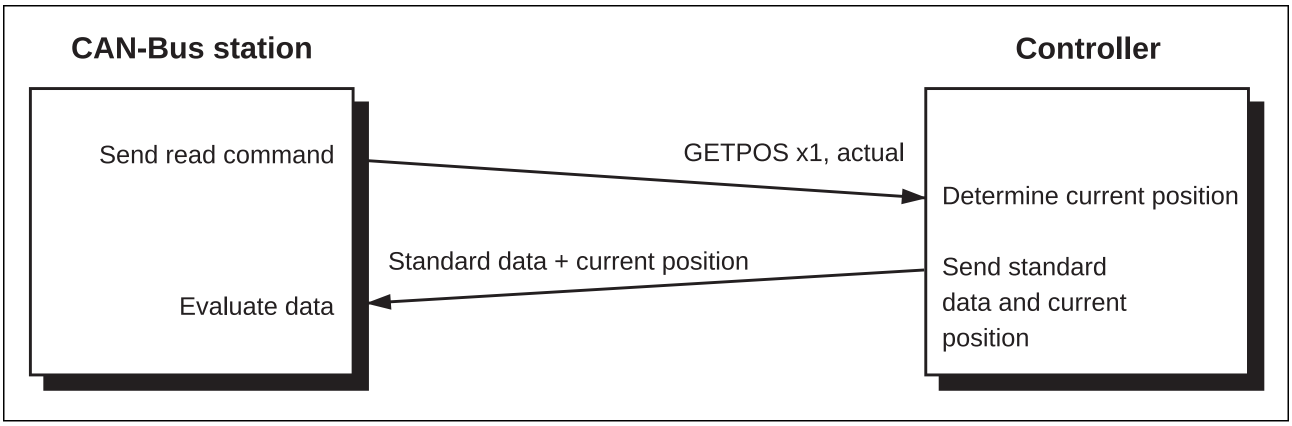


Fig. 4-2 Read command

4.4.3 Command structure

Commands are transmitted to the controller in a defined command structure.

The command structure consists of:

Command number (2 bytes)

Command data (6 bytes)

Word 1	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
Command number	Command data 6 bytes					

Command number (2 bytes)

Each command is assigned a number (hexadecimal value). This number is entered in the first word of the command structure.
For the command numbers, refer to the command tables in chapters 8 and 9.

Example:
The GETMODE command has the number 0058_h. This value must be loaded into the first word of the command structure.

0058_h

Command data (6 bytes)

Command data are parameters pertaining to the command.
Up to 6 bytes of parameters can be transmitted with a command. Unused data bytes must be padded with zeros. Command parameters are always left-justified (starting with byte 3) in the command structure, i.e. next to the command number.

NOTE
Commands and data are transmitted in code. A readable format is used here to facilitate understanding.

Sample command		**Function**
`SETPOS`	`x1, 1000`	Set current position of axis 1

– Command syntax

Command name	Axis identifier	Position value
`SETPOS`	`x1`	`1000`

– Command number 0005_h

The command number is entered right-justified in the first word of the command structure.

– Parameter for axis identifier x1

Axis selection is effected using an axis identifier. This is a hexadecimal value which is entered into the second word of the command structure.
The following axis identifiers are valid:

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

The axis identifiers x2, x3 and x4 can only be used for the WPM-311 multi-axis controller.

– Parameter for position 1000 ($3E8_h$)

Two words (4 bytes) in the command structure are reserved for the position value. The value contained in these 4 bytes is interpreted as a DINT value by the controller.

– Command structure as a memory representation

Command no.	Axis identifier (WORD)	Position (DINT)
0005_h	7800_h	1000 (0000 $03E8_h$)

4.5 Standard data and read data

The 8-byte data structure sent by a controller to another CAN-Bus station contains two types of data:

Standard data Standard data are used for monitoring the controller and error analysis by the CAN-Bus station.

Read data Read data are data originating from the controller, e.g. position and speed. A read command selects the read data to be transmitted by the controller. The read data are transmitted repeatedly after any subsequent write command until a new read command is sent.

4.5.1 Standard data

Standard data and read data are transmitted to the station after any command or by request using the GETDATA command.

The standard data comprise the axis status (2 bytes) and the axis signals (2 bytes). These data are always included in the data structure (8 bytes) which is sent to the station.

The remaining 4 bytes are used for transmitting the read data selected by the station with a read command; see chapter 4.5.2.

Axis status (WORD)	Axis signals (WORD)	(4 bytes of read data)

When an error occurs, an error code is transmitted in addition to the axis status and the axis signals (see chapter 7.3).

Axis status (WORD)	Axis signals (WORD)	Error code (WORD)	(2 bytes unused)
See chapter 4.5.1.1	See chapter 4.5.1.2	See chapter 7.3	

4.5.1.1 Axis status

The axis status is transmitted as a word (16 bits) and in bit code.

The axis status word contains the following information:

Bit no.	Designation	Description
15	–	–
14	KF	Command error
13	–	–
12	–	–
11	XE4	Error occurred on axis 4 (only for multi-axis controllers)
10	XE3	Error occurred on axis 3 (only for multi-axis controllers)
9	XE2	Error occurred on axis 2 (only for multi-axis controllers)
8	XE1	Error occurred on axis 1
7	ACC	Selected axis accelerates
6	CONST	Selected axis moves constantly
5	BRAKE	Selected axis decelerates
4	STAND	Selected axis stopped
3	–	–
2	REF_OK	Selected axis performed a reference movement
1	INIT_OK	Selected axis has been initialized
0	READY	Selected axis executed a command

Selected axis

The data transmitted by the controller always refer to the previously selected axis.
The selected axis is the axis which last performed an axis-related command without error. It does not matter whether this command was a write command or a read command.

The ACT_AXIS command can be used in multi-axis controllers to select an axis without initiating a controller function and without affecting the READY bit.

NOTE
With controllers with one axis, axis 1 (x1) is always the selected axis. With multi-axis controllers, the selected axis may be one of the axes 1 (x1) to 4 (x4), depending on which axis was last addressed by a command.
In a linear interpolation process, ACT_AXIS defines the master axis to be the selected axis.

Sample commands		Function
`POS`	`x1`	After acknowledgement of the command, axis 1 is the selected axis.
`CLRSIG_SR`	`x2`	After acknowledgement of the command, axis 2 is the selected axis.
`WRITE_PROCES`	`0, 5, bool, 1`	The selected axis does not change since the command does not refer to an axis.
`GETVEL`	`x4, actual`	After acknowledgement of the command, axis 4 is the selected axis.
`ACT_AXIS`	`x2`	Axis 2 is the selected axis.

Command error (KF)

The controller uses the command error bit (KF) to indicate a command error to the station.

KF = 0	No command error
KF = 1	The previous command was rejected due to an error.

A command error is generated when:

– an unrecognized command was sent,

– a new command was sent before the previous one had been acknowledged,

– a command cannot be executed.

In case of a command error, an error code is transmitted with the data sent by the controller (see chapter 7).

Error occurred on an axis (XE1 to XE4)

These bits are set if a movement in progress was interrupted due to an error on an axis (limit switch, power controller failure, stop, contouring error). The cause of the interruption can be determined from the axis signals.

NOTE
One of these bits may also be set when an axis is at a standstill. One example is actuation of a released limit switch.

Movement status of the selected axis (ACC, CONST, BRAKE, STAND)

These four bits reflect the current movement status of the selected axis.

ACC	Selected axis accelerates
CONST	Selected axis moves constantly
BRAKE	Selected axis decelerates
STAND	Selected axis stopped

Controller status (REF_OK, INIT_OK)

The two bits REF_OK and INIT_OK have the following meanings:

REF_OK	Selected axis performed a reference movement
	Reference found, i.e. reference movement to a limit switch (limp, limn, ref) has been carried out successfully.
INIT_OK	Selected axis has been initialized
	The power controller is switched on and ready. Temporarily stored axis signals have been deleted and the current axis position has been set to 0.

Execution status (READY)

The READY bit indicates the execution status of a command.

READY = 0	Command processing is still in progress
READY = 1	Command processing completed

The READY bit can be used for monitoring the execution of a command by a station.

ATTENTION
The READY bit is handled separately for each axis.
The READY bit is only valid for the previously issued command. In order to be able to monitor command execution also on multi-axis controllers where positioning commands must be sent to several axes simultaneously, the ACT_AXIS command is available.
The ACT_AXIS command can be used for redefining the currently selected axis without losing the validity of the READY bit.

In most cases, the READY bit is set simultaneously with command acknowledgement.
However, with the following commands, the READY bit is not set until the command has been processed completely.

Command	**READY bit is set when**
CONT	the target position has been reached
INITDRIVE	the axis has been initalized
LINMOVE	the relative linear interpolation has been executed
LINPOS	the absolute linear interpolation has been executed
MOVE	the target position has been reached
POS	the target position has been reached
REFPOS_LIMN	the reference movement has been executed
REFPOS_LIMP	the reference movement has been executed
REFPOS_REF	the reference movement has been executed
STOP_AXIS	the axis has stopped
VEL	the set speed has been reached (in speed mode only)

GETDATA requests the data object of the previous command without affecting the READY bit. It is therefore suitable for monitoring the execution of a write command.

Example:
The end of a positioning operation is to be monitored using the command POS x1, 10000.

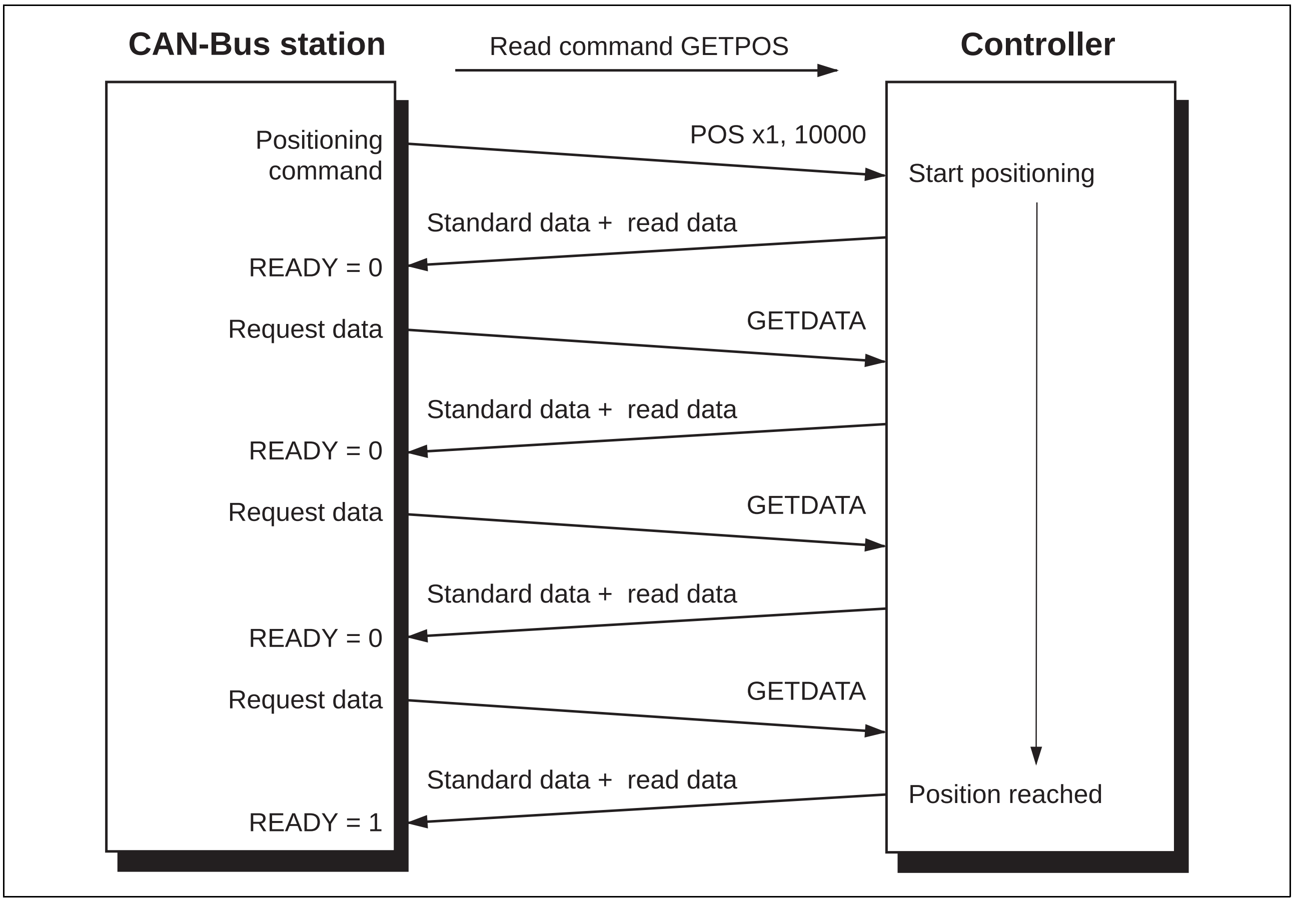


Fig. 4-3 Command monitoring with GETDATA

4.5.1.2 Axis signals

Axis signals are signals which indicate an event which occurred on an axis. Axis signals are temporarily stored in a buffer in the controller. Temporarily stored signals remain set until they are reset by the CLRSIG_SR command.

The signal states always relate to the currently selected axis (see chapter 4.5.1.1).

The axis signals are transmitted as a word (16 bits) and in bit code.

The axis signal word contains the following information:

Bit no.	Designation	Description
15	–	–
14	init_err	Initialization error on power controller
13	ref_err	Reference movement error
12	motortemp	Motor overtemperature
11	amptemp	Power controller overtemperature
10	ampnotready	Power controller not ready
9	encerr	Encoder error
8	dragerr	Contouring error
7	swstop	Software stop
6	swlimn	Negative software limit switch
5	swlimp	Positive software limit switch
4	trig	Hardware trigger input
3	stop	Hardware STOP input
2	ref	Reference switch input
1	limn	Negative hardware limit switch
0	limp	Positive hardware limit switch

NOTE
The axis signals can also be read with the GETSIG_SR command.

Example:
When the positive limit switch has been actuated briefly, bit 0 is set to 1. The axis signal word then has the following value: 0001_h. The bit remains set until the axis error condition has been eliminated and until the CLRSIG_SR command is used for resetting it.

4.5.2 Read data

The read data are selected by a station using a read command. The selected read data are transmitted to the station together with the standard data. The data are valid if KF = 0 and READY = TRUE.
Read data are, for example, speed, axis position, flags and input/output signals.

NOTE
With the GETDATA command, the previously requested read data can be read without affecting the READY bit.

Sample command	**Function**
`GETPOS       x1, actual`	The read command GETPOS x1, actual, can be used for interrogating the current position of axis 1. Four bytes in the data structure are reserved for the position value, which is interpreted as a DINT value. The position value in the example is equivalent to 5000 user-defined units.

The controller sends the following data structure to the station:

Axis status (WORD)	Axis signals (WORD)	Position value (DINT)
0013_h	0000_h	5000 (0000 1388_h)

See chapter 4.5.1.1   See chapter 4.5.1.2

NOTE
The meaning and the data type of read data differ from one read command to the next.
The read data associated with each read command are described in chapter 9.

4.6 Acknowledgement

In order to be able to communicate with a BERGER LAHR controller in a CAN-Bus network, a station must respect the following acknowledgement convention (see figure 4-4).

Two bits are relevant for acknowledgement:

KF	Command error bit (in axis status word)
READY	Execution bit (in axis status word)

The following table shows the interaction of the 2 bits which are relevant for the acknowledgement process.

KF	READY	Command
1	x	Command error
0	0	Recognized but not yet executed
0	1	Recognized and executed

The controller acknowledges a command by sending the requested data. Commands are marked valid by the station by means of the command error bit.

KF = 1 Error
KF = 0 Command o.k.

When a command has been recognized and executed (KF = 0, READY = 1), the read data sent to the station are valid and can be evaluated by the latter.

NOTE
For a more detailed description of the axis status, refer to chapter 4.5.1.1.

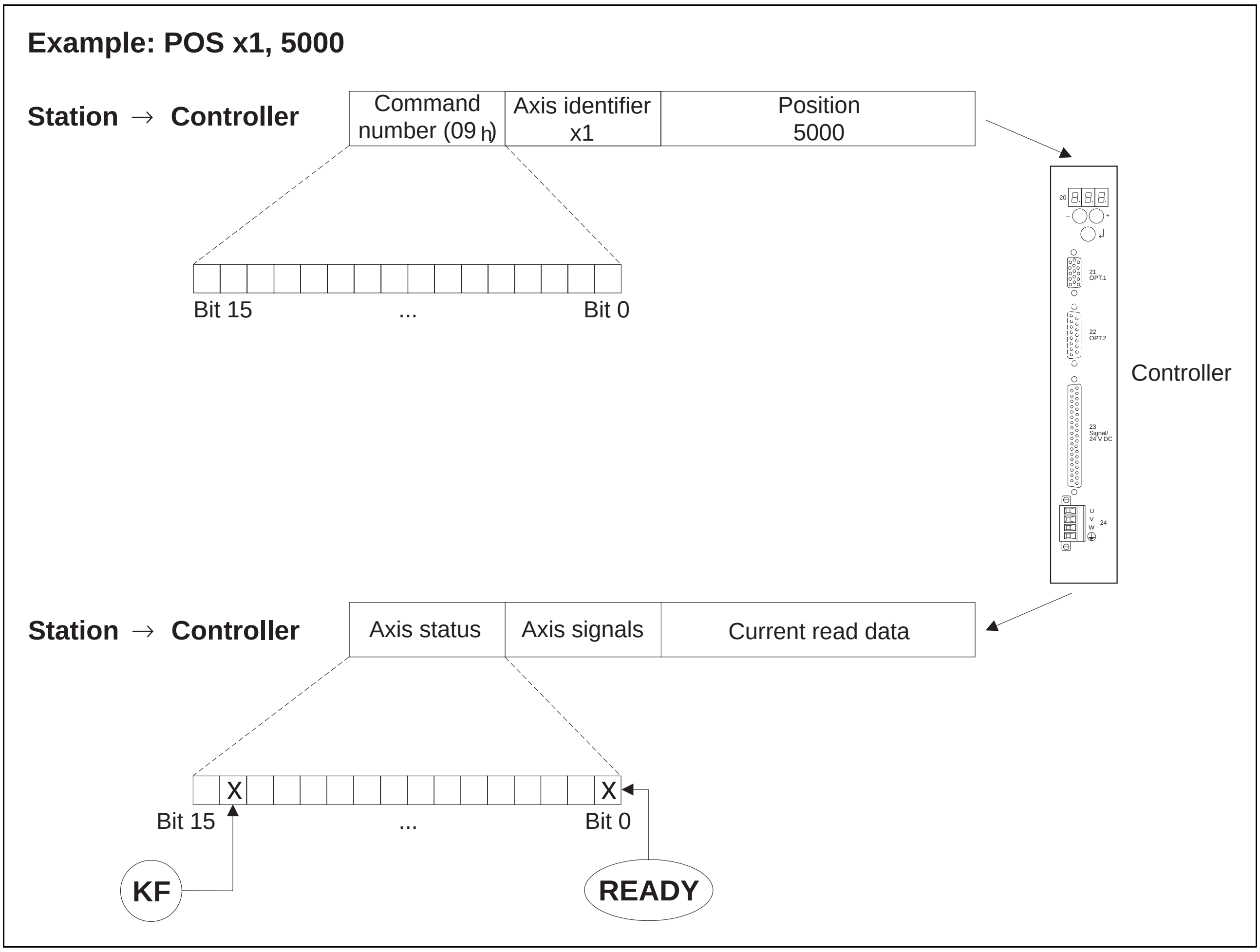


Fig. 4-4 CAN-Bus acknowledgement

Duplicate object

In systems which are exposed to a considerable amount of interference also the bus cable may be subject to interference. A common problem with the CAN-Bus is the fact that in case of such bus errors a full data packet is repeated although the recipient has already correctly received it. With BERGER LAHR controllers, this would cause one command to be executed twice.

Example:
A relative positioning operation of 1000 steps (MOVE x1, 1000) would eventually result in a 2000-step movement.

To solve this problem, the following conventions have been made:

If the same command (i.e. the same data) is to be transmitted twice, one following the other immediately, bit 15 must be set or reset for the (immediately following) second command to be recognized and executed. This is not applicable to the GETDATA command.

5 Functionality of controllers with CAN-Bus capability

5.1 Contents of this chapter

Controller functions

This chapter describes the controller functions which can be addressed through the CAN-Bus interface of a BERGER LAHR controller.
It explains the concepts and relationships required for understanding and using the command descriptions in chapters 8 and 9.

This description is applicable to all BERGER LAHR controllers with CAN-Bus capability.

NOTE
The various controllers differ with respect to their functional scope. The functional scope of each controller is described in the corresponding controller manual. The valid commands for each controller can be seen from the command summaries in chapters 8 and 9.

Examples

The sample commands in this chapter should not be regarded as complete programming examples. They are given in order to present individual commands related to each topic.

NOTE
Programming examples for all three axis operating modes are contained in chapter 6.5.

NOTE
Commands and data are usually transmitted in code. A readable format is used here for the commands to facilitate understanding.

5.2 Axis default settings

5.2.1 Preparing an axis

Initialization and hardware settings

Axis initialization

In order to position an axis, the power controller for the axis must be initialized with the INITDRIVE command. The axis is then ready for moving. The defaults for movement and axis parameters are valid (see controller manual).

Hardware settings

The SETHARDWARE command can be used for changing several hardware settings for an axis.

The following hardware settings can be set:

– Activate pulse output to power controller
– Deactivate pulse output to power controller
– Power controller enable
– Power controller disable
– Invert motor sense of rotation
– Set sense of rotation to default

Setting the motor current

Current setting

The motor current can be set for different movement states of a motor. The required current setting depends on the motor used and the load driven.

Motor currents can be set for:

– Axis at standstill
– Acceleration phase of an axis
– Constant movement of an axis

The current is set in two ways:

– On the controller front panel, select the maximum current setting for the specific motor.
– The SETCURRENT command can be used for setting the currents for various movement states of an axis individually. The current value is set as a percentage, with the percentage setting referring to the maximum current set on the power controller.
  Maximum current means:
  – for the WDP3-014 = 2.5 A
  – for the WDP3-018 = 6.8 A
  – for Series 300 controllers = The current set on the front panel rotary switch; see controller manual.

NOTE
Hardware and current settings can only be made when the axis is at a standstill.
The current settings should be made before initializing the power controller (INITDRIVE).

Command	Function
INITDRIVE	Initialize an axis
SETHARDWARE	Set hardware settings
SETCURRENT	Set motor current
GETCURRENT	Read electrical current values

NOTE
For the hardware and current setting defaults, refer to the controller manual.

Sample commands	Function
`SETCURRENT   x1, 50, stand`	Set current to 50% of maximum current for standstill
`SETCURRENT   x1, 90, accel`	Set current to 90% of maximum current for acceleration
`SETCURRENT   x1, 75, constant`	Set current to 75% of maximum current for constant movement
`INITDRIVE    x1, ampinit1`	Initialize axis 1
`SETHARDWARE  x1, dirinvert`	Invert the motor's sense of rotation

5.2.2 Axis operating modes

Each axis of a controller can move in one of three operating modes:

The following operating modes can be set:

- Point-to-point mode
- Speed mode
- Position following mode (e.g. for electronic gear)

The SETMODE command is used for setting the operating modes of an axis. The GETMODE command is used for determining the current operating mode of an axis.

In order to move an axis, the INITDRIVE command must have been used for axis initialization.

NOTE
The operating mode can only be changed when the axis is at a standstill.

Command	Function
SETMODE	Set operating mode
GETMODE	Read operating mode

NOTE
Point-to-point mode is set by default (see controller manual).

Sample commands		**Function**
`INITDRIVE`	`x1, ampinit1`	Initialize axis 1
`SETMODE`	`x1, ptp`	Set point-to-point mode for axis 1
`SETMODE`	`x1, velocity`	Set speed mode for axis 1
`SETMODE`	`x1, pos_drag`	Set position following mode for axis 1
`GETMODE`	`x1`	Determine operating mode of axis 1

NOTE
The axis identifiers x1 to x4 are used in a command for selecting the axis in a controller. The axis identifiers x2, x3, x4 are only valid for multi-axis controllers (e.g. WPM-311).

5.2.2.1 Point-to-point mode

In point-to-point mode, a positioning command is used for moving from point A to point B. Positioning of an axis can be effected with absolute values (relative to the zero point of the axis) or with incremental values (relative to the current position of the axis); see fig. 5-1.

Relative and absolute positioning

The MOVE command is used for relative (incremental) positioning, the POS command for absolute positioning.
The commands take the setpoints as parameters.
An axis accelerates or decelerates at the currently set acceleration ramp.

NOTE
Absolute positioning in point-to-point mode is only possible if either the SETPOS or the REFPOS_... command (reference movement) was used for defining a reference point (zero point) for the system of dimensions.

