



SEW
EURODRIVE