The STOP_AXIS command can be used for stopping a positioning operation. The CONT command can be used for resuming an interrupted axis movement.

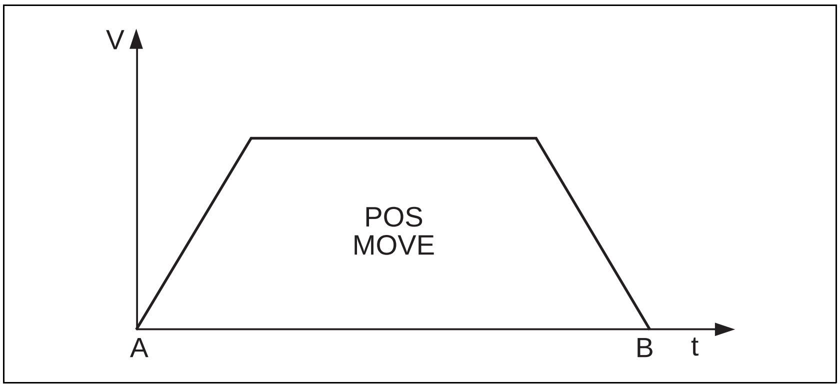


Fig. 5-1 Point-to-point mode

System limits

During an axis movement, the maximum system speed set with SETVEL_SYS and the maximum acceleration set with RAMP_... must not be exceeded.

Commands for point-to-point mode presettings

Command	Function
RAMP_EXP	Set exponential ramp
RAMP_LIN	Set linear ramp
RAMP_SIN	Set sine square ramp
SETNORM_POS_DEN	Normalizing factor denominator for positions
SETNORM_POS_NUM	Normalizing factor numerator for positions
SETNORM_VEL_DEN	Normalizing factor denominator for speeds
SETNORM_VEL_NUM	Normalizing factor numerator for speeds
SETPOS	Set current position
SETVEL_START	Set start/stop speed
SETVEL_SYS	Set maximum system speed
VEL	Set the set speed

Commands for point-to-point mode positioning operations

Command	Function
CONT	Continue interrupted axis movement
MOVE	Relative axis positioning
POS	Absolute axis positioning
STOP_AXIS	Stop axis movement

Sample commands		**Function**
`INITDRIVE`	`x1, ampinit1`	Initialize axis 1
`SETMODE`	`x1, ptp`	Set point-to-point mode for axis 1
`SETVEL_START`	`x1, 100`	Set start/stop speed to 100 steps/s
`SETVEL_SYS`	`x1, 10000`	Set maximum system speed to 10000 steps/s
`SETPOS`	`x1, 0`	Set current position to 0 (set dimensions)
`VEL`	`x1, 1000`	Set speed of axis 1 is 1000 steps/s
`POS`	`x1, 3000`	Absolute positioning of axis 1 to position 3000
`MOVE`	`x1, 2000`	Relative movement of axis 1 by 2000 user-defined units

5.2.2.2 Speed mode

In speed mode, the VEL command is used for defining a set speed and starting a movement. The axis continues to move at this speed until a different set speed is defined.
An axis accelerates or decelerates at the currently set acceleration ramp (fig. 5-2).

VEL = 0 stops the axis movement. VEL ≠ 0 can be used to resume the movement. When a negative speed value is specified, the sense of rotation of the axis is inverted.

The STOP_AXIS command can be used for stopping an axis movement. The CONT command can be used for resuming an interrupted axis movement.

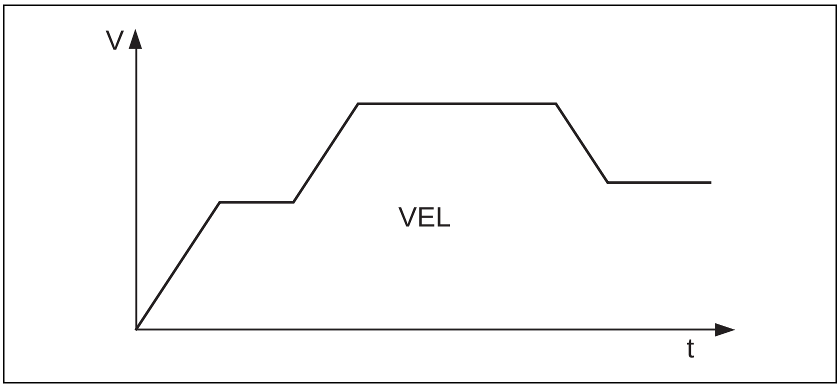


Fig. 5-2 Speed mode

System limits

During an axis movement, the maximum system speed set with SETVEL_SYS and the maximum acceleration set with RAMP_... must not be exceeded.

Commands for speed mode presettings

Command	Function
RAMP_EXP	Set exponential ramp
RAMP_LIN	Set linear ramp
RAMP_SIN	Set sine square ramp
SETNORM_VEL_DEN	Normalizing factor denominator for speeds
SETNORM_VEL_NUM	Normalizing factor numerator for speeds
SETVEL_START	Set start/stop speed
SETVEL_SYS	Set maximum system speed

Commands for speed mode positioning operations

Command	Function
CONT	Continue interrupted axis movement
STOP_AXIS	Stop axis movement
VEL	Set the set speed

Sample commands		Function
`INITDRIVE`	`x1, ampinit1`	Initialize axis 1
`SETMODE`	`x1, velocity`	Set speed mode for axis 1
`VEL`	`x1, 1000`	Axis 1 moves at set speed 1000 steps/s
`VEL`	`x1, 2000`	Axis 1 moves at set speed 2000 steps/s
`STOP_AXIS`	`x1`	Axis 1 is stopped
`CONT`	`x1`	Axis 1 movement is resumed

5.2.2.3 Position following mode (electronic gear)

This operating mode can be used for implementing an electronic gear (fig. 5-3).
In this mode, externally supplied pulses are counted, multiplied with an adjustable gear ratio and used as the reference variable for the position of a motor. The motor follows the supplied reference variable exactly.

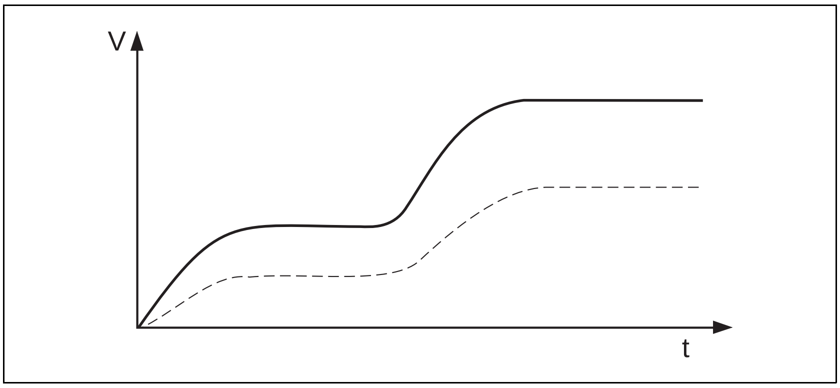


Fig. 5-3 Position following mode

External pulses

External pulses may be A/B signals or pulse/direction signals from the encoder input (fig. 5-4). The signal type is set with the SETENCODER command . The SETMODE command is used for selecting an encoder port of the controller and assigning it to an axis. On controllers with internal power controller, the encoder connection is selected with the p1 or p2 parameter. WDP3-014/018 controllers can only be equipped with the p2 encoder connection.

Gear ratio

The gear ratio is set with the SETNORM_GEAR_DEN and SET-NORM_GEAR_NUM commands.
The following applies:

Drive units = Reference variable x Gear ratio

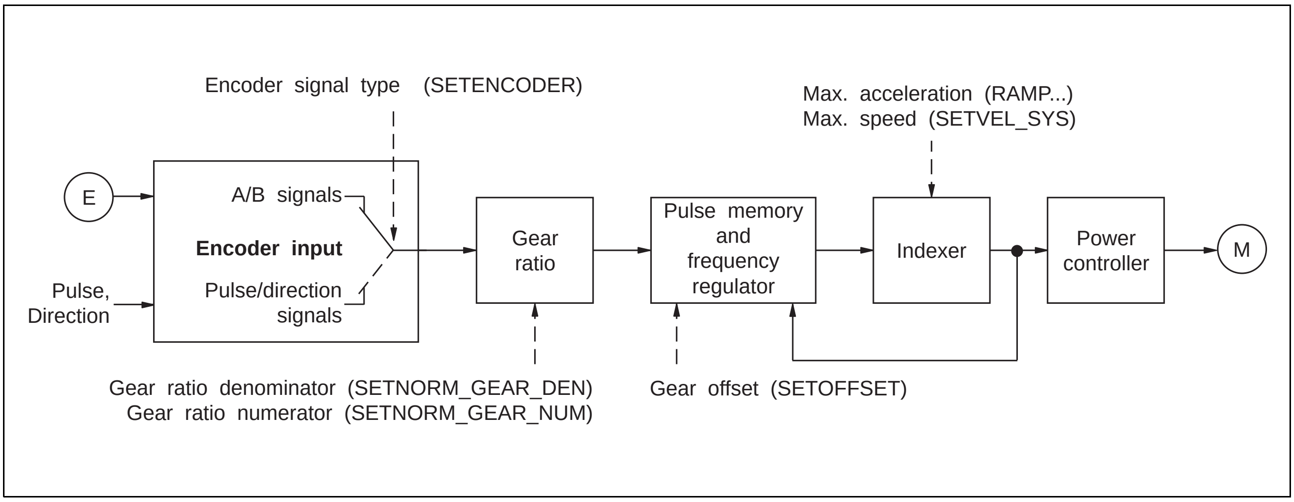


Fig. 5-4 Block diagram of the electronic gear

The axis moves when pulses are supplied at the encoder input and the gear ratio is ≠ 0. When the gear ratio is = 0, the axis is at a standstill.

The pulse frequency at the encoder input determines the acceleration and the speed of the axis. If no pulses are present, this is equivalent to axis standstill.
Normally the motor follows the supplied pulses exactly, i.e. position, speed and acceleration correspond to the reference variable multiplied with the gear ratio.

Example:
With the gear ratio set to 10, the following applies:

	Reference variable	Motor moves
Position	100 pulses	1000 steps
Speed	100 kHz	1000 kHz
Acceleration	10 kHz/s	100 kHz/s

Speed

If the frequency (speed) of the reference variable, multiplied with the gear ratio, is greater than the system speed set with the SETVEL_SYS command, the motor moves at this system speed.

Acceleration

If the acceleration of the reference variable, multiplied with the gear ratio, is greater than the acceleration set with the RAMP... command, the motor accelerates at this maximum acceleration.

Offset

The SETOFFSET command can be used for specifying an offset for the reference variable. An offset is a relative position which is added to the reference variable. Changing the offset accelerates or decelerates the axis. When the offset has been processed, the axis continues to run normally.
The following applies (only when changing the offset):

Drive units = Offset + (Reference variable x Gear ratio)

Movement range

In electronic gear mode, the absolute movement range is not limited, i.e. the motor can turn in one direction for an unlimited period.

The STOP_AXIS command can be used for stopping an axis movement. The CONT command can be used for resuming an interrupted axis movement.

Commands for position following mode presettings

Command	Function
RAMP_EXP	Set exponential ramp
RAMP_LIN	Set linear ramp
RAMP_SIN	Set sine square ramp
SETENCODER	Set signal type of encoder
SETMODE	Set operating mode and encoder source
SETNORM_GEAR_DEN	Set gear ratio denominator
SETNORM_GEAR_NUM	Set gear ratio numerator
SETOFFSET	Set reference variable offset
SETVEL_START	Set start/stop speed
SETVEL_SYS	Set maximum system speed

NOTE
By default, gear ratio numerator = 0 and denominator = 1 are set. This means that the axis is standing still when changing over to position following mode (see controller manual).

Sample commands		Function
`INITDRIVE`	`x1, ampinit1`	Initialize axis 1
`SETENCODER`	`p2, encpulsdir`	Encoder 1 signal type: Process pulse/ direction signals
`SETMODE`	`x1, pos_drag, p2`	Set position following mode for axis 1
`SETNORM_GEAR_DEN`	`x1, 1`	Set gear ratio 50, denominator = 1
`SETNORM_GEAR_NUM`	`x1, 50`	Set gear ratio 50, numerator = 50
`SETOFFSET`	`x1`	Reference variable offset for axis 1 is 10

NOTE
To specify a gear ratio, a numerator and a denominator must always be defined. The denominator of the gear ratio must be set prior to the numerator. The denominator is not accepted until the numerator is passed.

NOTE
The presettings (encoder assignment and signal type) for position following mode should be made in point-to-point mode prior to changing over to position following mode. Offset and gear ratio may then be changed during operation.

5.3 Normalizing factors

Normalizing factors are used for

- converting user-defined units for positions and speeds into drive units, and
- step-down or step-up gearing with the reference variable in position following mode (gear ratio for electronic gear).

Using normalizing factors allows you to specify positions in common units of measurement (cm, m/s, etc.) rather than in controller-specific drive units (e.g. motor steps).

The commands SETNORM_POS_DEN and SETNORM_POS_NUM or SETNORM_VEL_DEN and SETNORM_VEL_NUM are used for setting normalizing factors for position or speed values.

The SETNORM_GEAR_DEN and SETNORM_GEAR_NUM commands are used for setting the gear ratio for an electronic gear (see chapter 5.2.2.3).

NOTE
To specify a normalizing factor, a numerator and a denominator must always be defined. The denominator must be set prior to the numerator. The denominator is not accepted until the numerator is passed.

Command	**Function**
SETNORM_GEAR_DEN	Set gear ratio denominator
SETNORM_GEAR_NUM	Set gear ratio numerator
SETNORM_POS_DEN	Normalizing factor denominator for positions
SETNORM_POS_NUM	Normalizing factor numerator for positions
SETNORM_VEL_DEN	Normalizing factor denominator for speeds
SETNORM_VEL_NUM	Normalizing factor numerator for speeds

NOTE
The following defaults are used (see controller manual):
- *Normalizing factor for position values: 1 : 1*
- *Normalizing factor for speed values: 256 : 1*
- *Normalizing factor for gear ratios: 0 : 1 (axis stopped)*

5.3.1 Drive units

Drive units are processing parameters internal to the controller, which are used for positions, speed and acceleration values. Values specified in user-defined units are always converted to drive units before the controller executes a command.

Drive units are defined as follows:

Drive units for positions	Motor steps
Drive units for speeds	256 x Motor steps/s
Drive units for acceleration	Motor steps/s^2

5.3.2 User-defined units

User-defined units are processing parameters which can be freely defined by the user. They are used for enabling the user to specify position, speed and acceleration values in application-related units of measurement (metre, inch, degree, hertz, etc.).
In order to be able to specify these values in the desired user-defined units, the corresponding normalizing factor must be set previously. The following relationships apply:

For position values:

$$\text{Drive units} = \text{User-defined units} \times \text{Normalizing factor}$$

[Motor steps]

For speed values:

$$\text{Drive units} = \text{User-defined units} \times \frac{\text{Normalizing factor}}{256}$$

[Motor steps/s]

NOTE
A normalizing factor of 1 : 1 in a position specification means that user-defined units are equal to drive units.
The normalizing factor of 256 : 1 for speeds allows you to specify speeds using the hertz unit (1 Hz = 1 motor step/s).

Sample commands		Function
`INITDRIVE`	`x1, ampinit1`	Initialize axis 1
`SETMODE`	`x1, ptp`	Set point-to-point mode for axis 1
`SETNORM_POS_DEN`	`x1, 10`	Set normalizing factor 1/10 for positions, denominator = 10
`SETNORM_POS_NUM`	`x1, 1`	Set normalizing factor 1/10 for positions, numerator = 1
`SETNORM_VEL_DEN`	`x1, 1`	Set normalizing factor 512 for speed, denominator = 1
`SETNORM_VEL_NUM`	`x1, 512`	Set normalizing factor 512 for speed, numerator = 512
`VEL`	`x1, 1000`	Set speed is 2000 steps/s = 1000 x 512/256 steps/s
`MOVE`	`x1, 100`	100 user-defined units are equivalent to a motor rotation of 10 steps

5.4 Acceleration

When starting or when changing the speed, a motor must be accelerated in such a way that it is not overloaded and that no loss of synchronicity occurs.

The acceleration can be set by way of a ramp.

Acceleration ramps

There are three types of acceleration ramps:

- Linear ramp
- Exponential ramp (optimum for stepping motors)
- Sine square ramp (for smooth starting and braking)

The commands RAMP_LIN, RAMP_EXP and RAMP_SIN can be used for calculating specific acceleration ramps using the appropriate ramp shape.

Maximum acceleration

The maximum acceleration value and the basic shape of the ramp are used for calculating an acceleration curve for a specific axis load. The maximum acceleration is specified in Hz/ms (kHz/s) when the normalizing factor for speed is set to 256 : 1 (default).

Based on the maximum acceleration and the system speed set with SETVEL_SYS, a special acceleration curve is calculated. The acceleration curve is generated up to the maximum system speed (fig. 5-5).

When braking, the calculated curve is applied inversely.

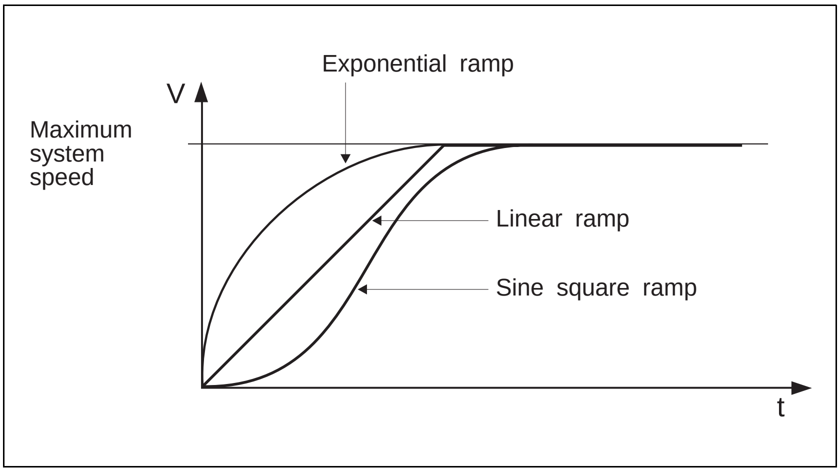


Fig. 5-5 Acceleration ramps

NOTE
Acceleration curves can only be changed in point-to-point mode when the axis is at a standstill.

Command	Function
RAMP_EXP	Set exponential ramp
RAMP_LIN	Set linear ramp
RAMP_SIN	Set sine square ramp
SETVEL_SYS	Set maximum system speed

NOTE
A linear ramp is set by default (see controller manual).

Sample commands		Function
`INITDRIVE`	`x1, ampinit1`	Initialize axis 1
`SETMODE`	`x1, velocity`	Set speed mode for axis 1
`SETVEL_SYS`	`x1, 10000`	Set maximum system speed to 10000 steps/s
`RAMP_EXP`	`x1, 500`	Axis 1 with exponential ramp and the maximum acceleration of 500 Hz/ms, at speed normalizing factor 256 : 1
`VEL`	`x1, 8000`	Axis 1 moves at set speed 8000 steps/s

ATTENTION
The acceleration ramp is calculated as a function of the maximum system speed and the maximum acceleration.
The actual movement curve is the section of the calculated curve between start/stop speed and set speed.
In the case of non-linear ramps (fig. 5-6) note that the ramps are only optimally utilized if
- ***the set speed is close to the maximum system speed;***
- ***the acceleration curve actually reaches the set speed during a positioning operation.***

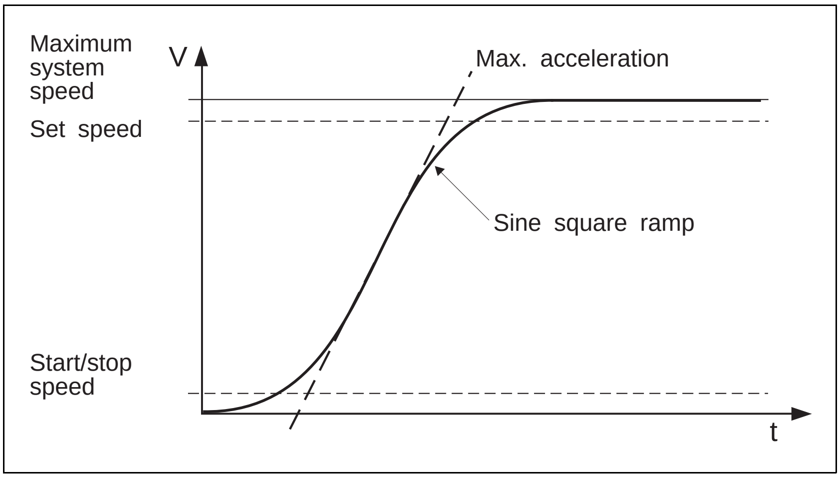


Fig. 5-6 Non-linear ramps

5.5 Speeds

5.5.1 Speed limit values

The SETVEL_START and SETVEL_SYS commands can be used for setting the start/stop speed and the maximum system speed.

Start/stop speed

The start/stop speed is the speed at which the motor starts from standstill or stops. The maximum start/stop speed which can be set for an axis depends on the load inertia (see motor characteristic).

Maximum system speed

The maximum system speed sets a limit value for the maximum permissible speed of an axis movement.

5.5.2 Set speed

The VEL command is used for programming the set speed for positioning operations in point-to-point mode.

In speed mode, VEL also initiates an axis movement in addition to setting the set speed.

In position following mode, the set speed is defined by the pulse frequency of the encoder source.

The GETVEL command can be used for determining the current speed of an axis in all of the three operating modes.

Command	Function
GETVEL	Read speed value
VEL	Set the set speed
SETVEL_START	Set start/stop speed
SETVEL_SYS	Set system speed

NOTE
For the system, start/stop, and set speed defaults, refer to the controller manual.

Sample commands		Function
`SETVEL_START`	`x1, 100`	Start/stop speed 100 Hz for axis 1, if normalizing factor = 256
`SETVEL_SYS`	`x1, 10000`	Maximum system speed 10000 Hz for axis 1, if normalizing factor = 256
`VEL`	`x1, 1000`	Set speed 1000 Hz for axis 1, if normalizing factor = 256

5.6 Reference movement

In a reference movement, a reference point is approached which is the reference point (zero point) for the system of dimensions. All subsequent absolute positioning operations exclusively refer to this zero point.

NOTE
Reference movements are only possible in point-to-point mode.

Reference movements can be executed towards the

- negative limit switch,
- positive limit switch, and
- reference switch.

Figures 5-7 and 5-8 illustrate the principles of the different reference movements.

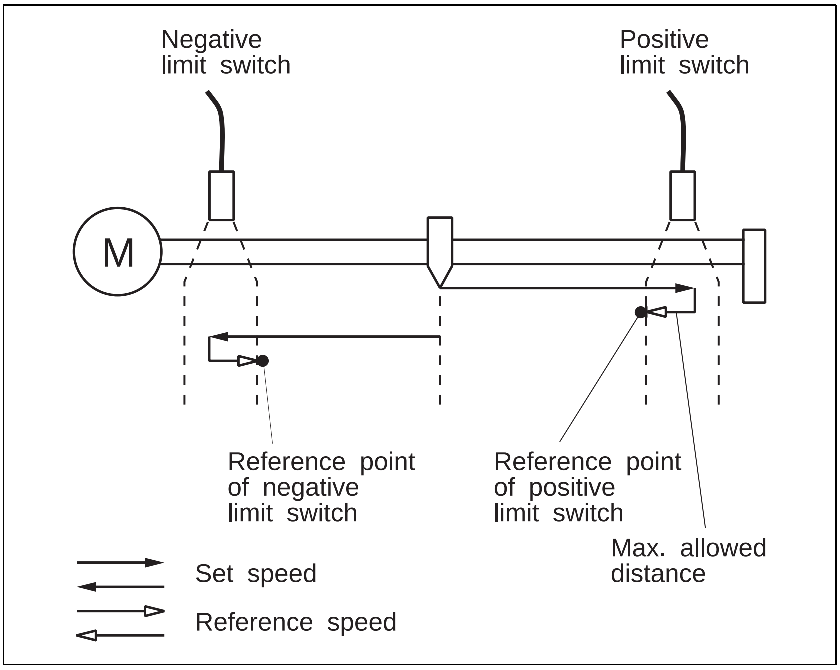


Fig. 5-7 Principle of reference movement to limit switch

The reference movement is executed at the set speed (VEL command). The reference speed passed with the REFPOS_... command is the speed at which the axis moves away from a limit or reference switch.

In a reference movement, the maximum allowed distance from the limit switch (reference switch) is monitored. The axis must have left the limit switch (reference switch) within this distance, otherwise the reference movement would be aborted (see REF_OUT_DISTANCE command).

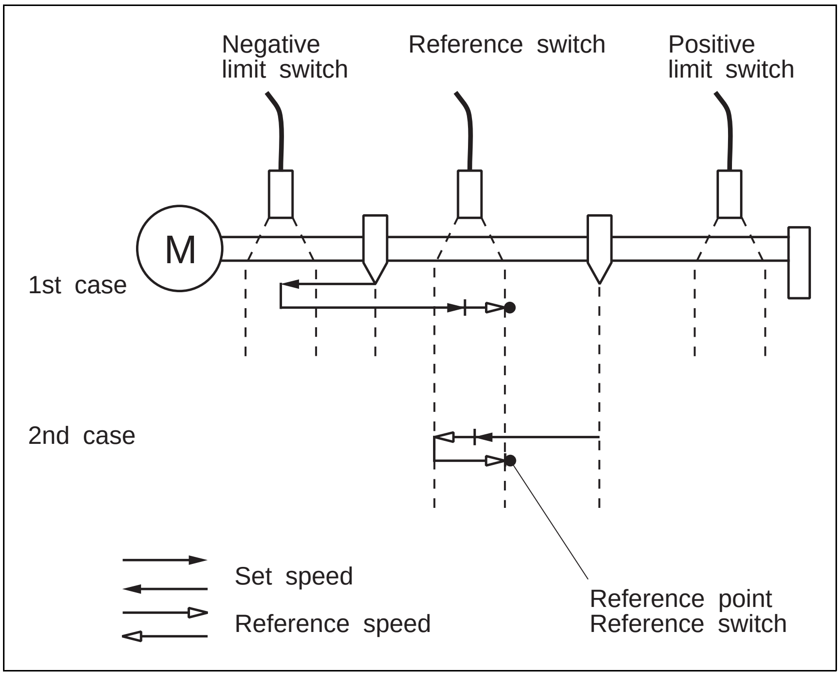


Fig. 5-8 Principle of reference movement to reference switch

NOTE
The limit switch/reference switch clearing speed should be equal to or less than the start/stop speed in order to ensure accurate stopping of the drive at the correct point.

Command	**Function**
REFPOS_LIMN	Reference movement towards the negative limit switch
REFPOS_LIMP	Reference movement towards the positive limit switch
REFPOS_REF	Reference movement towards the reference switch

NOTE
Instead of performing a reference movement it is also possible to set a reference point for the system of dimensions using the SETPOS command.

Sample commands		**Function**
`SETMODE`	`x1, ptp`	Set point-to-point mode for axis 1
`VEL`	`x1, 1000`	Set speed 1000 Hz for axis 1
`REFPOS_REF`	`x1, 200`	Reference movement of axis 1 towards the reference switch. The axis moves at set speed 1000 Hz in positive direction to the reference switch and at speed 200 Hz away from the reference switch.
`REFPOS_REF`	`x1, -200`	Reference movement of axis 1 towards the reference switch in negative direction

5.7 Rotation monitoring

Rotation monitoring

Rotation monitoring is used for detecting and avoiding positional deviations of motor movements. Positional deviations can occur when an obstacle or a load causes a loss of synchronicity of the motor and the motor cannot reach the preset position.

With rotation monitoring, the actual position is detected by an encoder and then compared with the setpoint. If the difference between set and actual position (following error) exceeds a predefined value (following error limit), a contouring error is reported and the motor brakes. A contouring error is registered as an axis error in the axis status word (see chapter 4.5.1.1). The type of the axis error can be determined from the axis signal word (see chapter 4.5.1.2). When a contouring error occurs, the dragerr bit in the axis signal word is set.

NOTE
The following error limit is a permanent setting of 9 increments at an encoder resolution of 500 marks and 18 increments at an encoder resolution of 1000 marks.
On controllers with an external power controller, the following error limit set on the power controller is valid.

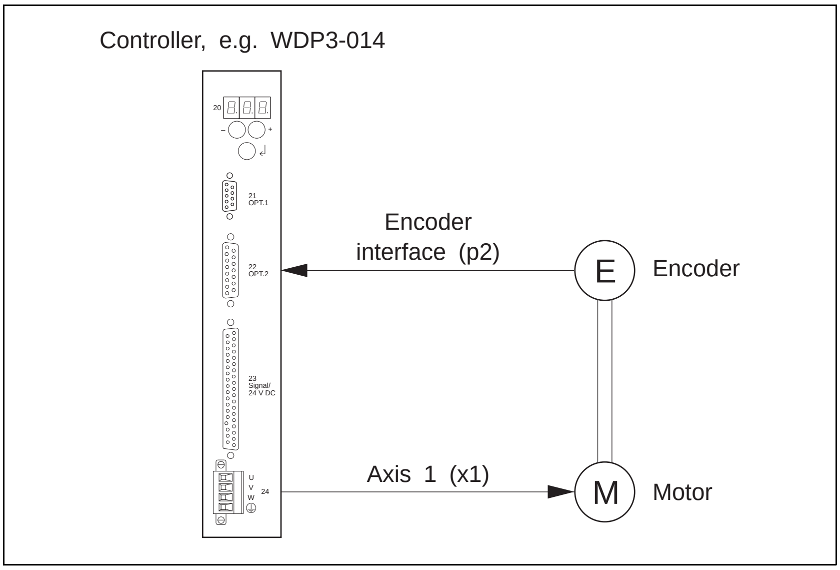


Fig. 5-9 Rotation monitoring principle

Commands for rotation monitoring

Command	Function
ROTMON_DISABLE	Disable rotation monitoring
ROTMON_ENABLE	Enable rotation monitoring
ROTMON_RESET	Reset rotation monitoring

The ROTMON_ENABLE command is used for initializing and activating rotation monitoring for an axis on a freely selectable encoder interface. The encoder interface is selected with the p1 or p2 parameter on controllers with internal power controller and with the pext parameter on controllers with external power controller. WDP3-014/018 controllers can only be equipped with the p2 encoder connection. At the same time, this command can be used for setting the resolution of the encoder (no. of encoder marks).

The ROTMON_RESET command is used for resetting rotation monitoring when an encoder error occurred. The active encoder error (dragerr) is cleared and the current encoder position is used as the current axis position. In this way, the position is defined, and the axis can move normally after the ROTMON_RESET command.

NOTE
An encoder error can also be reset using the CLRSIG_SR command. However, this command only resets the error bit. It does not handle the encoder position as with the ROTMON_RESET command.

The ROTMON_DISABLE command is used for deactivating rotation monitoring for an axis on an encoder interface. In this case, rotation monitoring is not performed any longer.

NOTE
It is preferable to use the encoder interface 2 (p2) for rotation monitoring since this encoder interface continues to record the encoder position in case of a power controller failure (i.e. encoder error). This allows using the ROTMON_RESET and CONT commands in order to reach the intended position even though the power controller may have failed.

Sample command		Function
`ROTMON_ENABLE`	`x1, p2, 1000`	Activate rotation monitoring for axis 1 on encoder interface 2 at an encoder resolution of 1000 marks
`ROTMON_RESET`	`x1, p2`	Reset rotation monitoring for axis 1 on encoder interface 2
`ROTMON_DISABLE`	`x1, p2`	Deactivate rotation monitoring for axis 1 on encoder interface 2

5.8 Controlling a brake

The BRAKE command can be used for addressing any output Qx for controlling a brake. Figure 5-10 shows the relationship between the ENABLE (power controller enable) and READY (power controller ready) signals and the output signal for the brake.

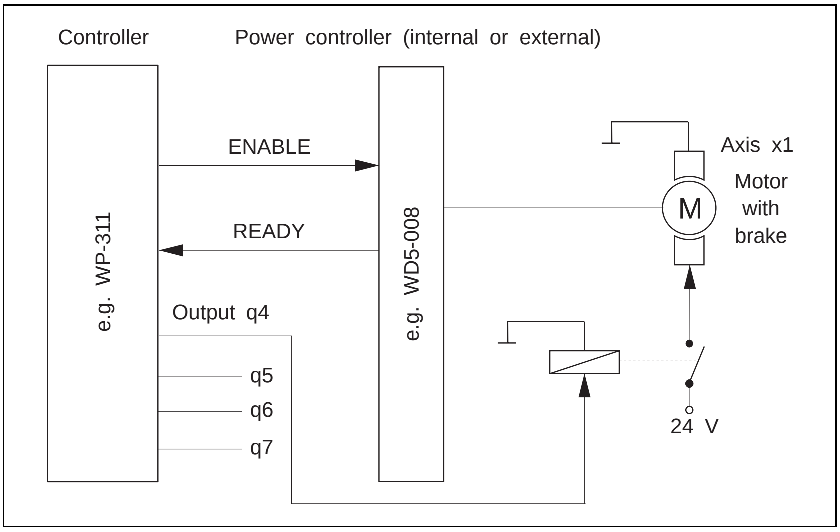


Fig. 5-10 Signals for the brake function

The brake (fig. 5-11) opens (Qx = high) when the power controller has been enabled (INITDRIVE command) and the power controller is ready. The brake is applied (Qx = low) when the power controller is no longer ready (READY = low).

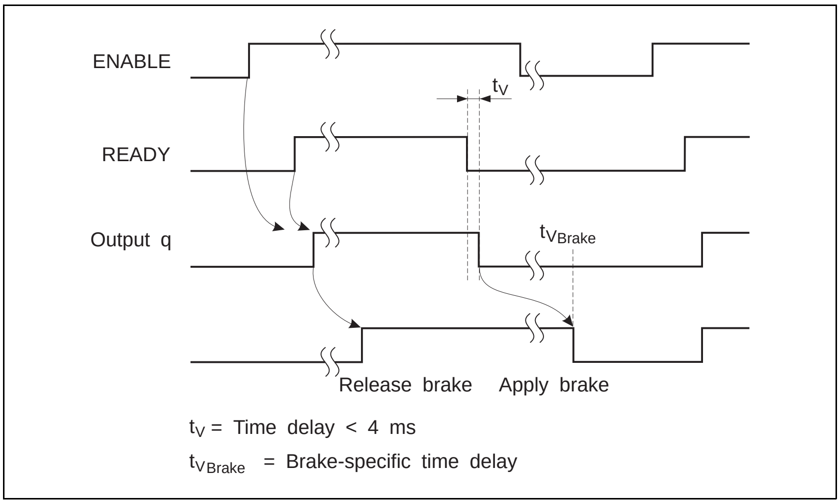


Fig. 5-11 Timing diagram for the brake function

Command	Function
BRAKE	Define output for brake

Sample command

`BRAKE          x1, 0, 5`

Function

Output 5 is used for the brake function

5.9 Axis states and axis signals

5.9.1 Axis states

An axis can have the following movement states:

Acceleration

Constant movement

Braking (deceleration)

Standstill

This information is transmitted in the standard data (axis status word) to the station (chapter 4.5.1.1).

Furthermore, the following axis states can be determined for error analysis using the GETSTATE command:

- Reference movement error caused by hardware STOP
- Reference movement error caused by reference switch
- Reference movement error caused by negative limit switch
- Reference movement error caused by positive limit switch
- Position overrun
- Acceleration not defined
- Actual position not defined
- Power controller not enabled
- Reference movement error due to axis blocked
- Reference movement error due to limit switch not enabled
- General reference movement error
- Reference movement active
- Reference movement o.k.

Command	Function
GETSTATE	Read error status of an axis

Sample command		Function
`GETSTATE`	`x1`	Read error status of axis 1

5.9.2 Axis signals

Several signals internal or external to the controller can be monitored on an axis. These signals report unexpectedly (asynchronously) occurring events.
When an event occurs on an axis, the corresponding signal is temporarily stored in a buffer on the controller. This allows detecting the event even when the signal is no longer present.

NOTE
Only those signals enabled with the ENSIG command respond to events. The limp and limn signals are enabled by default.

The following axis signals are available:

Axis signal	**Description**
ampnotready	Power controller not ready
amptemp	Power controller overtemperature
dragerr	Contouring error
encerr	Encoder error
limn	Negative limit switch
limp	Positive limit switch
motortemp	Motor overtemperature
ref	Reference switch
swlimn	Negative software limit switch
swlimp	Positive software limit switch
swstop	Software stop
stop	Stop input
trig	Trigger input

Temporarily stored signals

Temporarily stored signals can be read with the GETSIG_SR command. The GETSIG command is used for reading the current signal states directly (energized = 1, deenergized = 0). A temporarily stored signal is kept until it is cleared with the CLRSIG_SR command.

The SETSIG_ACTIV_H command can be used for setting the active state (active if energized = 1, active if deenergized = 0) for the limn, limp, ref and trig signals. The currently set active state can be read with the GETSIG_ACTIV_H command .

NOTE
Some signals cause an interruption of the axis movement. An interrupted axis movement can only be resumed when the cause of the error has been eliminated and the corresponding temporarily stored axis signal has been cleared using CLRSIG_SR.

NOTE
The CONT command clears the temporarily stored signals automatically and resumes the axis movement.

Command	Function
CLRSIG_SR	Clear temporarily stored axis signals
ENSIG	Enable or disable axis signals
GETENSIG	Read enabled or disabled axis signals
GETSIG	Read current axis signal states
GETSIG_ACTIV_H	Read active state of axis signals
GETSIG_SR	Read temporarily stored axis signals
SETSIG_ACTIV_H	Set active state of axis signals

NOTE
For the signal active state defaults, refer to the controller manual.

Sample commands		**Function**
`ENSIG`	`x1, limp`	Enable positive limit switch signal of axis 1
`GETENSIG`	`x1`	Determine enabled axis signals of axis 1
`CLRSIG_SR`	`x1, all`	Clear all temporarily stored axis signals of axis 1

Figure 5-12 shows the effect of the individual commands on signal evaluation.

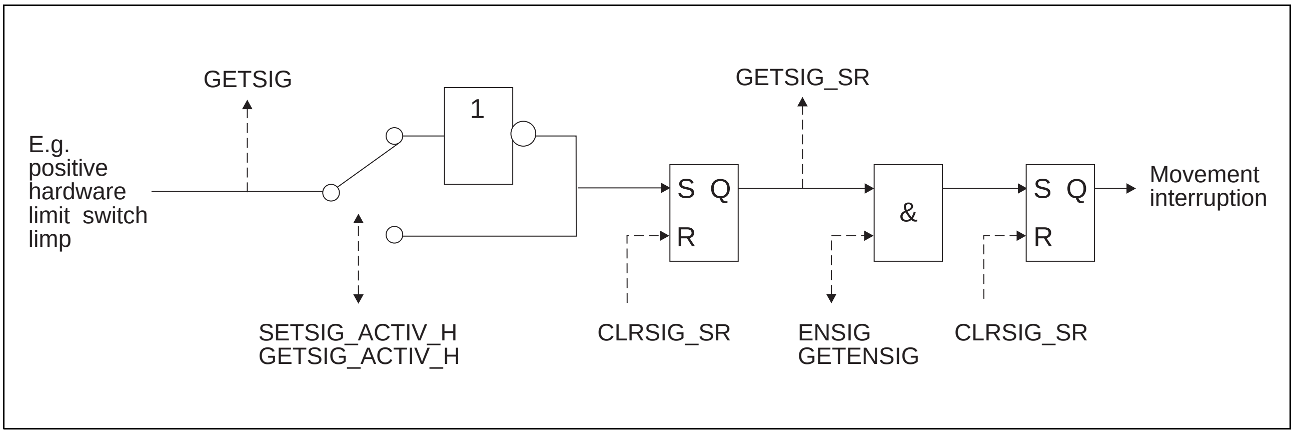


Fig. 5-12 Signal evaluation

5.10 Linear interpolation for Series 300 multi-axis positioning units (WPM)

With the Series 300 multi-axis positioning units (e.g. WPM-311), several axes can move using linear interpolation.
Linear interpolation means in this case that the axes can be activated simultaneously and move interdependently and that all axes reach their final positions at the same time.

Linear interpolation is possible with two or three axes.

NOTE
Linear interpolation can be effected with absolute values (LINPOS) or with relative values (LINMOVE).
When initiating a linear interpolation process, the axes addressed must already be referenced. Referencing, or setting dimensions, can be effected with the SETPOS command or implicitly with the INITDRIVE command or by executing a reference movement.

Commands	**Meaning**
SETIPOS	Prepare linear interpolation
LINPOS	Absolute linear interpolation
LINMOVE	Relative linear interpolation
STOP_AXIS	Stop linear interpolation
ACT_AXIS	Defines the master axis as the active one

The SETIPOS command can be used for preparing a linear interpolation, i.e. a target position is set for each axis involved in linear interpolation.

The linear interpolation process is started with the commands LINPOS or LINMOVE. The LINPOS command performs absolute linear interpolation relative to the zero point of the axes. The positions preset with SETIPOS are interpreted as absolute positions.
The LINMOVE command performs linear interpolation relative to the current position of the axes. The positions preset with SETIPOS are interpreted as relative positions.

The STOP_AXIS command can be used for stopping linear interpolation.

The ACT_AXIS command is used for defining the master axis in a linear interpolation as the active axis. The master axis is the one which travels the longest path during interpolation.

The following points must be observed for linear interpolation:

– The axes involved in linear interpolation must be set to point-to-point mode (SETMODE command).
– The speeds and accelerations of the axes involved must be set before the linear interpolation process.
– The target positions, speeds and accelerations of the axes involved cannot be changed during a linear interpolation process.
– You cannot perform several linear interpolation processes at the same time.
– An electronic gear is affected by interpolation (it may be necessary to reinitialize it after the linear interpolation process).
– When starting a linear interpolation process, the axes involved must be at a standstill and without error (see XE bit in axis status).

Monitoring a linear interpolation

The end of a linear interpolation process is monitored by the STAND bit and the XE bits in the axis status word.
If another field bus command is executed after a LINPOS or LINMOVE command, the ACT_AXIS l1 command must be used first to select the master axis as the active axis again in order to be able to check the corresponding bits.
The XE bits of the axes involved in linear interpolation indicate any errors which may have occurred during interpolation.

Principle of linear interpolation

The principle of linear interpolation is illustrated here by way of an example:

Example:

Two axes to move from position A (100, 100) to position B (600, 300) with linear interpolation. Die Linearinterpolation soll relativ zur aktuellen Position der Achsen erfolgen.

First the setpoints are passed to the two axes.

```
SETIPOS x1 500
SETIPOS x2 200
```

The linear interpolation is started with the following command.

```
LINMOVE l1
```

The linear interpolator uses the setpoints for calculating the speeds and accelerations required for the interpolation and controls the individual axes. It is ensured that the preset speeds and accelerations of the individual axes are not exceeded.

Figure 5-13 shows the principle of the linear interpolation.

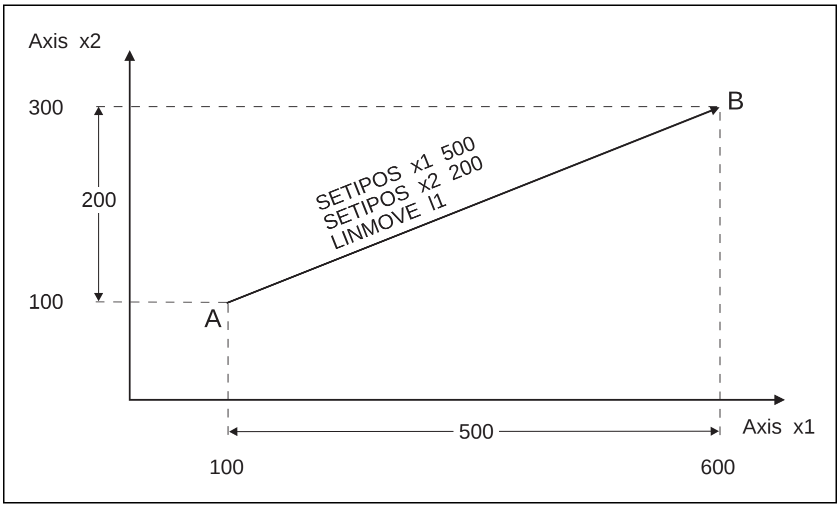


Fig. 5-13 Principle of linear interpolation

Sample commands			**Function**
1. Relative linear interpolation			
`SETIPOS`	`x1`	`6000`	Target position for axis x1
`SETIPOS`	`x2`	`7000`	Target position for axis x2
`LINMOVE`	`l1`		Start relative linear interpolation
2. Absolute linear interpolation			
`SETIPOS`	`x1`	`6000`	Target position for axis x1
`SETIPOS`	`x2`	`7000`	Target position for axis x2
`LINPOS`	`l1`		Start absolute linear interpolation

5.11 Analog inputs and outputs (Series 300 only)

Series 300 controllers may be equipped with an analog module (ANOZ).

The analog module has
- 1 analog output
- 5 analog inputs

The analog signal at the output can be set with the SETANALOG command. The voltage is specified in millivolts (mV).

The analog signals at the inputs can be read with the GETANALOG command.

The inputs and outputs are addressed by the analog module (a2) and the channel number of the input or output, respectively.

Commands	Meaning
SETANALOG	Set analog output
GETANALOG	Read analog input

Sample commands

```
SETANALOG   a2, 1, 5000
GETANALOG   a2, 1
```

5.12 Input/output signals

A controller has a fixed number of inputs and outputs. The input/output signals of a controller can be read or set directly or via the process image (indirectly).

The commands WRITE_OUTPUT and READ_INPUT are used for reading or setting input/output signals directly.

The commands WRITE_PROCESS and READ_PROCESS are used for reading or setting input/output signals indirectly via the process image.

Signals can be read or set either bit by bit or word by word.

Figure 5-14 illustrates accessing individual inputs or outputs of the controller process image.

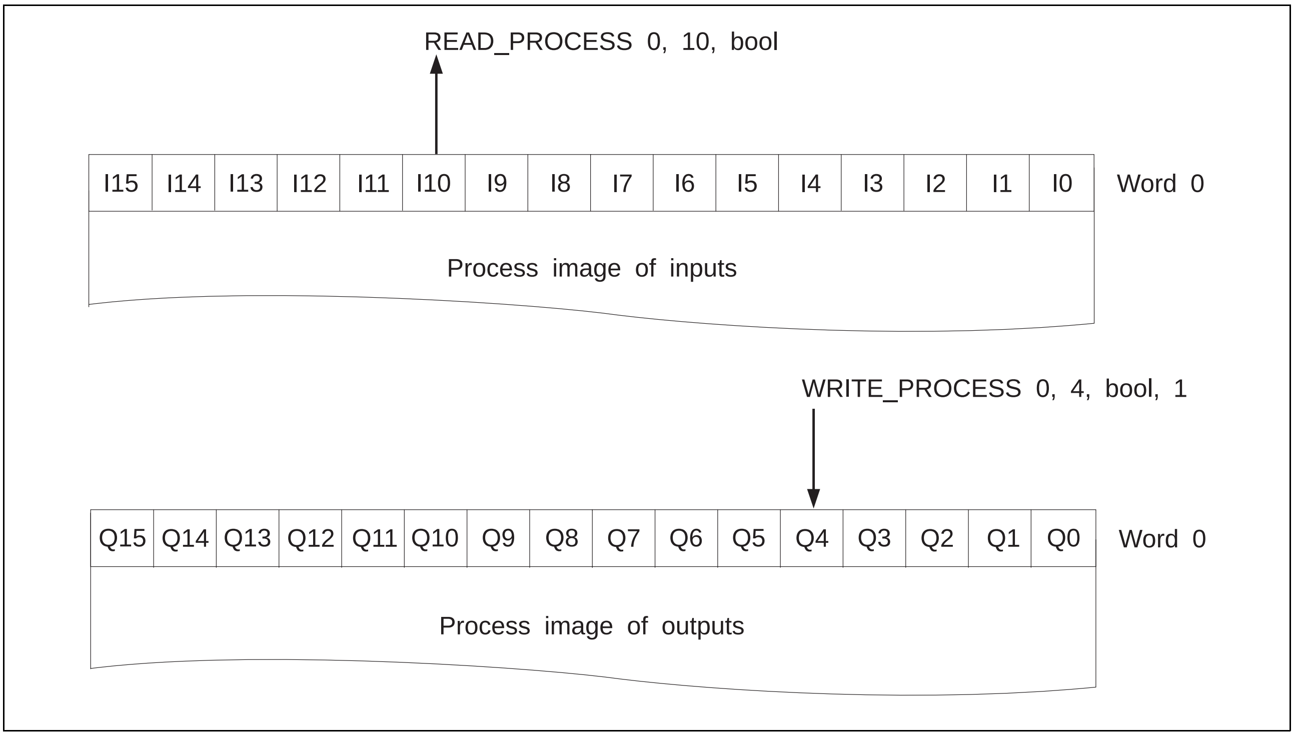


Fig. 5-14 Access to inputs and outputs

NOTE
Indirect setting and reading of input/output signals via the process image is only possible with an application program running on a Series 300 controller (see chapter 5.15).

Command	Function
READ_INPUT	Read inputs directly
READ_PROCESS	Read inputs via the process image
WRITE_OUTPUT	Set outputs directly
WRITE_PROCESS	Set outputs via the process image

Sample commands		**Function**
`WRITE_OUTPUT`	`0, 5, bool, 1`	Set output bit 5 directly
`WRITE_PROCESS`	`0, 4, bool, 1`	Set output bit 4 to 1 in the process image
`READ_INPUT`	`0, 0, word`	Read input word 0 directly
`READ_PROCESS`	`0, 10, bool`	Read input bit 10 (word 0, bit 10) from the process image

5.13 Flags

Flags are storage elements used for system data and user data.

The application program on the controller and also the station can access the flag area.

Flags can be read and written on a word or on a double word basis. Flags are addressed by the corresponding number of the word in the flag area.

NOTE
The flag area can be used for establishing communication between the application program on a Series 300 controller and a station (see chapter 5.15).

NOTE
The size of the flag area depends on the actual controller configuration.

The READ_FLAGS_WORD and WRITE_FLAGS_WORD commands are used for reading or writing individual flag words.
The READ_FLAGS_DWORD command and the WRITE_FLAGS_DWORD command are used for reading or writing double words.

Figure 5-15 shows how to read a flag word.

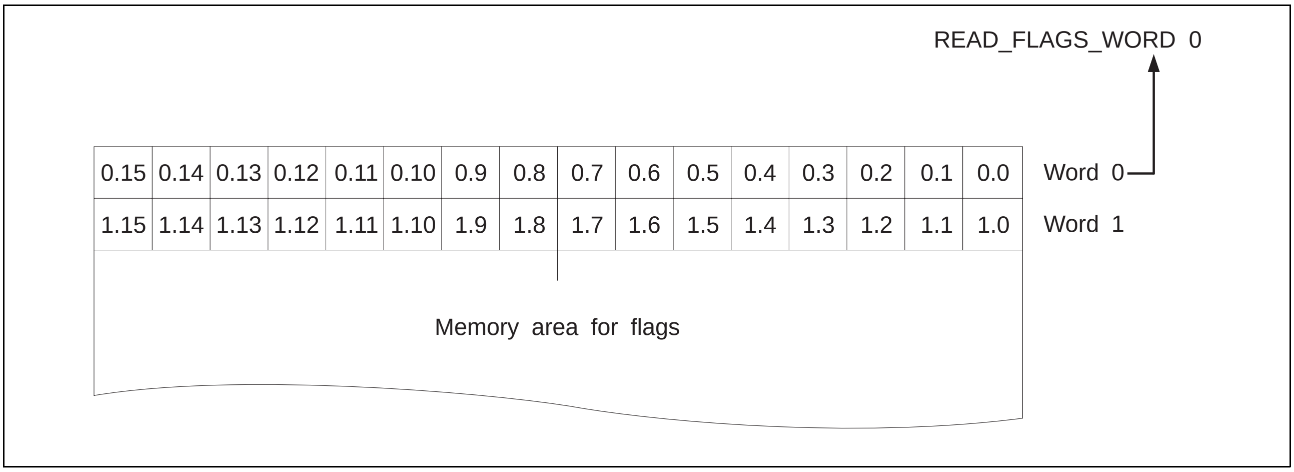


Fig. 5-15 Organisation of the flag area

Command	Function
READ_FLAGS_DWORD	Read flag as a double word from the flag area
READ_FLAGS_WORD	Read flag word from flag area
WRITE_FLAGS_DWORD	Write flag as a double word to the flag area
WRITE_FLAGS_WORD	Write flag word to the flag area

Sample commands		**Function**
`WRITE_FLAGS_WORD`	`10, 100`	Write the value 100 to the flag word 10
`WRITE_FLAGS_DWORD`	`10, 100`	Write the value 100 to the flag words 10 and 11
`READ_FLAGS_WORD`	`0`	Read flag word 0 from the flag area of the controller
`READ_FLAGS_DWORD`	`0`	Read flag words 0 and 1 from the flag area of the controller

5.14 Error

Errors are indicated by flashing numbers in the status display of the controller (see controller manual).

The station can recognize an error by the command error bit KF being set (see chapter 4.5.1.1). The type of the error is communicated to the station by means of an error code.

NOTE
The error codes and their meanings are listed in chapter 7.

The axis cannot move as long as an error is registered in the controller. Errors must first be remedied and then cleared using the CLRERROR command. The GETERROR command can be used for reading errors individually from the error memory and acknowledging them.

Command	**Function**
CLRERROR	Clear error
GETERROR	Read and acknowledge error entry

Sample commands		**Function**
`CLRERROR`	`x1`	Clear all errors on axis 1
`GETERROR`		Read error from the controller error memory

NOTE
A controller reset (RESET_PLC) also clears all active errors on a controller.

5.15 Positioning and sequence control using an application program (Series 300)

Application program

A CAN-Bus station can start and stop an application program on a Series 300 positioning and sequence controller (see fig. 5-16).
During application program execution on the controller, commands from a station can be processed simultaneously.

The START_PLC command starts an application program on a Series 300 controller.
The RESET_PLC command resets an application program, i.e. the program stops and restarts at the program beginning when it is invoked next. In addition, the RESET_PLC command stops all axis movements and resets all outputs.

NOTE
The START_PLC command is only valid for Series 300 controllers.

ATTENTION
Note that positioning operations can be initiated by the application program on the controller and by the station at the same time.

Command	Function
START_PLC	Start application program
RESET_PLC	Stop application program

Sample commands	**Function**
`START_PLC`	Start application program on controller
`RESET_PLC`	Stop application program on controller

Flag area

With Series 300 controllers, the application program on the controller as well as the station can access the flag area of the controller. This makes it possible to use the flag area for data exchange between the station and the application program on the controller (see fig. 5-16).

Command	Function
READ_FLAGS_DWORD	Read flag as a double word from the flag area
READ_FLAGS_WORD	Read flag word from flag area
WRITE_FLAGS_DWORD	Write flag as a double word to the flag area
WRITE_FLAGS_WORD	Write flag word to the flag area

Process image

With Series 300 controllers, the application program on the controller as well as the station can access the process image of the controller (fig. 5-16).

Command	Function
READ_PROCESS	Read inputs via the process image
WRITE_PROCESS	Set outputs via the process image

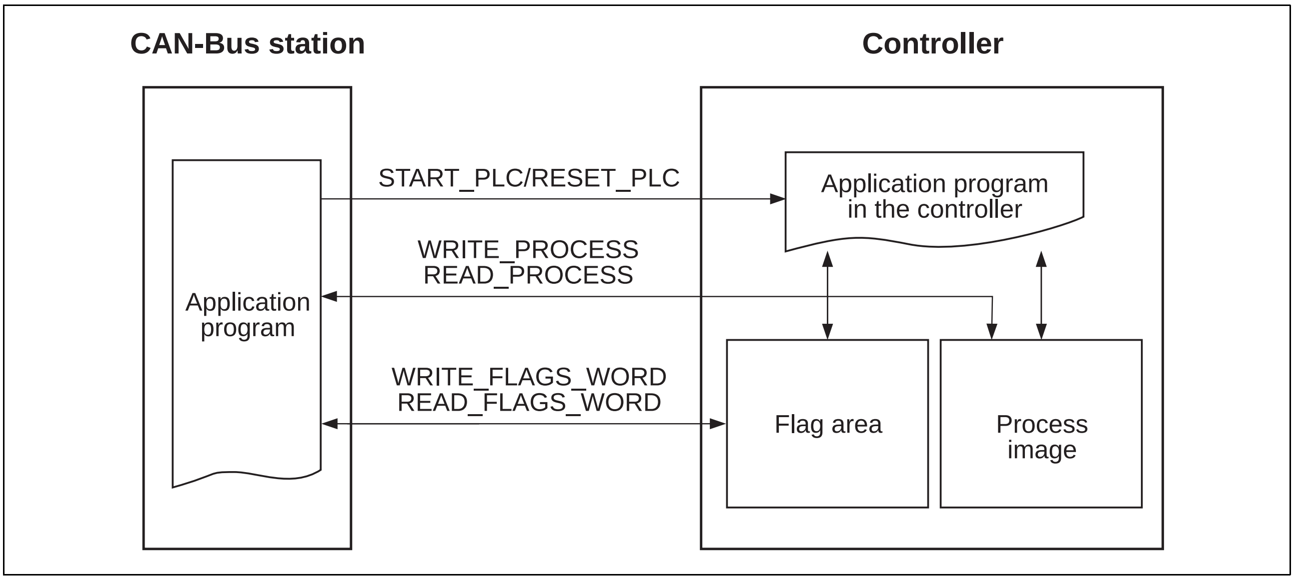
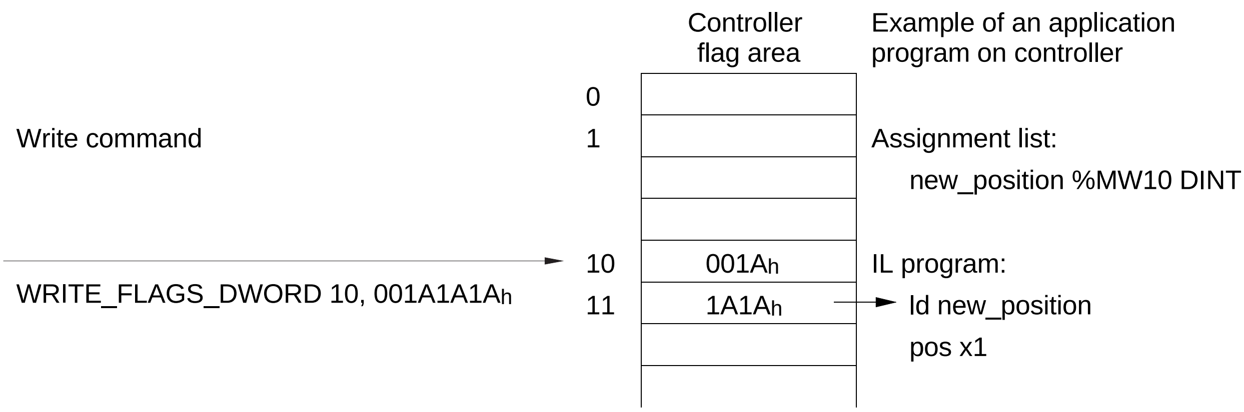


Fig. 5-16 Possibilities for accessing the flag area and the process image

Programming example for a Series 300 controller



6 Programming

6.1 Sample applications

BERGER LAHR controllers in a CAN-Bus network are controlled and monitored by the application program of another bus station.

This chapter provides sample applications for programming a BERGER LAHR controller in a CAN-Bus network.

One example is given for each axis operating mode. The examples illustrate the command sequences required for setting the appropriate operating modes and axis parameters and executing an axis movement.

6.1.1 Point-to-point mode

Sample application 1 shows how to perform a positioning operation in point-to-point mode with a few commands. For the movement parameters to be preset, such as start/stop speed, set speed, normalizing factors, electrical current values, hardware settings, etc., the defaults of the installed controller are used.

ATTENTION
Before initializing the axis with the INITDRIVE command, the nominal motor current must be set on the controller front panel or with the SETCURRENT command.

Command		**Function**
`INITDRIVE`	`x1, ampinit1`	Initialize axis 1. By default, point-to-point mode is set for the axis.
`REFPOS_LIMN`	`x1, 500`	Reference movement towards the negative limit switch. The axis approaches the limit switch at the system default set speed and acceleration ramp and clears away from the limit switch at a reference speed of 500 motor steps/s.
`POS`	`x1, 3000`	Absolute positioning operation to position 3000 motor steps: The axis moves at the set speed (1000 steps/s) and acceleration ramp (linear ramp) preset by the system.
`MOVE`	`x1, 2000`	Relative positioning operation: The axis moves by 2000 motor steps relative to the current position.

Note: By default, the normalizing factor for positions is set to 1 (numerator = 1, denominator = 1), i.e. the position specifications are equivalent to drive units (motor steps).

Sample application 2 is more comprehensive than sample application 1 and shows how to change preset movement parameters individually for point-to-point mode. Positioning operations include absolute positioning and relative positioning.

Command		Function
SETCURRENT	x1, 50, stand	Set current to 50% of maximum current for axis standstill
SETCURRENT	x1, 90, accel	Set current to 90% of maximum current for axis acceleration
SETCURRENT	x1, 75, constant	Set current to 75% of maximum current for axis constant movement
INITDRIVE	x1, ampinit1	Initialize axis 1. By default, point-to-point mode is set for the axis.
ENSIG	x1, 000A$_h$	Enable negative limit switch and STOP input monitoring. Monitoring is disabled for the positive limit switch.
SETVEL_START	x1, 200	Set start/stop speed to 200 motor steps/s
SETVEL_SYS	x1, 20000	Set maximum system speed to 20000 motor steps/s
RAMP_LIN	x1, 500	Set linear ramp: Maximum acceleration 500 motor steps/s^2
REFPOS_LIMN	x1, 400	Reference movement towards the negative limit switch. The axis approaches the limit switch at the set speed preset by the system and at the acceleration ramp set by the command and clears away from the limit switch at a reference speed of 400 motor steps/s.
VEL	x1, 3000	Set the set speed to 3000 motor steps/s
POS	x1, 3000	Absolute positioning operation to position 3000 motor steps. The axis moves at the new set speed changed by the command.
MOVE	x1, 2000	Relative axis positioning by 2000 motor steps

Note: By default, the normalizing factor for positions is set to 1 (numerator = 1, denominator = 1), i.e. the position specifications are equivalent to drive units (motor steps).

6.1.2 Speed mode

This sample application shows how to perform a positioning operation in speed mode with a few commands. For the movement parameters to be preset, such as start/stop speed, normalizing factors, electrical current values, hardware settings, etc., the defaults of the installed controller are used.

ATTENTION
Before initializing the axis with the INITDRIVE command, the nominal motor current must be set on the controller front panel or with the SETCURRENT command.

Command		Function
`INITDRIVE`	`x1, ampinit1`	Initialize axis 1
`ENSIG`	`x1, 0008h`	Enable STOP input monitoring. Disable limit switch monitoring.
`SETMODE`	`x1, velocity`	Set speed mode for axis 1
`VEL`	`x1, 1000`	Initiate movement on axis 1. The axis moves at a speed of 1000 motor steps/s. The acceleration ramp preset by the system is used.
`VEL`	`x1, 2000`	Axis 1 accelerates from speed 1000 to 2000 motor steps/s. Thereafter the axis rotates constantly at the new speed of 2000 motor steps/s.
`VEL`	`x1, 0`	Axis 1 decelerates to a speed of 0, i.e. to standstill.
`VEL`	`x1, -1000`	Axis 1 accelerates from speed 0 to 1000 motor steps/s with negative sense of rotation. Thereafter the axis rotates at a constant speed of 1000 motor steps/s in negative sense of rotation.

Note: By default, the normalizing factor for positions is set to 1 (numerator = 1, denominator = 1), i.e. the position specifications are equivalent to drive units (motor steps).

6.1.3 Position following mode

This sample application shows how to implement an electronic gear with a few commands. For the movement parameters to be preset, such as start/stop speed, electrical current values, hardware settings, etc., the defaults of the installed controller are used.

ATTENTION
Before initializing the axis with the INITDRIVE command, the nominal motor current must be set on the controller front panel or with the SETCURRENT command.

Command		Function
`INITDRIVE`	`x1, ampinit1`	Initialize axis 1
`ENSIG`	`x1, 0008h`	Enable STOP input monitoring. Disable limit switch monitoring.
`SETENCODER`	`p1, encpulsdir`	Encoder input p1 collects pulse/direction signals.
`SETMODE`	`x1, pos_drag, p1`	Set position following mode and assign encoder input p1 to axis 1.
`SETNORM_GEAR_DEN`	`x1, 1`	Set gear ratio 10, denominator = 1.
`SETNORM_GEAR_NUM`	`x1, 10`	Set gear ratio 10, numerator = 10. The motor starts as soon as pulses are received at the encoder input.
`SETOFFSET`	`x1, 100`	When the reference variable offset is changed, the motor accelerates or decelerates until the offset (relative position value) relative to the reference variable has been processed. Thereafter the axis continues to move normally.
`SETNORM_GEAR_DEN`	`x1, 1`	Set gear ratio 0, denominator = 1.
`SETNORM_GEAR_NUM`	`x1, 0`	Set gear ratio 0, numerator = 0. If a gear ratio of 0 is set, the axis does not move even if pulses are received on the encoder input.

7 Error handling

A station can respond to controller errors or errors occurring during command execution.

There are two categories of errors:

Command errors (KF)

Axis errors (XE1 to XE4)

7.1 Command errors

A command error is generated when the controller

– received an unrecognized command,
– received a new command before the previous one had been acknowledged,
– cannot execute a command.

The controller sets the command error bit (KF) in the axis status word to indicate a command error to the station.

KF = 0	No command error
KF = 1	The previous command was rejected due to an error.

When a command error occurs, the controller sends an error code to the station (see chapter 7.3). The cause of the error can be determined from the error code.

7.2 Axis errors

Axis errors are generated when a positioning operation in progress is aborted due to an error on an axis. Examples are limit switch errors, power controller failure, stop signal and contouring errors.

Axis errors are reported using the XE1, XE2, XE3 and XE4 bits in the axis status word. The cause of the error can be determined from the axis signal word.

When an axis error occurs, the controller does not send an error code to the station.

NOTE
These bits may also be set when an axis is at a standstill. One example is actuation of an enabled limit switch.

7.3 Error table

When a command error is detected (see chapter 4.6), the controller sends an error code to the station.

Axis status	Axis signals	**Error code**	–

The error table lists all error codes which may be generated when a command error occurs.

NOTE
Errors are also indicated by a number in the controller status display. For troubleshooting, see controller manual.

No. (hex)	**Error cause**
1048_h	Axis or encoder not available
1049_h	Invalid command for the axis
$104A_h$	Axis or encoder not ready
$104B_h$	Incorrect parameter value
$104C_h$	Precondition not fulfilled
$104D_h$	Value cannot be calculated
$104E_h$	Insufficient information on source
$104F_h$	Error in selection parameter
1052_h	Command only valid at standstill
1053_h	Acceleration not yet defined
1054_h	Error in acceleration curve
1055_h	Actual position not yet defined
1056_h	Invalid command, since encoder active
1058_h	Drive interrupted or blocked
1059_h	Encoder not ready
$105A_h$	Drive interrupted or blocked
$105C_h$	Reference movement active
$105F_h$	Reference movement error caused by positive limit switch
1060_h	Reference movement error caused by negative limit switch
1062_h	Cycle monitoring timeout
1069_h	Invalid output value for display
1072_h	No application program loaded
$107D_h$	Battery voltage low

No. (hex)	Error cause
$107E_h$	Short-circuit on 24 V output
$107F_h$	Invalid output address
1080_h	Invalid input address
$109B_h$	Rotation monitoring error
$109C_h$	Encoder error (line broken)
$109D_h$	Power controller readiness error
$109E_h$	Power controller overtemperature
$109F_h$	Motor overtemperature
$10A9_h$	Command only valid in point-to-point mode
$10AA_h$	Error related to software limit switch
$10AC_h$	Current program not saved in EEPROM
$10AF_h$	Positive and negative limit switch inactive
$10C1_h$	Interpolation active
$10CF_h$	Limit switch not enabled
$10D2_h$	Axis preconditions not fulfilled
$10D3_h$	Internal processing error during linear interpolation
$10D4_h$	Linear interpolation aborted because of axis error
$10DC_h$	Field bus link timeout
$10DD_h$	A new field bus command was sent before the previous one had been acknowledged
$10DE_h$	Field bus: Invalid denominator for SETNORM command
$10DF_h$	Field bus: Invalid data type
$10E0_h$	Field bus: Invalid flag word address
$10E1_h$	Field bus: Invalid command
$10E2_h$	Field bus system error
$10F5_h$	Field bus: Invalid command or command currently not permitted
1109_h	Rotation monitoring inactive

8 Write commands

This chapter lists all write commands in alphabetical order in a summary table and with a detailed description.

Write commands are those commands which initiate a control function in the controller. An example is initialization of an axis (INITDRIVE).
In contrast with read commands, write commands do not request any data from the controller.

Command	**Function**	**Series 300 controllers**	**WDP3-01X**
ACT_AXIS (25h)	Define selected axis	X	–
BRAKE (2Ah)	Define output for brake	X	X
CLRERROR (1Bh)	Clear error condition	X	X
CLRSIG_SR (1Dh)	Clear temporarily stored axis signals	X	X
CONT (1Ch)	Resume interrupted axis movement	X	X
DISP (24h)	Output in status display	X	–
ENSIG (1Eh)	Enable or disable axis signals	X	X
INITDRIVE (04h)	Initialize axis	X	X
LINMOVE (30h)	Relative linear interpolation	X	–
LINPOS (31h)	Absolute linear interpolation	X	–
MOVE (0Ah)	Relative positioning operation	X	X
POS (09h)	Absolute positioning operation	X	X
RAMP_EXP (19h)	Set exponential ramp	X	X
RAMP_LIN (18h)	Set linear ramp	X	X
RAMP_SIN (1Ah)	Set sine square ramp	X	X
REF_OUT_DISTANCE (33h)	Set maximum allowed distance from limit switch for reference movement	X	X
REFPOS_LIMN (07h)	Reference movement towards negative limit switch	X	X
REFPOS_LIMP (06h)	Reference movement towards positive limit switch	X	X
REFPOS_REF (08h)	Reference movement towards reference switch	X	X
RESET_PLC (02h)	Stop and reset application program on controller	X	–
ROTMON_DISABLE (2Dh)	Deactivate rotation monitoring	X	X
ROTMON_ENABLE (2Bh)	Activate rotation monitoring	X	X
ROTMON_RESET (2Ch)	Reset rotation monitoring	X	X
SETANALOG (32h)	Set analog output	X	–
SETCURRENT (0Dh)	Set motor current	X	X
SETENCODER (28h)	Set signal type of encoder	X	X
SETHARDWARE (26h)	Perform hardware settings	X	X
SETIPOS (2Fh)	Prepare linear interpolation	X	–

Write commands

Command	Function	Series 300 controllers	WDP3-01X
SETMODE (10h)	Set operating mode	X	X
SETNORM_GEAR_DEN (16h)	Set gear ratio denominator	X	X
SETNORM_GEAR_NUM (15h)	Set gear ratio numerator	X	X
SETNORM_POS_DEN (12h)	Normalizing factor denominator for positions	X	–
SETNORM_POS_NUM (11h)	Normalizing factor numerator for positions	X	–
SETNORM_VEL_DEN (14h)	Normalizing factor denominator for speed	X	–
SETNORM_VEL_NUM (13h)	Normalizing factor numerator for speed	X	–
SETOFFSET (27h)	Set reference variable offset	X	X
SETPOS (05h)	Set current position	X	X
SETSIG_ACTIV_H (17h)	Set active state of axis signals	X	X
SETVEL_START (0Fh)	Set start/stop speed	X	X
SETVEL_SYS (0Eh)	Set maximum system speed	X	X
START_PLC (03h)	Start application program on controller	X	–
STOP_AXIS (0Ch)	Stop axis movement	X	X
TIMEOUT (29h)	Set or disable timeout monitoring	X	X
VEL (0Bh)	Set the set speed	X	X
WRITE_FLAGS_DWORD (22h)	Write flag as a double word to the flag area	X	–
WRITE_FLAGS_WORD (21h)	Write flag word to the flag area	X	–
WRITE_OUTPUT (1Fh)	Set outputs directly	X	X
WRITE_PROCESS (20h)	Set outputs via the process image	X	–

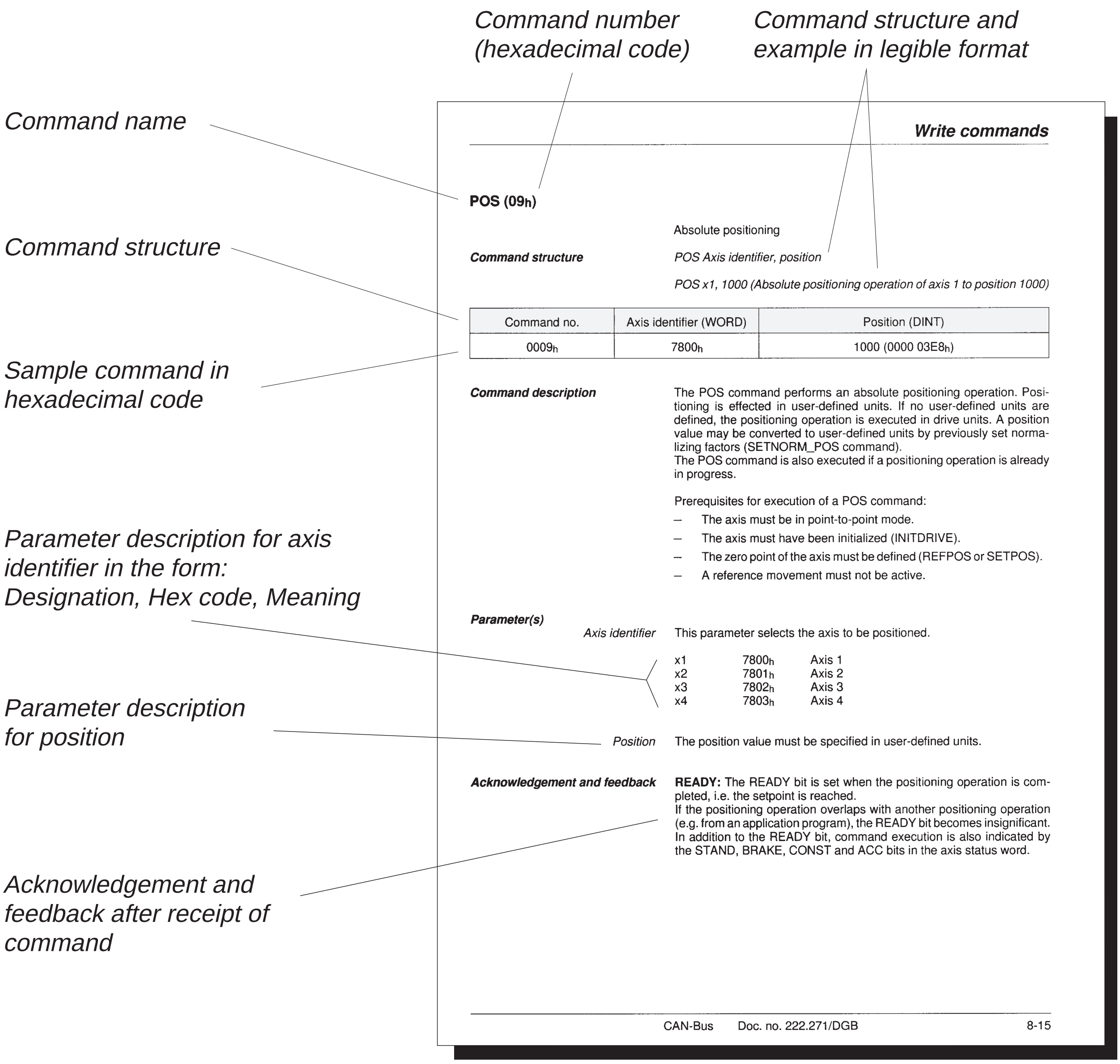
Entries in the two right-hand columns:

X Identifies commands which can be fully utilized with the specified controllers.

– Identifies commands which cannot be used with the specified controllers.

NOTE
Command execution depends on the controller type, the interface configuration (unit variant) and the operating mode setting.

The command descriptions on the following pages are structured as follows:



Write commands

POS (09h)

Absolute positioning

Command structure *POS Axis identifier, position*

POS x1, 1000 (Absolute positioning operation of axis 1 to position 1000)

Command no.	Axis identifier (WORD)	Position (DINT)
0009h	7800h	1000 (0000 03E8h)

Command description The POS command performs an absolute positioning operation. Positioning is effected in user-defined units. If no user-defined units are defined, the positioning operation is executed in drive units. A position value may be converted to user-defined units by previously set normalizing factors (SETNORM_POS command).
The POS command is also executed if a positioning operation is already in progress.

Prerequisites for execution of a POS command:

- The axis must be in point-to-point mode.
- The axis must have been initialized (INITDRIVE).
- The zero point of the axis must be defined (REFPOS or SETPOS).
- A reference movement must not be active.

Parameter(s)

Axis identifier This parameter selects the axis to be positioned.

x1	7800h	Axis 1
x2	7801h	Axis 2
x3	7802h	Axis 3
x4	7803h	Axis 4

Position The position value must be specified in user-defined units.

Acknowledgement and feedback **READY:** The READY bit is set when the positioning operation is completed, i.e. the setpoint is reached.
If the positioning operation overlaps with another positioning operation (e.g. from an application program), the READY bit becomes insignificant.
In addition to the READY bit, command execution is also indicated by the STAND, BRAKE, CONST and ACC bits in the axis status word.

CAN-Bus Doc. no. 222.271/DGB 8-15

ACT_AXIS (25_h) (Series 300 only)

Define selected axis or master axis

Command structure

ACT_AXIS Axis identifier

ACT_AXIS x2 (define axis 2 as the selected axis)

Command no.	Axis identifier (WORD)	—	—
0025_h	7801_h	0000_h	0000_h

Command description

This command is used for defining an axis as the selected axis. The axis status and the axis signals relate to the currently selected axis or to the master axis during linear interpolation.
The selected axis is the axis which last received an axis-related command. This command can be used for changing over to a different axis without initiating a controller function. The READY bit is not affected by this command and continues to be valid for the previous command passed to the axis.

NOTE
This command is only useful with multi-axis controllers (e.g. WPM-311). With single-axis controllers, axis 1 (x1) is always the selected axis.

Parameter(s)

Axis identifier Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4
l1	$6C00_h$	Linear interpolator

Acknowledgement and feedback

READY: The READY bit in the axis status word is not affected.

BRAKE ($2A_h$)

Define output for brake

Command structure

BRAKE Axis identifier, word no., bit no.

BRAKE x1, 0, 5 (Define output 5 for brake function)

Command no.	Axis identifier (WORD)	Word no. (WORD)	Bit no. (WORD)
$002A_h$	7800_h	0000_h	0005_h

Command description

This command is used for interlocking any output (Qx) of the controller with the READY and ENABLE signals of the power controller. This output can then be used directly for controlling a brake.

ATTENTION
An output with the brake function assigned can still be modified via the process image or by setting/resetting it directly from an application program.

ATTENTION
Unplugging the motor connector on the unit is not recognized and fed back to the controller by all power amplifiers by resetting the ready signal.

Parameter(s)

Axis identifier

Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Word number

Selection of the output word in which the output signal is to be used as a brake function.

Word 0	0000_h	Outputs Q0 to Q15
Word 1	0001_h	Outputs Q16 to Q31
etc.		

Bit number

Number of the output to be used for the brake function.

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

CLRERROR ($1B_h$)

Clear error condition

Command structure *CLRERROR Axis identifier*

CLRERROR x1 (Clear error condition of axis 1)

Command no.	Axis identifier (WORD)	—	—
$001B_h$	7800_h	0000_h	0000_h

Command description

This command clears the error condition (error word) of an axis or a linear interpolator. At the same time, the error indication in the controller status display is cleared.
It is also possible to clear all controller errors.

Parameter(s)

Axis identifier Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4
l1	$6C00_h$	Linear interpolator
plc	1000_h	Clear all errors on the controller

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

CLRSIG_SR (1Dh)

Clear temporarily stored axis signals

Command structure

CLRSIG_SR Axis identifier, signal template

CLRSIG_SR x1, FFFF (Clear all axis signals of axis 1)

Command no.	Axis identifier (WORD)	Signal template (WORD)	—
001Dh	7800h	FFFFh	0000h

Command description

This command clears the temporarily stored axis signals of an axis.

Parameter(s)

Axis identifier

Axis selection

x1	7800h	Axis 1
x2	7801h	Axis 2
x3	7802h	Axis 3
x4	7803h	Axis 4

Signal template

The signal template is used for clearing the corresponding temporarily stored signals if they are inactive.

limp	0001h	Positive hardware limit switch
limn	0002h	Negative hardware limit switch
ref	0004h	Reference switch input
stop	0008h	Hardware STOP input
trig	0010h	Hardware trigger input
swlimp	0020h	Positive software limit switch
swlimn	0040h	Negative software limit switch
swstop	0080h	Software stop
dragerr	0100h	Contouring error
encerr	0200h	Encoder error
ampnotready	0400h	Power controller not ready
amptemp	0800h	Power controller overtemperature
motortemp	1000h	Motor overtemperature
all	FFFFh	All signals

The signal template may comprise an OR operation for several signals.

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

CONT ($1C_h$)

Continue interrupted axis movement

Command structure

CONT Axis identifier

CONT x1 (Resume positioning operation on axis 1)

Command no.	Axis identifier (WORD)	—	—
$001C_h$	7800_h	0000_h	0000_h

Command description

This command can be used for resuming an interrupted axis movement.

NOTE
The command automatically clears the temporarily stored axis signals and then resumes the interrupted positioning operation.

Parameter(s)

Axis identifier Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

DISP (24_h)

Output to status display

Command structure *DISP Value*

DISP 10 (Output the value 10 to the status display)

Command no.	Output value (INT)	—	—
0024_h	10	0000_h	0000_h

Command description This command can be used for outputting a numerical value to the controller status display.

Parameter(s)

Output value The output value parameter is output on the status display. You can output values in the range from 0 to 99.

Acknowledgement and feedback **READY:** The READY bit in the axis status word is directly set on acknowledgement of the command.

ENSIG (1E$_h$)

Enable or disable axis signals

Command structure

ENSIG Axis identifier, signal template

ENSIG x1, 0003$_h$ (Enable negative and positive hardware limit switches of axis 1)

Command no.	Axis identifier (WORD)	Signal template (WORD)	—
001E$_h$	7800$_h$	0003$_h$	0000$_h$

Command description

This command can be used for enabling or disabling axis signals for evaluation. To enable a signal means that it is monitored by the controller. A disabled signal is not monitored by the controller.
A signal is enabled by a "1" at the corresponding position of the signal template, and it is disabled by a "0".

NOTE
The limp and limn inputs are enabled by default.

Parameter(s)

Axis identifier

Axis selection

x1	7800$_h$	Axis 1
x2	7801$_h$	Axis 2
x3	7802$_h$	Axis 3
x4	7803$_h$	Axis 4

Signal template

The signal templates are used for enabling the corresponding signals. An OR operation with several signal templates can be used for enabling any combination of signals.

limp	0001$_h$	Positive hardware limit switch
limn	0002$_h$	Negative hardware limit switch
ref	0004$_h$	Reference switch input
stop	0008$_h$	Hardware STOP input
trig	0010$_h$	Hardware trigger input
swlimp	0020$_h$	Positive software limit switch
swlimn	0040$_h$	Negative software limit switch
swstop	0080$_h$	Software stop
dragerr	0100$_h$	Contouring error
encerr	0200$_h$	Encoder error
ampnotready	0400$_h$	Power controller not ready
amptemp	0800$_h$	Power controller overtemperature
motortemp	1000$_h$	Motor overtemperature
all	FFFF$_h$	All signals

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

INITDRIVE (04_h)

Axis initialization

Command structure

INITDRIVE Axis identifier, selection

INITDRIVE x1, ampinit1

Command no.	Axis identifier (WORD)	Selection (WORD)	—
0004_h	7800_h	0000_h	0000_h

Command description

The INITDRIVE command is used for initializing the individual axes of a controller. You can select either standard or extended initialization.

Standard initialization (ampinit1):

- Switching on the power controller with internal time monitoring
- Clearing temporarily stored axis signals
- Setting the actual position of the axis to zero

Extended initialization (ampinit2) involves standard initialization and the following additional actions:

- Momentary movement of the drive
- With rotation monitoring active, the encoder position is transferred to the indexer position.

When an axis has been successfully initialized, movements can be performed, i.e. positioning and reference movement commands can be executed.

ATTENTION
Before initializing an axis with the INITDRIVE command, the nominal motor current must be set on the front panel of the controller or with the SETCURRENT command.
Controlling a brake with BRAKE and rotation monitoring with ROTMON_ENABLE should also be activated before initializing an axis.

Parameter(s)

Axis identifier

The axis identifier is used for selecting the axis

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Selection

The selection parameter selects the type of initialization.

ampinit1	0000_h	Standard initialization
ampinit2	1000_h	Extended initialization

Acknowledgement and feedback

READY: The READY bit in the axis status word is set when the command has been executed completely or if the power amplifier does not indicate readiness after a device-specific testing time (approx. 1 s).

LINMOVE (30$_h$)
(Series 300 only)

Relative linear interpolation

Command structure

LINMOVE Linear interpolator

LINMOVE l1

Command no.	Interpolator (WORD)	—
0030$_h$	6C00$_h$	0000 0000$_h$

Command description

The LINMOVE command is used for initiating a relative linear interpolation process.
With relative linear interpolation, the target positions set with SETIPOS are interpreted as relative positions to the current axis position.
The number of axes involved in linear interpolation depends on how many axes were prepared for linear interpolation with the SETIPOS command.
The linear interpolator checks whether a correct number of axes were initialized for interpolation.
The end of a linear interpolation process is indicated by the READY bit in the axis status word of the master axis.

NOTE
The LINMOVE command is only available in point-to-point mode.

Parameter(s)

Linear interpolator Selection of the linear interpolator

l1	6C00$_h$	Linear interpolator 1

Acknowledgement and feedback

READY: The READY bit in the axis status word is set as soon as the linear interpolation has been completed successfully.

However, the READY bit is no longer significant if a different field bus command was executed with any of the axes involved after starting the linear interpolation. In this case, the linear interpolation must be monitored using the STAND bit and the XE bits in the axis status word.

LINPOS (31_h)
(Series 300 only)

Absolute linear interpolation

Command structure

LINPOS Linear interpolator

LINPOS l1

Command no.	Interpolator (WORD)	—
0031_h	$6C00_h$	$0000\ 0000_h$

Command description

The LINPOS command is used for initiating an absolute linear interpolation process.
With absolute linear interpolation, the target positions set with SETIPOS are interpreted as absolute positions.
The number of axes involved in linear interpolation depends on how many axes were prepared for linear interpolation with the SETIPOS command.
The linear interpolator checks whether a correct number of axes were initialized for interpolation.
The end of a linear interpolation process is indicated by the READY bit in the axis status word of the master axis.

NOTE
The LINPOS command is only available in point-to-point mode.

Parameter(s)

Linear interpolator Selection of the linear interpolator

l1	$6C00_h$	Linear interpolator 1

Acknowledgement and feedback

READY: The READY bit in the axis status word is set as soon as the linear interpolation has been completed successfully.

However, the READY bit is no longer significant if a different field bus command was executed with any of the axes involved after starting the linear interpolation. In this case, the linear interpolation must be monitored using the STAND bit and the XE bits in the axis status word.

MOVE (0A$_h$)

Relative positioning

Command structure

MOVE Axis identifier, position

MOVE x1, 200 (Relative positioning operation of axis 1 by 200)

Command no.	Axis identifier (WORD)	Position (DINT)
000A$_h$	7800$_h$	200 (0000 00C8$_h$)

Command description

The MOVE command performs a relative positioning operation. Positioning is effected in user-defined units. A position value may be modified by previously set normalizing factors (SETNORM_POS command).
The MOVE command is also executed if a positioning operation is already in progress (e.g. from an application program).

Prerequisites for execution of a MOVE command:

– The axis must be in point-to-point mode.
– The axis must have been initialized (INITDRIVE).
– A reference movement must not be active.

Parameter(s)

Axis identifier

Axis selection

x1	7800$_h$	Axis 1
x2	7801$_h$	Axis 2
x3	7802$_h$	Axis 3
x4	7803$_h$	Axis 4

Position

The position value must be specified in user-defined units.

NOTE
If the same command (i.e. the same data) is to be transmitted twice, one following the other immediately, bit 15 must be set or reset for the (immediately following) second command to be recognized and executed (see chapter 4.6).

Acknowledgement and feedback

READY: The READY bit is set when the positioning operation is completed, i.e. the setpoint is reached.
If the positioning operation overlaps with another positioning operation (e.g. from an application program), the READY bit becomes insignificant.
In addition to the READY bit, command execution is also indicated by the STAND, BRAKE, CONST and ACC bits in the axis status word.

POS (09_h)

Absolute positioning

Command structure

POS Axis identifier, position

POS x1, 1000 (Absolute positioning operation of axis 1 to position 1000)

Command no.	Axis identifier (WORD)	Position (DINT)
0009_h	7800_h	1000 (0000 $03E8_h$)

Command description

The POS command performs an absolute positioning operation. Positioning is effected in user-defined units. If no user-defined units are defined, the positioning operation is executed in drive units. A position value may be converted to user-defined units by previously set normalizing factors (SETNORM_POS command).
The POS command is also executed if a positioning operation is already in progress.

Prerequisites for execution of a POS command:

- The axis must be in point-to-point mode.
- The axis must have been initialized (INITDRIVE).
- The zero point of the axis must be defined (REFPOS or SETPOS).
- A reference movement must not be active.

Parameter(s)

Axis identifier

This parameter selects the axis to be positioned.

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Position

The position value must be specified in user-defined units.

Acknowledgement and feedback

READY: The READY bit is set when the positioning operation is completed, i.e. the setpoint is reached.
If the positioning operation overlaps with another positioning operation (e.g. from an application program), the READY bit becomes insignificant.
In addition to the READY bit, command execution is also indicated by the STAND, BRAKE, CONST and ACC bits in the axis status word.

RAMP_EXP (19_h)

Set exponential ramp

Command structure

RAMP_EXP Axis identifier, maximum acceleration

RAMP_EXP x1, 500 (Exponential acceleration ramp for axis 1 with maximum acceleration of 500 Hz/ms)

Command no.	Axis identifier (WORD)	Maximum acceleration (DINT)
0019_h	7800_h	500 (0000 $01F4_h$)

Command description

The RAMP_EXP command is used for selecting an exponential acceleration ramp for an axis. The acceleration curve is calculated such that the values for maximum acceleration and maximum system speed are not exceeded.

This command may only be executed when the axis is at a standstill.

Parameter(s)

Axis identifier

Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Maximum acceleration

The maximum acceleration is specified in Hz/ms when the normalizing factor for speed is set to 256 (default).

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

RAMP_LIN (18h)

Set linear ramp

Command structure

RAMP_LIN Axis identifier, maximum acceleration

RAMP_LIN x1, 500 (Linear acceleration ramp for axis 1 with maximum acceleration of 500 Hz/ms)

Command no.	Axis identifier (WORD)	Maximum acceleration (DINT)
0018h	7800h	500 (0000 01F4h)

Command description

The RAMP_LIN command is used for selecting a linear acceleration ramp for an axis. The acceleration curve is calculated such that the values for maximum acceleration and maximum system speed are not exceeded.

This command may only be executed when the axis is at a standstill.

Parameter(s)

Axis identifier

Axis selection

x1	7800h	Axis 1
x2	7801h	Axis 2
x3	7802h	Axis 3
x4	7803h	Axis 4

Maximum acceleration

The maximum acceleration is specified in Hz/ms when the normalizing factor for speed is set to 256 (default).

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

RAMP_SIN ($1A_h$)

Set sine square ramp

Command structure

RAMP_SIN Axis identifier, maximum acceleration

RAMP_SIN x1, 500 (Sine square acceleration ramp for axis 1 with maximum acceleration of 500 Hz/ms)

Command no.	Axis identifier (WORD)	Maximum acceleration (DINT)
$001A_h$	7800_h	500 ($0000\ 01F4_h$)

Command description

The RAMP_SIN command is used for selecting a sine square acceleration ramp for an axis. The acceleration curve is calculated such that the values for maximum acceleration and maximum system speed are not exceeded.

This command may only be executed when the axis is at a standstill.

Parameter(s)

Axis identifier

Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Maximum acceleration

The maximum acceleration is specified in Hz/ms when the normalizing factor for speed is set to 256 (default).

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

REF_OUT_DISTANCE (33h)

Set maximum allowed distance from limit switch for reference movement

Command structure

REF_OUT_DISTANCE Axis identifier, distance

REF_OUT_DISTANCE x1, 20000 (Maximum allowed distance from limit switch or reference switch for axis 1)

Command no.	Axis identifier (WORD)	Distance (DINT)
0033h	7800h	20000 (0000 4E20h)

Command description

The REF_OUT_DISTANCE command sets the distance for the reference movement after which the axis must have cleared away from an actuated limit switch (reference switch). This distance is the maximum allowed distance from the limit switch (reference switch). If the limit switch (reference switch) is still actuated after passing the maximum allowed distance, the reference movement is aborted and an error generated. The error is indicated by the ref_err bit in the axis signal word.

Default: See the controller manual

Parameter(s)

Axis identifier Axis selection

x1	7800h	Axis 1
x2	7801h	Axis 2
x3	7802h	Axis 3
x4	7803h	Axis 4

Distance Maximum allowed distance from the limit switch or reference switch
The distance is indicated in drive units (motor steps).

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

REFPOS_LIMN (07h)

Reference movement towards negative limit switch

Command structure

REFPOS_LIMN Axis identifier, reference speed

REFPOS_LIMN x1, 100 (Reference movement of axis 1 towards negative limit switch at reference speed of 100 Hz)

Command no.	Axis identifier (WORD)	Reference speed (DINT)
0007h	7800h	100 (0000 0064h)

Command description

The REFPOS_LIMN command performs a reference movement towards the negative limit switch. The motor approaches the limit switch at set speed and then moves in the opposite direction towards the reference point at reference speed.

Reference movements are only possible in point-to-point mode.

Parameter(s)

Axis identifier

Axis selection

x1	7800h	Axis 1
x2	7801h	Axis 2
x3	7802h	Axis 3
x4	7803h	Axis 4

Reference speed

The speed at which the motor clears away from the limit switch.

Acknowledgement and feedback

READY: The READY bit in the axis status word is set when the reference movement has been processed completely.

Execution of the command is reflected in the two bits REF_OK in the axis status word and ref_err in the axis signal word in addition.

REFPOS_LIMP (06h)

Reference movement towards positive limit switch

Command structure

REFPOS_LIMP Axis identifier, reference speed

REFPOS_LIMP x1, 100 (Reference movement of axis 1 towards positive limit switch at reference speed of 100 Hz)

Command no.	Axis identifier (WORD)	Reference speed (DINT)
0006h	7800h	100 (0000 0064h)

Command description

The REFPOS_LIMP command performs a reference movement towards the positive limit switch. The motor approaches the limit switch at set speed and then moves in the opposite direction towards the reference point at reference speed.

Reference movements are only possible in point-to-point mode.

Parameter(s)

Axis identifier

Axis selection

x1	7800h	Axis 1
x2	7801h	Axis 2
x3	7802h	Axis 3
x4	7803h	Axis 4

Reference speed

The speed at which the motor clears away from the limit switch.

Acknowledgement and feedback

READY: The READY bit in the axis status word is set when the reference movement has been processed completely.

Execution of the command is reflected in the two bits REF_OK in the axis status word and ref_err in the axis signal word in addition.

REFPOS_REF (08h)

Reference movement towards the reference switch

Command structure

REFPOS_REF Axis identifier, reference speed

REFPOS_REF x1, 100 (Reference movement of axis 1 towards the reference switch at a reference speed of 100 Hz)

Command no.	Axis identifier (WORD)	Reference speed (DINT)
0008h	7800h	100 (0000 0064h)

Command description

The REFPOS_REF command performs a reference movement towards the reference switch. The motor approaches the reference switch at set speed and then moves in the opposite direction towards the reference point at reference speed.

Reference movements are only possible in point-to-point mode.

Parameter(s)

Axis identifier

Axis selection

x1	7800h	Axis 1
x2	7801h	Axis 2
x3	7802h	Axis 3
x4	7803h	Axis 4

Reference speed

This parameter is used for setting the reference speed. The motor clears away from the limit switch at this speed.

Bit 31 of the reference speed defines the direction in which the reference switch is approached.

Bit 31 = 0
Approaches the reference switch towards the right.
e.g. 0000 0064h Movement to the right at a reference speed of 100 Hz

Bit 31 = 1
Approaches the reference switch towards the left.
e.g. 8000 0064h Movement to the left at a reference speed of 100 Hz

Acknowledgement and feedback

READY: The READY bit in the axis status word is set when the reference movement has been processed completely.

Execution of the command is reflected in the two bits REF_OK in the axis status word and ref_err in the axis signal word in addition.

RESET_PLC (02_h)
(Series 300 only)

Stop and reset application program on controller

Command structure

RESET_PLC

RESET_PLC (Carry out a RESET on the controller addressed)

Command no.	—	—	—
0002_h	0000_h	0000_h	0000_h

Command description

The RESET_PLC command performs a RESET on the controller. This involves stopping all axis movements and resetting all outputs. Any running application program is set to RESET status.

NOTE
A brake function initiated with the BRAKE command is deactivated.

Parameter(s)

None

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

ROTMON_DISABLE (2Dh)

Disable rotation monitoring

Command structure

ROTMON_DISABLE Axis identifier, encoder

ROTMON_DISABLE x1, p2 (Disable rotation monitoring of axis 1 via encoder 2)

Command no.	Axis identifier (WORD)	Encoder (WORD)	—
002Dh	7800h	7001h	0000h

Command description

This command is used for deactivating rotation monitoring for an axis on any encoder interface. Rotation monitoring then does not take place any longer.

Parameter(s)

Axis identifier

Axis selection

x1	7800h	Axis 1
x2	7801h	Axis 2
x3	7802h	Axis 3
x4	7803h	Axis 4

Encoder

Encoder interface selection

p1*	7000h	Encoder 1
p2	7001h	Encoder 2
pext*	7008h	Encoder on external power controller

* Not applicable on WDP3-01X, WPM-311 and WDPM3-314 controllers

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

ROTMON_ENABLE ($2B_h$)

Activate rotation monitoring

Command structure

ROTMON_ENABLE Axis identifier, encoder, resolution

ROTMON_ENABLE x1, p2, 1000 (Rotation monitoring for axis 1 via encoder 2 with an encoder resolution of 1000 marks)

Command no.	Axis identifier (WORD)	Encoder (WORD)	Resolution (WORD)
$002B_h$	7800_h	7001_h	$03E8_h$

Command description

This command is used for initializing rotation monitoring for an axis on any encoder interface.

Parameter(s)

Axis identifier

Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Encoder

Encoder interface selection

p1*	7000_h	Encoder 1
p2	7001_h	Encoder 2
pext*	7008_h	Encoder on external power controller

* Not applicable on WDP3-01X, WPM-311 and WDPM3-314 controllers

Resolution

Encoder resolution

500	$1F4_h$	500 marks
1000	$3E8_h$	1000 marks

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

ROTMON_RESET (2Ch)

Reset rotation monitoring

Command structure

ROTMON_RESET Axis identifier, encoder

ROTMON_RESET x1, p2 (Reset rotation monitoring of axis 1 via encoder 2)

Command no.	Axis identifier (WORD)	Encoder (WORD)	—
002Ch	7800h	7001h	0000h

Command description

This command is used for resetting rotation monitoring for an axis on any encoder interface. Any active rotation monitoring error is cleared. The current encoder position is used as the current axis position.

Parameter(s)

Axis identifier

Axis selection

x1	7800h	Axis 1
x2	7801h	Axis 2
x3	7802h	Axis 3
x4	7803h	Axis 4

Encoder

Encoder interface selection

p1*	7000h	Encoder 1
p2	7001h	Encoder 2
pext*	7008h	Encoder on external power controller

* Not applicable on WDP3-01X, WPM-311 and WDPM3-314 controllers

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

SETANALOG (32_h)
(Series 300 only)

Set analog output

Command structure

SETANALOG Analog module, channel, voltage

SETANALOG a2, 1, 5000

Command no.	Analog module (WORD)	Channel (WORD)	Voltage (INT)
0032_h	6101_h	0001_h	5000 ($0000\ 1388_h$)

Command description

The SETANALOG command can be used for outputting voltages via an analog output. The ANOZ analog module (see controller manual) has one output for this purpose.
The analog output is addressed by the identifier of the analog module and the channel number of the output.
Voltages are specified in millivolts (mV).

Parameter(s)

Analog module ID

Selection of the analog module

a2	6101_h	Analog module (2nd adapter slot)

Channel

Selection of the analog output of the analog module

1	0001_h	Analog output 1

Voltage

Value of voltage output via the analog output. The voltage is specified in millivolts (mV); the data type is INT.

The range of values of the voltage is specified in the controller manual for the Series 300 controllers.

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

SETCURRENT ($0D_h$)

Set the motor current

Command structure

SETCURRENT Axis identifier, electrical current value, selection

SETCURRENT x1, 75, stand (Set 75% of maximum current for axis 1 at standstill)

Command no.	Axis identifier (WORD)	El. current value (INT)	Selection (WORD)
$000D_h$	7800_h	75 ($004B_h$)	0008_h

Command description

The SETCURRENT command can be used for setting the electrical current values for various movement states of an axis.
Electrical current values can be set with the SETCURRENT command in the range from 0% to 100% of the maximum current.
Maximum current means:

– for the WDP3-014 = 2.5 A
– for the WDP3-018 = 6.8 A
– for Series 300 controllers = The current set on the front panel rotary switch; see controller manual.

ATTENTION
The set motor current for standstill or constant movement of the axis must be equal to or less than the motor current specified on the motor type plate (the lower the set motor current, the lower the motor torque).

Parameter(s)

Axis identifier — Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Electrical current value — The electrical current value specifies the current as a percentage of the maximum current.

Selection — Selection of the movement status for which the set current is to be effective.

stand	0008_h	Axis standstill
accel	0001_h	Acceleration or deceleration
constant	0002_h	Constant movement

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

SETENCODER (28_h)

Set encoder signal type

Command structure

SETENCODER Encoder ID, selection

SETENCODER p1, encpulsdir (Encoder 1 evaluates pulse/direction signals)

Command no.	Encoder ID (WORD)	Selection (WORD)	—
0028_h	7000_h	0005_h	0000_h

Command description

The SETENCODER command is used for setting the signal type for an encoder.

An encoder can operate with the following signal types:

– Detect pulse/direction signals
– Detect A/B signals

This setting is required for implementing an electronic gear.

Parameter(s)

Encoder ID

Encoder selection

p1*	7000_h	Encoder 1
p2	7001_h	Encoder 2

* Not applicable on WDP3-01X, WPM-311 and WDPM3-314 controllers

Selection

The selection parameter is used for setting the signal type for the encoder.

encquad	0002_h	Detect A/B signals (quadruple resolution)
encdouble	0003_h	Detect A/B signals (double resolution)
encsingle	0004_h	Detect A/B signals (single resolution)
encpulsdir	0005_h	Detect pulse/direction signals (default)

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

SETHARDWARE (26_h)

Perform hardware settings

Command structure

SETHARDWARE Axis identifier, selection

SETHARDWARE x1, ampon (Enable power controller for axis 1)

Command no.	Axis identifier (WORD)	Selection (WORD)	—
0026_h	7800_h	0003_h	0000_h

Command description

The SETHARDWARE command can be used for specific hardware settings.

⚠ ***ATTENTION***
Hardware settings can only be adjusted when the axis is at a standstill.

Parameter(s)

Axis identifier

Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Selection

The selection parameter selects the type of hardware setting.

pulson	0001_h	Activate pulse output to power controller (default)
pulsoff	0002_h	Deactivate pulse output to power controller
ampon	0003_h	Enable power controller
ampoff	0004_h	Disable power controller (default)
dirinvert	0005_h	Invert the motor’s sense of rotation
dirnormal	0006_h	Set sense of rotation to basic setting (default)

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

SETIPOS ($2F_h$)
(Series 300 only)

Prepare linear interpolation

Command structure

SETIPOS Axis identifier, position

SETIPOS x1, 5000

Command no.	Axis identifier (WORD)	Position (DINT)
$002F_h$	7800_h	5000 (0000 1388_h)

Command description

The SETIPOS command is used for preparing an axis for a linear interpolation process.
A linear interpolation may involve 2 or 3 axes. A target position must be set with SETIPOS for each axis involved in linear interpolation.
The linear interpolation process is initiated with the LINPOS or LINMOVE command.
The target positions must be set again with SETIPOS before each linear interpolation process.

NOTE
The SETIPOS command is only available in point-to-point mode.

Parameter(s)

Axis identifier

Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Position

Specifies the target position for linear interpolation

The target position is specified in user-defined units as a hexadecimal value with the DINT data type.

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

SETMODE (10_h)

Set the operating mode

Command structure

SETMODE Axis identifier, operating mode, source

SETMODE x1, velocity (Set speed mode on axis 1)

Command no.	Axis identifier (WORD)	Operating mode (WORD)	Source (WORD)
0010_h	7800_h	0002_h	0000_h

Command description

The SETMODE command is used for setting the operating mode for an axis.

Three different operating modes are available:

- Point-to-point mode
- Speed mode
- Position following mode

Default: Point-to-point mode

In position following mode, the encoder can be selected for implementing an electronic gear.

Parameter(s)

Axis identifier

Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Operating mode

Operating mode selection

ptp	0001_h	Point-to-point mode
velocity	0002_h	Speed mode
pos_drag	0003_h	Position following mode

Source

Select an encoder connection in position following mode, e.g. for an electronic gear.

p1*	7000_h	Encoder connection 1
p2	7001_h	Encoder connection 2

* Not applicable on WDP3-01X, WPM-311 and WDPM3-314 controllers

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

SETNORM_GEAR_DEN (16_h)

Set gear ratio denominator

Command structure

SETNORM_GEAR_DEN Axis identifier, denominator

SETNORM_GEAR_DEN x1, 10 (Set denominator = 10 for the gear ratio of axis 1)

Command no.	Axis identifier (WORD)	Denominator (DINT)
0016_h	7800_h	10 (0000 $000A_h$)

Command description

The SETNORM_GEAR_DEN command is used for setting the denominator for the gear ratio of an axis in position following mode.

NOTE
The denominator of the normalizing factor must be passed prior to the numerator. The denominator is not accepted until the numerator is passed.

Default: Numerator = 0 and Denominator = 1 (axis at standstill)

Parameter(s)

Axis identifier Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Denominator Value of the denominator in the gear ratio

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

SETNORM_GEAR_NUM (15h)

Set gear ratio numerator

Command structure

SETNORM_GEAR_NUM Axis identifier, numerator

SETNORM_GEAR_NUM x1, 1 (Set numerator = 1 for the gear ratio of axis 1 and accept gear ratio)

Command no.	Axis identifier (WORD)	Numerator (DINT)
0015_h	7800_h	1 (0000 0001_h)

Command description

The SETNORM_GEAR_NUM command is used for setting the numerator for the gear ratio of an axis in position following mode.

NOTE
The denominator of the normalizing factor must be passed prior to the numerator. The denominator is not accepted until the numerator is passed.

Default: Numerator = 0 and Denominator = 1 (axis at standstill)

Parameter(s)

Axis identifier

Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Numerator

Value of the numerator in the gear ratio

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

SETNORM_POS_DEN (12_h) (Series 300 only)

Set normalizing factor denominator for positions

Command structure

SETNORM_POS_DEN Axis identifier, denominator

SETNORM_POS_DEN x1, 1 (Denominator = 1 for the normalizing factor for positions of axis 1)

Command no.	Axis identifier (WORD)	Denominator (DINT)
0012_h	7800_h	1 (0000 0001_h)

Command description

The SETNORM_POS_DEN command is used for setting the normalizing factor denominator for positions.
This normalizing factor is used for converting position values given in user-defined units into controller-specific drive units.

Drive units = User-defined units x Normalizing factor

NOTE
The denominator of the normalizing factor must be passed prior to the numerator. The denominator is not accepted until the numerator is passed.

Default: Numerator = 1 and Denominator = 1 (i.e. user-defined units are equivalent to drive units, one user-defined unit is equivalent to one motor step)

Parameter(s)

Axis identifier

Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Denominator

Value of the denominator in the normalizing factor

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

SETNORM_POS_NUM (11_h) (Series 300 only)

Set normalizing factor numerator for positions

Command structure

SETNORM_POS_NUM Axis identifier, numerator

SETNORM_POS_NUM x1, 10 (Set numerator = 10 for the normalizing factor for positions of axis 1)

Command no.	Axis identifier (WORD)	Numerator (DINT)
0011_h	7800_h	10 (0000 $000A_h$)

Command description

The SETNORM_POS_NUM command is used for setting the normalizing factor numerator for positions.
This normalizing factor is used for converting position values given in user-defined units into controller-specific drive units.

Drive units = User-defined units x Normalizing factor

NOTE
The denominator of the normalizing factor must be passed prior to the numerator. The denominator is not accepted until the numerator is passed.

Default: Numerator = 1 and Denominator = 1 (i.e. user-defined units are equivalent to drive units, one user-defined unit is equivalent to one motor step)

Parameter(s)

Axis identifier

Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Numerator

Value of the numerator in the normalizing factor

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

SETNORM_VEL_DEN (14_h) (Series 300 only)

Set normalizing factor denominator for speeds

Command structure

SETNORM_VEL_DEN Axis identifier, denominator

SETNORM_VEL_DEN x1, 1 (Denominator = 1 for the normalizing factor for speeds of axis 1)

Command no.	Axis identifier (WORD)	Denominator (DINT)
0014_h	7800_h	1 (0000 0001_h)

Command description

The SETNORM_VEL_DEN command is used for setting the normalizing factor denominator for speeds.
This normalizing factor is used for converting speed values given in user-defined units into controller-specific drive units.

$$\text{Drive units} = \text{User-defined units} \times \frac{\text{Normalizing factor}}{256}$$

NOTE
The denominator of the normalizing factor must be passed prior to the numerator. The denominator is not accepted until the numerator is passed.

Default: Numerator = 256 and Denominator = 1 (256/1, i.e. speed values can be specified in hertz, one user-defined unit is equivalent to one motor step/s)

Parameter(s)

Axis identifier

Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Denominator

Value of the denominator in the normalizing factor

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

SETNORM_VEL_NUM (13_h) (Series 300 only)

Set normalizing factor numerator for speeds

Command structure

SETNORM_VEL_NUM Axis identifier, numerator

SETNORM_VEL_NUM x1, 10 (Numerator = 10 for the normalizing factor for speeds of axis 1)

Command no.	Axis identifier (WORD)	Numerator (DINT)
0013_h	7800_h	10 (0000 $000A_h$)

Command description

The SETNORM_VEL_NUM command is used for setting the normalizing factor numerator for speeds.
This normalizing factor is used for converting speed values given in user-defined units into controller-specific drive units.

$$\text{Drive units} = \text{User-defined units} \times \frac{\text{Normalizing factor}}{256}$$

NOTE
The denominator of the normalizing factor must be passed prior to the numerator. The denominator is not accepted until the numerator is passed.

Default: Numerator = 256 and Denominator = 1 (256/1, i.e. speed values can be specified in hertz, one user-defined unit is equivalent to one motor step/s)

Parameter(s)

Axis identifier Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Numerator Value of the numerator in the normalizing factor

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

SETOFFSET (27_h)

Set reference variable offset

Command structure

SETOFFSET Axis identifier, offset

SETOFFSET x1, 10 (Set reference variable offset of axis 1 to 10)

Command no.	Axis identifier (WORD)	Offset (DINT)
0027_h	7800_h	10 ($0000\ 000A_h$)

Command description

The SETOFFSET command can be used for overlaying the reference variable of an electronic gear with an offset.
Changing the offset accelerates or decelerates the axis. When the offset has been processed, the axis continues to run normally.

The following relationship applies:

Drive units = Offset + (Reference variable x Gear ratio)

Parameter(s)

Axis identifier

Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Offset

The offset parameter specifies the offset value in user-defined units. The offset value defines a relative position by which the axis position lags behind the reference variable.

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

SETPOS (05_h)

Set current position

Command structure

SETPOS Axis identifier, position

SETPOS x1, 5000 (Set current position of axis 1 to 5000)

Command no.	Axis identifier (WORD)	Position (DINT)
0005_h	7800_h	5000 (0000 1388_h)

Command description

The SETPOS command is used for setting a new reference point for absolute positioning operations in point-to-point mode (dimension setting).

NOTE
This command may only be executed when the axis is at a standstill.

Parameter(s)

Axis identifier

Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Position

Position value for the current position. The position value is specified in user-defined units.

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

SETSIG_ACTIV_H (17_h)

Set active state of axis signals

Command structure

SETSIG_ACTIV_H Axis identifier, signal template

SETSIG_ACTIV_H x1, 0007_h (The limp, limn, ref and trig signals of axis 1 are active when the associated inputs are energized.)

Command no.	Axis identifier (WORD)	Signal template (WORD)	—
0017_h	7800_h	0007_h	0000_h

Command description

The SETSIG_ACTIV_H command is used for setting the active states (active low, active high) of the limp, limn, ref and trig axis monitoring signals.
If the signal template contains a 0 at the bit position of the associated signal, it is interpreted to be active when the input is deenergized.
If the signal template contains a 1 at the bit position of the associated signal, it is interpreted to be active when the input is energized.

Default:
The limp, limn and ref inputs are active when deenergized.
The trig input is active when energized.

Parameter(s)

Axis identifier

Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Signal template

The signal template can be used for setting the corresponding signal states.
When using a bit-by-bit OR operation with the signal templates, several signal states can be set simultaneously, e.g. 0003_h means limp and limn.

limp	0001_h	Positive hardware limit switch
limn	0002_h	Negative hardware limit switch
ref	0004_h	Reference switch input
trig	0010_h	Hardware trigger input

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

SETVEL_START (0Fh)

Set start/stop speed

Command structure

SETVEL_START Axis identifier, start/stop speed

SETVEL_START x1, 200 (Set start/stop speed to 200 Hz for axis 1)

Command no.	Axis identifier (WORD)	Start/stop speed (DINT)
000Fh	7800h	200 (0000 00C8h)

Command description

The SETVEL_START command is used for setting the start/stop speed for an axis.
The start/stop speed is used by the motor to start from standstill and to decelerate to zero without using the acceleration curve.
The start/stop speed may only be changed when the axis is at a standstill.

Default: See controller manual

⚠ ***ATTENTION***
On WDP3-01X controllers, the start/stop speed must be set to a value < 800 Hz.

Parameter(s)

Axis identifier

Axis selection

x1	7800h	Axis 1
x2	7801h	Axis 2
x3	7802h	Axis 3
x4	7803h	Axis 4

Start/stop speed

The start/stop speed must be specified in user-defined units.

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

SETVEL_SYS ($0E_h$)

Set the maximum system speed

Command structure *SETVEL_SYS Axis identifier, system speed*

SETVEL_SYS x1, 20000 (Maximum system speed 20000 Hz for axis 1)

Command no.	Axis identifier (WORD)	System speed (DINT)
$000E_h$	7800_h	20000 (0000 $4E20_h$)

Command description

The SETVEL_SYS command is used for setting the maximum system speed for an axis.
The maximum system speed is the speed which must never be exceeded in axis movements.
The maximum system speed may only be changed when the axis is at a standstill.

NOTE
When calculating acceleration curves, the curve is only calculated up to the set maximum system speed. This implies that a new acceleration curve must be defined whenever the maximum system speed is changed (see RAMP_LIN, RAMP_EXP, RAMP_SIN commands).

Default: See controller manual

Parameter(s)

Axis identifier Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

System speed The system speed must be specified in user-defined units.

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

START_PLC (03h)
(Series 300 only)

Start application program on controller

Command structure

START_PLC

START_PLC (Start application program on controller)

Command no.	—	—	—
0003h	0000h	0000h	0000h

Command description

The START_PLC command is used for starting an application program on the controller.

NOTE
If no application program is loaded on the controller, a command error with an error code is output.

Parameter(s)

None

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

STOP_AXIS ($0C_h$)

Stop axis movement or linear interpolation

Command structure

STOP_AXIS Axis identifier

STOP_AXIS x1 (Stop axis 1)

Command no.	Axis identifier (WORD)	—	—
$000C_h$	7800_h	0000_h	0000_h

Command description

The STOP_AXIS command is used for stopping the movement of an axis or a linear interpolation.
The swstop axis monitoring signal is set to active.

NOTE
Interrupted axis movement can be resumed using the CONT command; this is not possible during linear interpolation. As part of this procedure, all temporarily stored axis monitoring signals are reset automatically.

Parameter(s)

Axis identifier Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4
l1	$6C00_h$	Linear interpolator (Series 300 only)

Acknowledgement and feedback

READY: The READY bit in the axis status word is set as soon as the axis has stopped. In a linear interpolation process, the READY bit is not set until all axes are at a standstill.

In addition to the READY bit, execution of this command can also be monitored with the STAND, BRAKE, CONST and ACC bits in the axis status word.

NOTE
If the STOP_AXIS command overlaps with another positioning command (e.g. from a controller application program), the READY bit becomes insignificant.

TIMEOUT (29_h)

Set or disable timeout monitoring (bus monitoring)

Command structure

TIMEOUT Time

TIMEOUT 256 (Set timeout to 256 ms)

Command no.	Time (WORD)	—	—
0029_h	0100_h	0000_h	0000_h

Command description

The TIMEOUT command is used for setting the monitoring time between the reception of any two command blocks or for disabling monitoring.

Default: 0 = Timeout monitoring OFF

Parameter(s)

In case of a timeout, the controller initiates a RESET (power amplifier deenergized, outputs reset).

Time Monitoring time in ms

Range of values: 0 to 65535 ms

Time = 0 disables timeout monitoring.

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

ATTENTION
Since the CAL protocol has its own monitoring methods, this timeout monitoring method is only useful with the simple CAN protocol.

VEL ($0B_h$)

Set the set speed

Command structure

VEL Axis identifier, set speed

VEL x1, 2000 (Set the set speed of axis 1 to 2000)

Command no.	Axis identifier (WORD)	Set speed (DINT)
$000B_h$	7800_h	2000 (0000 $07D0_h$)

Command description

The VEL command is used in point-to-point mode for setting the set speed for an axis.

In speed mode, the VEL command modifies the set speed, i.e. it changes the speed during an axis movement.

Default: See controller manual

Parameter(s)

Axis identifier

Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Set speed

The set speed is the speed at which an axis moves in point-to-point mode and speed mode.
The set speed must be specified in user-defined units.

Acknowledgement and feedback

READY:
Point-to-point mode:
The READY bit in the axis status word is directly set on acknowledgement of the command.

Speed mode:
The READY bit in the axis status word is set as soon as the new speed is reached.

WRITE_FLAGS_DWORD (22_h) (Series 300 only)

Write flag as a double word to the flag area

Command structure

WRITE_FLAGS_DWORD Flag number, flag data

WRITE_FLAGS_DWORD 10, $001A1A1A_h$ (Write value $001A1A1A_h$ to the flag words 10 and 11 in the flag area)

Command no.	Flag number (WORD)	Flag data (DWORD)
0022_h	$000A_h$	$001A1A1A_h$

Command description

The WRITE_FLAGS_DWORD command writes two flag words as a double word to the flag area on the controller.

NOTE
This command can be used for implementing data exchange between an application program on the controller and an application program on the CAN-Bus station.

Parameter(s)

Flag number — Number of the flag word in the flag area on the controller. Two flag words are written to the flag area starting with this number.

Flag data — This data value is written to the two flag words.

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

NOTE
The data can be read with the DINT data type by the application program on a Series 300 controller.

WRITE_FLAGS_WORD (21_h) (Series 300 only)

Write flag word to the flag area

Command structure

WRITE_FLAGS_WORD Flag number, flag data

WRITE_FLAGS_WORD 10, $001A_h$ (Write value $001A_h$ to the flag word 10 in the flag area)

Command no.	Flag number (WORD)	Flag data (WORD)	—
0021_h	$000A_h$	$001A_h$	—

Command description

The WRITE_FLAGS_WORD command writes a flag word to the flag area.

NOTE
This command can be used for implementing data exchange between an application program on the controller and an application program on the CAN-Bus station.

Parameter(s)

Flag number Number of the flag word in the flag area on the controller. The flag data are written into this flag word in the flag area.

Flag data This data value is written to the flag word.

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

WRITE_OUTPUT ($1F_h$)

Set outputs directly

Command structure

WRITE_OUTPUT Word no., bit no., data type, signal state

WRITE_OUTPUT 0, 5, bool, 1 (Set output 5 directly)

Command no.	Word no. (WORD)	Bit no. (BYTE)	Data type (BYTE)	Signal state (WORD)
$001F_h$	0000_h	05_h	00_h	0001_h

Command description

This command can be used for setting or resetting all outputs of a controller directly.
The outputs can be changed bit by bit (individually) or word by word (16 bits).

NOTE
Only outputs included in the process image of the controller can be addressed. External inputs/outputs can be addressed if they are identified to the controller via the application program.

Parameter(s)

Word number

Selection of the output word in which the output signals are set or reset. Selection is effected by specifying a word number.

Word 0	0000_h	Outputs Q0 to Q15
Word 1	0001_h	Outputs Q16 to Q31 (ext. inputs/outputs)
etc.		

Bit number

Number of the bit to be set or reset. The number refers to the bit position in the selected output word. The bit number specification is only significant for the **bool** data type (see below).

Data type

Choose bit-by-bit or word-by-word setting or resetting of the outputs

Valid data types

bool	00_h	bit by bit
word	02_h	word by word

Signal status

Bit pattern of the signal states for the outputs to be set or reset

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

WRITE_PROCESS (20_h) (Series 300 only)

Set outputs via the process image

Command structure

WRITE_PROCESS Word no., bit no., data type, signal state

WRITE_PROCESS 0, 5, bool, 1 (Set output 5 via the process image)

Command no.	Word no. (WORD)	Bit no. (BYTE)	Data type (BYTE)	Signal state (WORD)
0020_h	0000_h	05_h	00_h	0001_h

Command description

This command can be used for setting or resetting all outputs of a controller via the process image.
The outputs can be changed bit by bit (individually) or word by word (16 bits).

Parameter(s)

Word number

Selection of the output word or output byte in which output signals are set or reset.

Word 0	0000_h	Outputs Q0 to Q15
Word 1	0001_h	Outputs Q16 to Q31
etc.		

Bit number

Number of the bit to be set or reset. The number refers to the bit position in the selected output word. The bit number specification is only significant for the **bool** data type (see below).

Data type

Choose bit-by-bit or word-by-word setting or resetting of the outputs

Valid data types

bool	00_h	bit by bit
word	02_h	word by word

Signal status

Bit pattern of the signal states for the outputs to be set or reset

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

9 Read commands

This chapter lists all read commands in alphabetical order in a summary table and with a detailed description.

Read commands are those commands which read data from a controller. Read data requested in this way are transmitted to the station in addition to the axis status and the axis signals.

When the controller receives and recognizes a read command from a station, it transmits the read data (e.g. position values) to the station.

Command	**Function**	**Series 300 controllers**	**WDP3-01X**
GETANALOG (61h)	Read analog input	X	–
GETCURRENT (53h)	Read electrical current values	X	X
GETDATA (60h)	Repeat read data request	X	X
GETENSIG (54h)	Determine enabled or disabled axis signals	X	X
GETERROR (5Fh)	Read error	X	X
GETMODE (58h)	Read operating mode	X	X
GETPOS (50h)	Read position values	X	X
GETSIG (55h)	Read current axis signal states	X	X
GETSIG_ACTIV_H (57h)	Read active state of axis signals	X	X
GETSIG_SR (56h)	Read temporarily stored axis signals	X	X
GETSTATE (52h)	Read error state of an axis	X	X
GETVEL (51h)	Read speed value	X	X
READ_FLAGS_DWORD (5Dh)	Read flag as a double word from the flag area	X	–
READ_FLAGS_WORD (5Ch)	Read flag word from flag area	X	–
READ_INPUT (5Ah)	Read inputs directly	X	X
READ_PROCESS (5Bh)	Read inputs via the process image	X	–

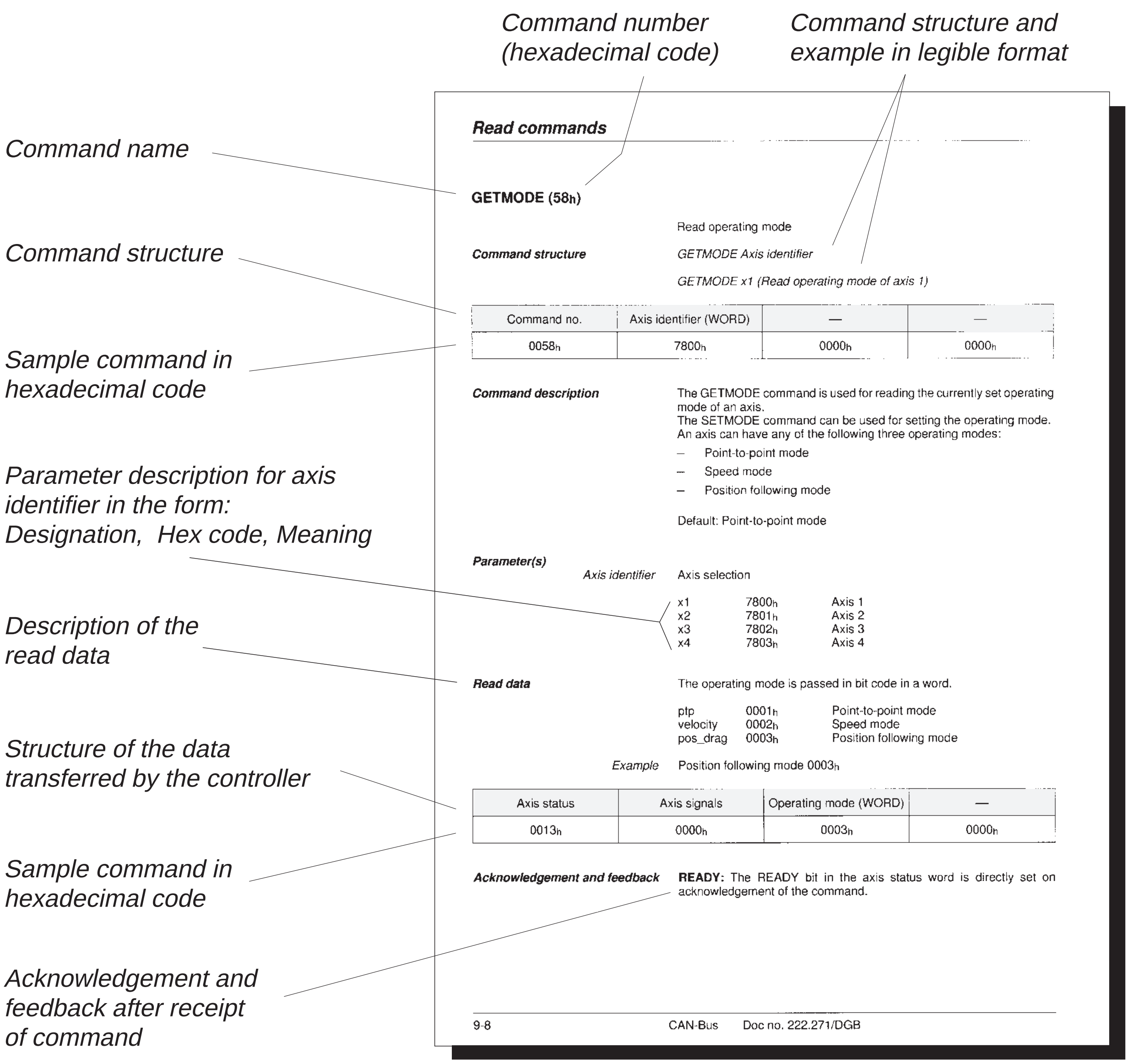
Entries in the two right-hand columns:

X Identifies commands which can be fully utilized with the specified controllers.

– Identifies commands which cannot be used with the specified controllers.

NOTE
Command execution depends on the controller type, the interface configuration (unit variant) and the operating mode setting.

The command descriptions on the following pages are structured as follows:



Read commands

GETMODE (58h)

Read operating mode

Command structure *GETMODE Axis identifier*

GETMODE x1 (Read operating mode of axis 1)

Command no.	Axis identifier (WORD)	—	—
0058h	7800h	0000h	0000h

Command description The GETMODE command is used for reading the currently set operating mode of an axis.
The SETMODE command can be used for setting the operating mode.
An axis can have any of the following three operating modes:

- Point-to-point mode
- Speed mode
- Position following mode

Default: Point-to-point mode

Parameter(s)

Axis identifier Axis selection

x1	7800h	Axis 1
x2	7801h	Axis 2
x3	7802h	Axis 3
x4	7803h	Axis 4

Read data The operating mode is passed in bit code in a word.

ptp	0001h	Point-to-point mode
velocity	0002h	Speed mode
pos_drag	0003h	Position following mode

Example Position following mode 0003h

Axis status	Axis signals	Operating mode (WORD)	—
0013h	0000h	0003h	0000h

Acknowledgement and feedback **READY:** The READY bit in the axis status word is directly set on acknowledgement of the command.

9-8 CAN-Bus Doc no. 222.271/DGB

GETANALOG (61_h) (Series 300 only)

Read analog input

Command structure

GETANALOG Analog module, channel

GETANALOG a2, 1

Command no.	Analog module (WORD)	Channel (WORD)	—
0061_h	6101_h	0001_h	0000_h

Command description

The GETANALOG command can be used for reading in voltages from an analog input. The ANOZ analog module (see controller manual) has 5 analog inputs for this purpose.
The analog input is addressed by the identifier of the analog module and the channel number of the input.

Parameter(s)

Analog module ID

Selection of the analog module

a2	6101_h	Analog module (2nd adapter slot)

Channel

Selects the analog input of the analog module

1	0001_h	Analog input 1
2	0002_h	Analog input 2
3	0003_h	Analog input 3
4	0004_h	Analog input 4
5	0005_h	Analog input 5

Read data

The voltages are made available in the read data and specified as hexadecimal values (DINT data type) in millivolts (mV).

Example

Voltage 5000 mV (1388_h) at analog input 2 on analog module a2 (ANOZ)

Axis status	Axis signals	Voltage (DINT)
0013_h	0000_h	$0000\ 1388_h$

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

GETCURRENT (53_h)

Read electrical current values

Command structure

GETCURRENT Axis identifier, selection

GETCURRENT x1, stand (Read electrical current setting for standstill of axis 1)

Command no.	Axis identifier (WORD)	Selection (WORD)	—
0053_h	7800_h	0008_h	0000_h

Command description

The GETCURRENT command can be used for reading the electrical current values set for an axis with SETCURRENT.
The electrical current values can be set with the SETCURRENT command for the following axis movement states:

- Standstill
- Acceleration and deceleration
- Constant movement

The electrical current value is specified as a percentage of the controller's maximum current; see SETCURRENT command.

Default: See controller manual

Parameter(s)

Axis identifier Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Selection Selection of the electrical current setting to be read

stand	0008_h	Current at axis standstill
accel	0001_h	Current at acceleration and deceleration
constant	0002_h	Current at constant movement

Read data

Electrical current values are passed as a percentage of the set maximum phase current (100%).

Example Electrical current setting of 50% at standstill

Axis status	Axis signals	El. current value (INT)	—
0013_h	0000_h	50 (0032_h)	0000_h

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

GETDATA (60_h)

Repeat read data request

Command structure

GETDATA Axis identifier

GETDATA x1 (Request data of last read command for axis 1)

Command no.	Axis identifier (WORD)	—	—
0060_h	7800_h	0000_h	0000_h

Command description

The GETDATA command can be used for re-reading the read data of the previous read command from the controller.
The axis is selected using the axis identifier parameter.
If an axis did not yet receive any read command, the read data normally read by the GETSTATE command are transmitted.

NOTE
The GETDATA command does not affect the READY bit in the axis status word. It is therefore suitable for monitoring the execution of write commands (positioning commands).

Parameter(s)

Axis identifier Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Read data

The read data of the command are the read data of the previously executed read command.

Example If the previous read command was a GETPOS command, GETDATA transmits, for example, the current actual position (here, 2500). Completion of the GETPOS command can be checked with the READY bit of the axis status word.

Axis status	Axis signals	Read data of previous read command
0013_h	0000_h	$0000\ 09C4_h$ (2500)

Acknowledgement and feedback

READY: The READY bit in the axis status word is not modified by this command.

GETENSIG (54h)

Determine enabled or disabled axis signals

Command structure

GETENSIG Axis identifier

GETENSIG x1 (Read enabled or disabled axis signals of axis 1)

Command no.	Axis identifier (WORD)	—	—
0054h	7800h	0000h	0000h

Command description

The GETENSIG command is used for determining the axis signals enabled or disabled using ENSIG.
Only enabled signals are monitored by the controller and can respond to events.

Parameter(s)

Axis identifier Axis selection

x1	7800h	Axis 1
x2	7801h	Axis 2
x3	7802h	Axis 3
x4	7803h	Axis 4

Read data

The enabled or disabled axis signals are passed in a word. A "1" at the corresponding bit position means signal enabled, a "0" means signal disabled.

limp	0001h	Positive hardware limit switch
limn	0002h	Negative hardware limit switch
ref	0004h	Reference switch input
stop	0008h	Hardware STOP input
trig	0010h	Hardware trigger input
swlimp	0020h	Positive software limit switch
swlimn	0040h	Negative software limit switch
swstop	0080h	Software stop
dragerr	0100h	Contouring error
encerr	0200h	Encoder error
ampnotready	0400h	Power controller not ready
amptemp	0800h	Power controller overtemperature
motortemp	1000h	Motor overtemperature

Example limp, limn and trig enabled: 0013h

Axis status	Axis signals	Enabled signals (WORD)	—
0013h	0000h	0013h	0000h

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

GETERROR (5Fh)

Read error from the error memory

Command structure

GETERROR

GETERROR (Read error from the controller's error memory)

Command no.	—	—	—
005Fh	0000h	0000h	0000h

Command description

The GETERROR command is used for reading an error entry from the error memory of the controller. If there are several entries in the error memory, the first entry recorded is read. To read further error entries, the command must be re-sent.

Parameter(s)

None

Read data

If an error is registered in the error memory, an error code is transmitted (see chapter 6.3). If there is no error entry, the error code transmitted is 0.

Example

No error entered in the error memory (error code is 0)

Axis status	Axis signals	Error code (WORD)	—
0013h	0000h	0000h	0000h

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

GETMODE (58_h)

Read operating mode

Command structure *GETMODE Axis identifier*

GETMODE x1 (Read operating mode of axis 1)

Command no.	Axis identifier (WORD)	—	—
0058_h	7800_h	0000_h	0000_h

Command description

The GETMODE command is used for reading the currently set operating mode of an axis.
The SETMODE command can be used for setting the operating mode.
An axis can have any of the following three operating modes:

- Point-to-point mode
- Speed mode
- Position following mode

Default: Point-to-point mode

Parameter(s)

Axis identifier Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Read data

The operating mode is passed in bit code in a word.

ptp	0001_h	Point-to-point mode
velocity	0002_h	Speed mode
pos_drag	0003_h	Position following mode

Example Position following mode 0003_h

Axis status	Axis signals	Operating mode (WORD)	—
0013_h	0000_h	0003_h	0000_h

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

GETPOS (50h)

Read position values

Command structure

GETPOS Identifier, selection

GETPOS x1, actual (Read current actual position of axis 1)

Command no.	Identifier (WORD)	Selection (WORD)	—
0050_h	7800_h	1000_h	0000_h

Command description

The GETPOS command is used for reading the current position of an axis. It is possible to read either the current actual position of the motor or the setpoint preset with MOVE or POS. The GETPOS command can also be used for reading the current position of an encoder, e.g. with GETPOS p1 actual.

In position following mode, the current following error (difference between set and actual position) can be read.

Parameter(s)

Axis identifier Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Encoder ID Encoder selection

p1*	7000_h	Encoder 1
p2	7001_h	Encoder 2

* Not applicable on WDP3-01X, WPM-311 and WDPM3-314 controllers

Selection Selection of the requested position type

actual	1000_h	Current actual position
target	2000_h	Current setpoint
scrdiff	$000A_h$	Following error with electronic gear

Read data

Position values are passed in user-defined units.

Example Current actual position 1000

Axis status	Axis signals	Position (DINT)
0013_h	0000_h	1000 (0000 $03E8_h$)

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

GETSIG (55_h)

Read current axis signal states

Command structure

GETSIG Axis identifier

GETSIG x1 (Read current axis signals of axis 1)

Command no.	Axis identifier (WORD)	—	—
0055_h	7800_h	0000_h	0000_h

Command description

The GETSIG command is used for reading the current states of axis signals. The signals are directly determined at the inputs.

Parameter(s)

Axis identifier Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Read data

The current axis signals are passed in a word. A "1" at the corresponding bit position means signal input energized, a "0" means signal input deenergized.

limp	0001_h	Positive hardware limit switch
limn	0002_h	Negative hardware limit switch
ref	0004_h	Reference switch input
stop	0008_h	Hardware STOP input
trig	0010_h	Hardware trigger input
swlimp	0020_h	Positive software limit switch
swlimn	0040_h	Negative software limit switch
swstop	0080_h	Software stop
dragerr	0100_h	Contouring error
encerr	0200_h	Encoder error
ampnotready	0400_h	Power controller not ready
amptemp	0800_h	Power controller overtemperature
motortemp	1000_h	Motor overtemperature

Example limp and limn active: 0003_h

Axis status	Axis signals	Current signals (WORD)	—
0013_h	0000_h	0003_h	0000_h

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

GETSIG_ACTIV_H (57h)

Read active state of axis signals

Command structure

GETSIG_ACTIV_H Axis identifier

GETSIG_ACTIV_H x1 (Get active states of axis signals of axis 1)

Command no.	Axis identifier (WORD)	—	—
0057h	7800h	0000h	0000h

Command description

The GETSIG_ACTIV_H command is used for reading the active states (active low, active high) of the limp, limn, ref and trig axis signals. The active states can be set using the SETSIG_ACTIV_H command.

Parameter(s)

Axis identifier Axis selection

x1	7800h	Axis 1
x2	7801h	Axis 2
x3	7802h	Axis 3
x4	7803h	Axis 4

Read data

The active states of the four signals are passed in bit code in a word. A "1" at the corresponding bit position means signal active when energized (active high), a "0" means signal active when deenergized (active low).

limp	0001h	Positive hardware limit switch
limn	0002h	Negative hardware limit switch
ref	0004h	Reference switch input
trig	0010h	Hardware trigger input

Example limp, limn and ref are active low, trig is active high: 0010h

Axis status	Axis signals	Active state (WORD)	—
0013h	0000h	0010h	0000h

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

GETSIG_SR (56h)

Read temporarily stored axis signals

Command structure *GETSIG_SR Axis identifier*

GETSIG_SR x1 (Read temporarily stored axis signals of axis 1)

Command no.	Axis identifier (WORD)	—	—
0056h	7800h	0000h	0000h

Command description

The GETSIG_SR command is used for getting the temporarily stored axis signals. The axis signal states are temporarily stored in a buffer. This makes it possible to respond to a signal even when it has become inactive. A signal is stored until it is reset using CLRSIG_SR.

Parameter(s)

Axis identifier Axis selection

x1	7800h	Axis 1
x2	7801h	Axis 2
x3	7802h	Axis 3
x4	7803h	Axis 4

Read data

The temporarily stored axis signals are passed in bit code in a word. A "1" at the corresponding bit position means signal input active, a "0" means signal input inactive.

limp	0001h	Positive hardware limit switch
limn	0002h	Negative hardware limit switch
ref	0004h	Reference switch input
stop	0008h	Hardware STOP input
trig	0010h	Hardware trigger input
swlimp	0020h	Positive software limit switch
swlimn	0040h	Negative software limit switch
swstop	0080h	Software stop
dragerr	0100h	Contouring error
encerr	0200h	Encoder error
ampnotready	0400h	Power controller not ready
amptemp	0800h	Power controller overtemperature
motortemp	1000h	Motor overtemperature

Example limp and limn were active: 0003h

Axis status	Axis signals	Temporary signals (WORD)	—
0013h	0000h	0003h	0000h

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

GETSTATE (52_h)

Read error status of an axis

Command structure *GETSTATE Axis identifier*

GETSTATE x1 (Read error status of axis 1)

Command no.	Axis identifier (WORD)	—	—
0052_h	7800_h	0000_h	0000_h

Command description The GETSTATE command is used for reading the error status of an axis.

Parameter(s)

Axis identifier Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Read data The error status of an axis is passed in bit code in a word (WORD).

Bit 15	–	–
Bit 14	stop	Ref. movement error caused by hardware STOP
Bit 13	ref	Ref. movement error caused by reference switch
Bit 12	limn	Ref. movement error caused by neg. limit switch
Bit 11	limp	Ref. movement error caused by pos. limit switch
Bit 10*	man_active	Manual mode (0 = inactive, 1 = active)
Bit 9	–	–
Bit 8	pos_over	Position overrun
Bit 7	acc_undef	Acceleration not defined
Bit 6	pos_undef	Actual position not defined
Bit 5	ampdis	Power controller not enabled
Bit 4	refblo	Ref. movement error caused by blocked axis
Bit 3	refnen	Ref. movement error, limit switch not enabled
Bit 2	referr	Reference movement error
Bit 1	refact	Reference movement active
Bit 0	refok	Reference movement o.k.

* For WDP3-01X controllers only

Example Error status 0002_h (reference movement active)

Axis status	Axis signals	Error status (WORD)	—
0013_h	0000_h	0002_h	0000_h

Acknowledgement and feedback **READY:** The READY bit in the axis status word is directly set on acknowledgement of the command.

GETVEL (51_h)

Read speed value

Command structure

GETVEL Axis identifier, selection

GETVEL x1, actual (Read current actual speed of axis 1)

Command no.	Axis identifier (WORD)	Selection (WORD)	—
0051_h	7800_h	1000_h	0000_h

Command description

The GETVEL command is used for reading speed values of an axis.

Any of the following speeds can be selected:
- Current actual speed
- Current set speed
- Start speed
- System speed

Parameter(s)

Axis identifier Axis selection

x1	7800_h	Axis 1
x2	7801_h	Axis 2
x3	7802_h	Axis 3
x4	7803_h	Axis 4

Selection Selection of the requested speed type

actual	1000_h	Current actual speed
target	2000_h	Current set speed
start	0001_h	Start speed
system	0002_h	System speed

Read data

Speed values are passed in user-defined units.

Example Current actual speed 2500

Axis status	Axis signals	Speed (DINT)
0013_h	0000_h	2500 (0000 $09C4_h$)

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

READ_FLAGS_DWORD ($5D_h$) (Series 300 only)

Read flag as a double word from the flag area

Command structure

READ_FLAGS_DWORD Flag number

READ_FLAGS_DWORD 10 (Read flag words 10 and 11)

Command no.	Flag number (WORD)	—	—
$005D_h$	10 ($000A_h$)	0000_h	0000_h

Command description

The READ_FLAGS_DWORD command reads two flag words as a double word from the flag area on the controller.

Parameter(s)

Flag number Number of the flag word in the flag area on the controller, at which the reading process of the two flag words starts.

Read data

The two flag words are passed as a double word (DWORD).

Example

Contents	10	11
Value	0 0 1 A	1 A 1 A $_h$

Axis status	Axis signals	Flag words (DWORD)
0013_h	0000_h	$001A1A1A_h$

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

NOTE
The data can be written with the DINT data type by the application program on a Series 300 controller.

READ_FLAGS_WORD ($5C_h$) (Series 300 only)

Read flag word from flag area

Command structure

READ_FLAGS_WORD Flag number

READ_FLAGS_WORD 10 (Read flag word 10)

Command no.	Flag number (WORD)	—	—
$005C_h$	10 ($000A_h$)	0000_h	0000_h

Command description

The READ_FLAGS_WORD command is used for reading a flag word from the flag area of the controller.

Parameter(s)

Flag number

Number of the flag word to be read from the flag area on the controller. The number is passed with the INT data type.

Read data

The flag word to be read is passed as a word (WORD).

Example

Flag word $001A_h$

Axis status	Axis signals	Flag word (WORD)	—
0013_h	0000_h	$001A_h$	0000_h

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

READ_INPUT (5A$_h$)

Read inputs directly

Command structure

READ_INPUT Word number, bit number, data type

READ_INPUT 0, 5, bool (Read input 5)

Command no.	Word no. (WORD)	Bit number (BYTE)	Data type (BYTE)	—
005A$_h$	0000$_h$	05$_h$	00$_h$	0000$_h$

Command description

The READ_INPUT command is used for reading controller inputs directly. The inputs can be read bit by bit (individually) or word by word (16 bits).

NOTE
Only those inputs included in the process image of the controller will be read. Limit switch inputs etc. must be read with the GETSIG command.

Parameter(s)

Word number

Selection of the input word from which the input signals are read

Word 0	0000$_h$	Inputs I0 to I15
Word 1	0001$_h$	Inputs I16 to I31 (ext. inputs/outputs)
etc.		

Bit number

Individual inputs are read by specifying a bit number. The bit number determines the position of the bit within the specified input word.
The bit number specification is only significant for the **bool** data type (see below).

Data type

Choose bit-by-bit or word-by-word input reading.

Valid data types

bool	00$_h$	bit by bit
word	02$_h$	word by word

Read data

The inputs to be read are passed in a word (WORD).

Example

Input 5 is set 0001$_h$

Axis status	Axis signals	Inputs (WORD)	—
0013$_h$	0000$_h$	0001$_h$	—

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

READ_PROCESS ($5B_h$)
(Series 300 only)

Read inputs via the process image

Command structure

READ_PROCESS Word number, bit number, data type

READ_PROCESS 1, 0, bool (Read input 16)

Command no.	Word no. (WORD)	Bit number (BYTE)	Data type (BYTE)	—
$005B_h$	0001_h	00_h	01_h	0000_h

Command description

The READ_PROCESS command is used for reading inputs from the process image of the controller. The inputs from the process image can be read bit by bit (individually) or word by word (16 bits).

NOTE
Inputs in the process image are only updated if an application program runs with an active PLC program on the controller.

Parameter(s)

Word number

Selection of the input word from which the input signals are read

Word 0	0000_h	Inputs I0 to I15
Word 1	0001_h	Inputs I16 to I31
etc.		

Bit number

Individual inputs are read by specifying a bit number. The bit number determines the position of the bit within the specified input word.
The bit number specification is only significant for the **bool** data type (see below).

Data type

Choose bit-by-bit or word-by-word input reading.

Valid data types

bool	00_h	bit by bit
word	02_h	word by word

Read data

The inputs to be read are passed in a word (WORD).

Example

Input 16 is set 0001_h

Axis status	Axis signals	Inputs (WORD)	—
0013_h	0000_h	0001_h	—

Acknowledgement and feedback

READY: The READY bit in the axis status word is directly set on acknowledgement of the command.

10 Appendix

10.1 Command lists

10.1.1 Write commands

ACT_AXIS	**Axis identifier**	–	–
25_h	7800_h to 7803_h I1 $6C00_h$	–	–

BRAKE	**Axis identifier**	**Word number**	**Bit number**
$2A_h$	7800_h to 7803_h	0000_h to 0010_h	0000_h to $000F_h$

CLRERROR	**Axis identifier**	–	–
$1B_h$	7800_h to 7803_h I1 $6C00_h$ plc 1000_h	–	–

CLRSIG_SR	**Axis identifier**	**Signal template**	–
$1D_h$	7800_h to 7803_h	0000_h to $FFFF_h$	–

CONT	**Axis identifier**	–	–
$1C_h$	7800_h to 7803_h	–	–

DISP	**Output value**	–	–
24_h	0 to 99	–	–

ENSIG	**Axis identifier**	**Signal template**	–
$1E_h$	7800_h to 7803_h	0000_h to $FFFF_h$	–

INITDRIVE	**Axis identifier**	**Selection**	–
04_h	7800_h to 7803_h	0000_h or 1000_h	–

LINMOVE	**Interpolator**	–
30_h	I1 6C00h	–

LINPOS	**Interpolator**	–
31_h	I1 6C00h	–

MOVE	Axis identifier	Position
$0A_h$	7800_h to 7803_h	WDP3-01X: ±223696213 Series 300: ±2147483647

POS	Axis identifier	Position
09_h	7800_h to 7803_h	WDP3-01X: ±223696213 Series 300: ±2147483647

RAMP_EXP	Axis identifier	Maximum acceleration
19_h	7800_h to 7803_h	0 to 999 Hz/ms

RAMP_LIN	Axis identifier	Maximum acceleration
18_h	7800_h to 7803_h	0 to 999 Hz/ms

RAMP_SIN	Axis identifier	Maximum acceleration
$1A_h$	7800_h to 7803_h	0 to 999 Hz/ms

REF_OUT_DISTANCE	Axis identifier	Max. allowed dist. limit switch
33_h	7800_h to 7803_h	WDP3-01X: 10 to 223696213 Series 300: 10 to 2147483647

REFPOS_LIMN	Axis identifier	Reference speed
07_h	7800_h to 7803_h	0 to 5000 Hz

REFPOS_LIMP	Axis identifier	Reference speed
06_h	7800_h to 7803_h	0 to 5000 Hz

REFPOS_REF	Axis identifier	Reference speed
08_h	7800_h to 7803_h	0 to 5000 Hz

RESET_PLC	–	–	–
02_h	–	–	–

ROTMON_DISABLE	Axis identifier	Encoder	–
$2D_h$	7800_h to 7803_h	p1 7000_h p2 7001_h pext 7008_h	–

ROTMON_ENABLE	Axis identifier	Encoder	Resolution
$2B_h$	7800_h to 7803_h	p1 7000_h p2 7001_h pext 7008_h	$1F4_h$ 500 marks $3E8_h$ 1000 marks

ROTMON_RESET	Axis identifier	Encoder	–
$2C_h$	7800_h to 7803_h	p1 7000_h p2 7001_h pext 7008_h	–

SETANALOG	Analog module	Channel	Voltage
32_h	6101_h	0001_h	0 to 10000 mV

SETCURRENT	Axis identifier	Electrical current value	Selection
$0D_h$	7800_h to 7803_h	0 to 100%	stand 0008_h accel 0001_h constant 0002_h

SETENCODER	Encoder ID	Selection	–
28_h	p1 7000_h p2 7001_h	encquad 0002_h encdouble 0003_h encsingle 0004_h encpulsdir 0005_h	–

SETHARDWARE	Axis identifier	Selection	–
26_h	7800_h to 7803_h	pulson 0001_h pulsoff 0002_h ampon 0003_h ampoff 0004_h dirinvert 0005_h dirnormal 0006_h	–

SETIPOS	Axis identifier	Position
$2F_h$	7800_h to 7803_h	±2147483647

SETMODE	Axis identifier	Operating mode	Source
10_h	7800_h to 7803_h	ptp 0001_h velocity 0002_h pos_drag 0003_h	p1 7000_h p2 7001_h

SETNORM_GEAR_DEN	Axis identifier	Denominator
16_h	7800_h to 7803_h	WDP3-01X: ±429496729 (except 0) Series 300: ±2147483647 (except 0)

SETNORM_GEAR_NUM	Axis identifier	Numerator
15_h	7800_h to 7803_h	WDP3-01X: ±44739242 Series 300: ±2147483647

SETNORM_POS_DEN	Axis identifier	Denominator
12_h	7800_h to 7803_h	±2147483647

SETNORM_POS_NUM	Axis identifier	Numerator
11_h	7800_h to 7803_h	±2147483647

SETNORM_VEL_DEN	Axis identifier	Denominator
14_h	7800_h to 7803_h	±2147483647

SETNORM_VEL_NUM	Axis identifier	Numerator
13_h	7800_h to 7803_h	±2147483647

SETOFFSET	Axis identifier	Offset
27_h	7800_h to 7803_h	WDP3-01X: ±223696213 Series 300: ±2147483647

SETPOS	Axis identifier	Position
05_h	7800_h to 7803_h	WDP3-01X: ±223696213 Series 300: ±2147483647

SETSIG_ACTIV_H	Axis identifier	Signal template	–
17_h	7800_h to 7803_h	limp 0001_h limn 0002_h ref 0004_h trig 0010_h	–

SETVEL_START	Axis identifier	Start/stop speed
$0F_h$	7800_h to 7803_h	Up to maximum system speed

SETVEL_SYS	Axis identifier	System speed
$0E_h$	7800_h to 7803_h	WDP3-01X: 3414 Hz to 40000 Hz Series 300: 1024 Hz to 524287 Hz

START_PLC	–	–	–
03_h	–	–	–

STOP_AXIS	Axis identifier	–	–
$0C_h$	7800_h to 7803_h I1 $6C00_h$	–	–

TIMEOUT	Time	–	–
29_h	0000_h to $FFFF_h$	–	–

VEL	Axis identifier	Set speed
$0B_h$	7800_h to 7803_h	Up to maximum system speed

WRITE_FLAGS_DWORD	Flag number	Flag data
22_h	0000_h to 0800_h	0000 0000_h to FFFF $FFFF_h$

WRITE_FLAGS_WORD	Flag number	Flag data	–
21_h	0000_h to 0800_h	0000_h to $FFFF_h$	–

WRITE_OUTPUT	Word number	Bit number	Data type	Signal status
$1F_h$	00_h to 10_h	00_h to $0F_h$	bool 00_h word 02_h	00_h to FF_h

WRITE_PROCESS	Word number	Bit number	Data type	Signal status
20_h	00_h to 10_h	00_h to $0F_h$	bool 00_h word 02_h	00_h to FF_h

10.1.2 Read commands

GETANALOG	**Analog module**	**Channel**	–
61_h	6101_h	0001_h	–

GETCURRENT	**Axis identifier**	**Selection**	–
53_h	7800_h to 7803_h	stand 0008_h accel 0001_h constant 0002_h	–

GETDATA	**Axis identifier**	–	–
60_h	7800_h to 7803_h	–	–

GETENSIG	**Axis identifier**	–	–
54_h	7800_h to 7803_h	–	–

GETERROR	–	–	–
$5F_h$	–	–	–

GETMODE	**Axis identifier**	–	–
58_h	7800_h to 7803_h	–	–

GETPOS	**Identifier**	**Selection**	–
50_h	7800_h to 7803_h 7000_h to 7001_h	actual 1000_h target 2000_h scrdiff $000A_h$	–

GETSIG	**Axis identifier**	–	–
55_h	7800_h to 7803_h	–	–

GETSIG_ACTIV_H	**Axis identifier**	–	–
57_h	7800_h to 7803_h	–	–

GETSIG_SR	**Axis identifier**	–	–
56_h	7800_h to 7803_h	–	–

GETSTATE	**Axis identifier**	–	–
52_h	7800_h to 7803_h	–	–

GETVEL	Axis identifier	Selection	–
51_h	7800_h to 7803_h	actual 1000_h target 2000_h start 0001_h system 0002_h	–

READ_FLAGS_DWORD	Flag number	–	–
$5D_h$	0000_h to 0800_h	–	–

READ_FLAGS_WORD	Flag number	–	–
$5C_h$	0000_h to 0800_h	–	–

READ_INPUT	Word number	Bit number	Data type	–
$5A_h$	00_h to 10_h	00_h to $0F_h$	bool 00_h word 02_h	–

READ_PROCESS	Word number	Bit number	Data type	–
$5B_h$	00_h to 10_h	00_h to $0F_h$	bool 00_h word 02_h	–

10.2 Data transmission structures

This section summarizes all CAN-Bus data transmission structures.

10.2.1 Data transmission format

The data transmission format is an 8-byte data structure. This structure is used for transferring commands and data between a CAN-Bus station and a BERGER LAHR controller.

Byte 7	Byte 8	high address
Byte 5	Byte 6	
Byte 3	Byte 4	
Byte 1	Byte 2	low address

Low address High address

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
Word 1		Word 2		Word 3		Word 4	
Double word 1				Double word 2			

10.2.2 Data transmission from station to controller

The station uses a command structure for sending commands and data to the controller.

Command no. (2 bytes)	Command data (6 bytes)

Command no.: Command number
Command data: Data and parameters of the command

10.2.3 Data transmission from controller to station

Standard data, read data and error codes are transmitted from the controller to the CAN-Bus station in the data structures shown below.

Normal data structure:

Axis status (WORD)	Axis signals (WORD)	Read data (4 bytes)

Data structure in case of an error:

Axis status (WORD)	Axis signals (WORD)	Error code (WORD)	unused (2 bytes)

10.2.4 Axis status

The axis status is transmitted as a word (16 bits) and the axis states are contained in this word in bit code.

Axis status word:

Bit no.	Designation	Description
15	–	–
14	KF	Command error
13	–	–
12	–	–
11	XE4	Error occurred on axis 4
10	XE3	Error occurred on axis 3
9	XE2	Error occurred on axis 2
8	XE1	Error occurred on axis 1
7	ACC	Selected axis accelerates
6	CONST	Selected axis moves constantly
5	BRAKE	Selected axis decelerates
4	STAND	Selected axis stopped
3	–	–
2	REF_OK	Selected axis performed a reference movement
1	INIT_OK	Selected axis has been initialized
0	READY	Selected controller executed a command

10.2.5 Axis signals

The axis signals are transmitted as a word (16 bits) and the axis signal states are contained in this word in bit code.

Axis signal word:

Bit no.	Designation	Description
15	–	–
14	init_err	Initialization error on power controller
13	ref_err	Reference movement error
12	motortemp	Motor overtemperature
11	amptemp	Power controller overtemperature
10	ampnotready	Power controller not ready
9	encerr	Encoder error
8	dragerr	Contouring error
7	swstop	Software stop
6	swlimn	Negative software limit switch
5	swlimp	Positive software limit switch
4	trig	Hardware trigger input
3	stop	Hardware STOP input
2	ref	Reference switch input
1	limn	Negative hardware limit switch
0	limp	Positive hardware limit switch

10.2.6 Error table

No. (hex)	Error cause
1048_h	Axis or encoder not available
1049_h	Invalid command for the axis
$104A_h$	Axis or encoder not ready
$104B_h$	Incorrect parameter value
$104C_h$	Precondition not fulfilled
$104D_h$	Value cannot be calculated
$104E_h$	Insufficient information on source
$104F_h$	Error in selection parameter
1052_h	Command only valid at standstill
1053_h	Acceleration not yet defined
1054_h	Error in acceleration curve
1055_h	Actual position not yet defined
1056_h	Invalid command, since encoder active
1058_h	Drive interrupted or blocked
1059_h	Encoder not ready

No. (hex)	Error cause
105A$_h$	Drive interrupted or blocked
105C$_h$	Reference movement active
105F$_h$	Reference movement error caused by positive limit switch
1060$_h$	Reference movement error caused by negative limit switch
1062$_h$	Cycle monitoring timeout
1069$_h$	Invalid output value for display
1072$_h$	No application program loaded
107D$_h$	Battery voltage low
107E$_h$	Short-circuit on 24 V output
107F$_h$	Invalid output address
1080$_h$	Invalid input address
109B$_h$	Rotation monitoring error
109C$_h$	Encoder error (line broken)
109D$_h$	Power controller readiness error
109E$_h$	Power controller overtemperature
109F$_h$	Motor overtemperature
10A9$_h$	Command only valid in point-to-point mode
10AA$_h$	Error related to software limit switch
10AC$_h$	Current program not saved in EEPROM
10AF$_h$	Positive and negative limit switch inactive
10C1$_h$	Interpolation active
10CF$_h$	Limit switch not enabled
10D2$_h$	Axis preconditions not fulfilled
10D3$_h$	Internal processing error during linear interpolation
10D4$_h$	Linear interpolation aborted because of axis error
10DC$_h$	Field bus link timeout
10DD$_h$	A new field bus command was sent before the previous one had been acknowledged
10DE$_h$	Field bus: Invalid denominator for SETNORM command
10DF$_h$	Field bus: Invalid data type
10E0$_h$	Field bus: Invalid flag word address
10E1$_h$	Field bus: Invalid command
10E2$_h$	Field bus system error
10F5$_h$	Field bus: Invalid command or command currently not permitted
1109$_h$	Rotation monitoring inactive

10.3 Glossary

A/B signals
Pulse signals of an encoder. For one motor revolution, a defined number of pulse signals (e.g. 1000) is generated by the encoder.

Absolute positioning
For absolute positioning, the position value refers to the zero point of the axis (see *Point-to-point mode*).

Acceleration
An axis is accelerated using a preset acceleration curve (accel-eration ramp). The acceleration curve to be set depends on the load conditions of the axis. The ramp type, maximum acceleration and maximum system speed are used for calculating the curve. The RAMP_LIN, RAMP_EXP or RAMP_SIN commands are used for setting acceleration ramps.

Acknowledgement method
A controller acknowledges a command by sending the requested data. The data indicate whether an error occurred with a command, if its execution is still in progress or if execution has been completed.

Actual position
The current position of the axis. The GETPOS command can be used to read the actual position from the controller.

ANOZ
ANOZ is the name of the analog module (hardware) which can be installed in Series 300 controllers.

Application program
Application-specific program in a controller for performing automation-technology tasks.

Axis error
An axis error can occur during axis control. Axis errors are reported to the station using the XE1, XE2, XE3 and XE4 bits in the axis status word. The cause of an axis error can be determined from the axis signal word.

Axis identifier
The axis identifier refers to the axis of a controller which is to execute a command. In the case of single-axis controllers, axis identifier x1 must always be used. In the case of multi-axis controllers, axis identifiers x1, x2, x3 and x4 can be used.

Axis operating mode
An axis can move in any of three operating modes:
Point-to-point mode, speed mode or position following mode.
The SETMODE command is used for setting the axis operating modes.

Axis selection

The standard data (axis status, axis signals) which the controller sends to the station always refer to the axis which is currently selected. The selected axis is the axis which last performed an axis-related command without error. It does not matter whether this command was a write command or a read command. The ACT_AXIS command can be used on multi-axis controllers to select an axis without initiating a controller function. On single-axis controllers, axis 1 is always the selected axis.

Axis signal

Events which occur unexpectedly (asynchronous events) on an axis are reported using axis signals. Examples of such events are limit switch errors or power controller overtemperature.

Axis status word

Error and status information about an axis are stored in an axis status word. The axis status word (16 bits) contains the status information in bit code. The axis status word is part of the standard data which the controller transfers to the station without explicit instruction.

BOOL

Data type (1 bit) for storing data in bit code.

BYTE

Data type (8 bit) for storing data in bit code.

CAN-Bus capability

A device or a controller has CAN-Bus capability if it can be used as a station or controller in a CAN-Bus network.

CAN-Bus network

Standardized field bus for data exchange in automation technology. The CAN-Bus standard can be used for interconnecting several devices from different manufacturers and with different functionality through one uniform interface.
A CAN-Bus network can consist of several stations.

CAN-Bus station

A device in a CAN-Bus network.

Command

The functions of a controller are accessed using commands. Commands are sent from the station to a controller. The controller interprets and executes the commands.
A distinction is made between write commands and read commands.

Command data

Command data are the parameters which belong to a command. 6 bytes are reserved for command data in the command structure.

Command error

A controller reports a command error to the station if it receives an unknown command, if it receives a new command before the previous command has been acknowledged, or if it cannot execute a command.

A controller reports a command error to the station using the command error bit (KF) in the axis status word and an error code.

Command number

Each command is assigned a number (hexadecimal value). This number must be entered in the first word of the command structure.

Command structure

Commands must be transferred from the station to the controller in a defined command structure (8 bytes).

The command structure consists of:

– Command number
  2 bytes, first word of the command structure
– Command data
  6 bytes, words 2, 3 and 4 of the command structure

Contouring error

A contouring error occurs when the difference between the setpoint and the actual position is too large.

Data transmission format

Commands and data are transferred between the station and a controller as an 8-byte data structure.

DINT

Data type (32 bits) for storing large integer numbers (double integer).

Double word

A double word (32 bits) is made up of 2 memory words (16 bits).

Drive unit

Drive units are processing parameters internal to the controller, which are used for positions, speed and acceleration values.

Electronic gear

The controller can implement an electronic gear in position following mode. This involves counting external pulses (A/B signals of an encoder or pulse/direction signals). The pulses are multiplied by a gear ratio and used as a reference variable for positioning a motor.

Encoder

A sensor for detecting the actual position of a motor or for specifying a setpoint for an electronic gear.

Execution condition of a command
The execution of, for example, a positioning command takes a certain period of time. The execution condition of a command can be monitored using the execution bit (READY bit) in the axis status word.

Exponential ramp
An acceleration ramp with an exponential curve.

Field bus
In automation engineering, a data transmission system for sensors and actuators is referred to as a field bus. CAN-Bus, Profibus-DP and Interbus-S are examples of standardized field buses.

Flag
Flags are storage elements in the controller used for system data and user data.

Flag area
A memory area for flags. The size of the flag area depends on the controller, or the controller configuration.

Flag word
A memory word (16 bits) in the flag area.

Following error
The difference between set and actual position of a motor.

Following error limit
A limit value for the following error. If this limit value is exceeded during rotation monitoring, a contouring error is reported.

Function block language
Symbolic language for graphic representation of PLC blocks.

Gear ratio
With an electronic gear, an external reference variable (pulses) is multiplied by a gear ratio and used as the reference variable for moving an axis.
The following applies:

Drive units = Reference variable x Gear ratio

Input signal
A controller has a fixed number of digital inputs.
These inputs can be used for recording digital signals from a technical process. Each 16 inputs are combined into 16-bit input words. Input signals are accessed by the word number of the corresponding input word.

INT
Data type (16 bits) for storing integer numbers.

Intermediately stored axis signal
Axis signals are intermediately stored in a buffer in the controller. Intermediately stored axis signals are not cleared until the signal is inactive and has been reset with the CLRSIG_SR command.

Interpolation
(see *Linear interpolation*)

Linear ramp
An acceleration ramp with a linear progression.

Linear interpolation
With the Series 300 multi-axis positioning units (e.g. WPM-311), several axes can move using linear interpolation. Linear interpolation means in this case that the axes can be activated simultaneously and move interdependently and that all axes reach their final positions at the same time. Linear interpolation can be performed with two or three axes.

Master axis
The master axis is the axis which travels the longest distance in a linear interpolation process. The master axis determines the speeds and accelerations of the individual axes involved in linear interpolation.

Maximum acceleration
The maximum possible acceleration during axis movement. The maximum acceleration value is required for calculating the acceleration curve (RAMP_LIN, RAMP_EXP and RAMP_SIN).

Maximum system speed
The maximum system speed sets a limit value for the maximum permissible speed of an axis movement. This speed is also used when calculating acceleration curves.

Movement range
The movement range of an axis is defined by the smallest and the largest position which can be occupied on an axis. The movement range of an axis is not limited in speed mode and in position following mode.

Multi-axis controller
A controller which can control several axes (e.g. WPM-311).

Negative limit switch
A limit switch in the negative sense of rotation (motor rotating in a counterclockwise direction as seen from front towards the motor shaft).

Network address
Each device in a CAN-Bus network has a unique address which is used for addressing the device during network operation.

Network configuration program

There is a configuration program for each CAN-Bus station which is used for configuring network and station. Each controller must be setup, or identified to the station using the configuration program.

Normalizing factor

Normalizing factors are used to convert

– position, speed and acceleration data
  from user-defined units to drive units internal to the controller and
– to set the gear ratio for an electronic gear.

Normalizing factors make it possible to express these data in a familiar unit of measurement (cm, ms, m/s, etc.) instead of in drive units (e.g. motor steps) which are specific to the controller.

Offset

(see *Reference variable offset*)

Output signal

Output signals (outputs) of a controller are digital signals used to control a technical process. Output signals can be set by the CAN-Bus station using the WRITE_OUTPUT command. Each 16 output signals of a controller are combined into 16-bit output words. Output signals are accessed by the word number of the corresponding output word.

Parameter

Parameters are all fixed and variable values in a controller which affect an axis, e.g. motor phase current and start/stop speed.

Point-to-point mode

In point-to-point mode, the axis moves from a position A to a position B. Positioning can be absolute or relative. For absolute positioning, the position value (absolute position) refers to the zero point of the axis. For relative (incremental) positioning, the position value (relative position) refers to the current position of the axis.

Position following mode

This axis operating mode can be used for implementing an electronic gear.

Positioning command

Positioning commands are commands which initiate an axis movement (POS, MOVE, VEL, CONT).

Positive limit switch

A limit switch in the positive sense of rotation (motor rotating in a clockwise direction as seen from front towards the motor shaft).

Power controller

The motor is controlled by a power controller. The power controller converts positioning signals from the processor control into signals for motor control.

Power controller enable

The power controller must be enabled before a controller can process commands. Power controller enabling is effected by the INITDRIVE command.

Pulse/direction signal

Signals for reference variable input for an electronic gear.

Ramp

A distinction is made between exponential, linear and sine square acceleration ramps.

Read command

A read command instructs the controller to transmit specific data to the station. These data are called read data. For example, the station can use the GETPOS command to read the current position of an axis.

Read data

Data which are read from the controller in response to read commands from the station.

READY bit

The READY bit (axis status word) can be used for monitoring the execution condition of a command.

Reference movement

During a reference movement, the axis advances to a reference point (hardware limit switch or reference switch) which is to serve as the reference point (zero point) for the subsequent absolute positioning operations. Reference movements are only possible in point-to-point mode.

Reference point

A reference point must be defined for absolute positioning in point-to-point mode. All absolute positioning operations in point-to-point mode are related to this reference point. The zero point is frequently used as a reference point.
The reference point can be determined either by a reference movement or using the SETPOS command.

Reference speed

The reference speed is the speed at which the axis moves away from a limit switch.

Reference switch

A hardware switch which can be approached from either direction for a reference movement.

Reference variable

External pulses are counted in position following mode (electronic gear) and used as a reference variable for the position of an axis. The axis follows the supplied reference variable exactly.

Reference variable offset

The reference variable offset is a relative position which is added to the reference variable in an electronic gear.
The following applies:

Drive units = Offset + (Reference variable x Gear ratio)

Relative positioning

For relative positioning, the position value refers to the current position of the axis (see *Point-to-point mode*).

Rotation monitoring

Rotation monitoring is used for detecting positional deviations of motor movements. The actual position is detected by an encoder and then compared with the setpoint. If the difference between actual and set position exceeds a preset value, a contouring error is reported and the motor is decelerated.

RS 485 interface

Standardized serial interface for safe data transmission.

Set speed

The speed at which the axis is to move. The VEL command is used to specify the set speed.

Setpoint (set position)

The target position which should be reached during a positioning operation in point-to-point mode. The POS and MOVE commands take the setpoint as a parameter.

Sine square ramp

An acceleration ramp with a sine square curve.

Software limit switch

An adjustable position which behaves like a limit switch. When the axis reaches this position, a limit switch error is generated and the axis is stopped.

Speed mode

In speed mode, the axis moves at a specified set speed.
The VEL command can be used to set new speeds during axis movement. The axis then continues moving at the new speed.

Standard data

Standard data consist of the axis status word and the axis signal word. They are transferred from the controller to the station together with the read data or an error code in an 8-byte data structure.

Start/stop speed
The speed at which the axis starts from standstill and the speed at which it is decelerated until standstill.

Status display
The seven-segment display on the front panel of a controller.

User-defined unit
User-defined units are processing parameters which can be freely defined by the user for positions, speed and acceleration values. They are used for enabling the user to specify data in an application-related unit of measurement (metre, inch, degree, hertz, etc.). User-defined units are converted into drive units using normalizing factors.
The following formulae are used for conversion:
For position data:

Drive units = User-defined units x Normalizing factor

For speed data:

$$\text{Drive units} = \text{User–defined units} \times \frac{\text{Normalizing factor}}{256}$$

WORD
Data type (16 bits) for storing data in bit code.

Word
A memory word (16 bits).

Write command
A write command initiates a function in a controller, e.g. a positioning process. A write command can also take parameters for transferring data to the controller.

Zero point
(see *Reference point*)

10.4 Abbreviations

a2	Identifier for analog module
CAL	CAN Application Layer
cm	Centimetres
COB	Communication objects
DC	Direct current
GND	Ground
Hz	Hertz
Hz/ms	Hertz per millisecond
I0 to I15	Inputs 0 to 15
KF	Command error flag
kHz	Kilohertz
kHz/s	Kilohertz per second
l1	Identifier for linear interpolator
m/s	Metres per second

p1, p2	Encoder interface
PC	Personal Computer
Q0 to Q15	Outputs 0 to 15
RS 485	Serial interface
s	Seconds
PLC	Programmable Logic Controller
t	Time
v	Speed (velocity)
x1 to x4	Axis identifiers (axes 1 to 4)

11 Index

12 Corrections and additions

At present there are no corrections or additions.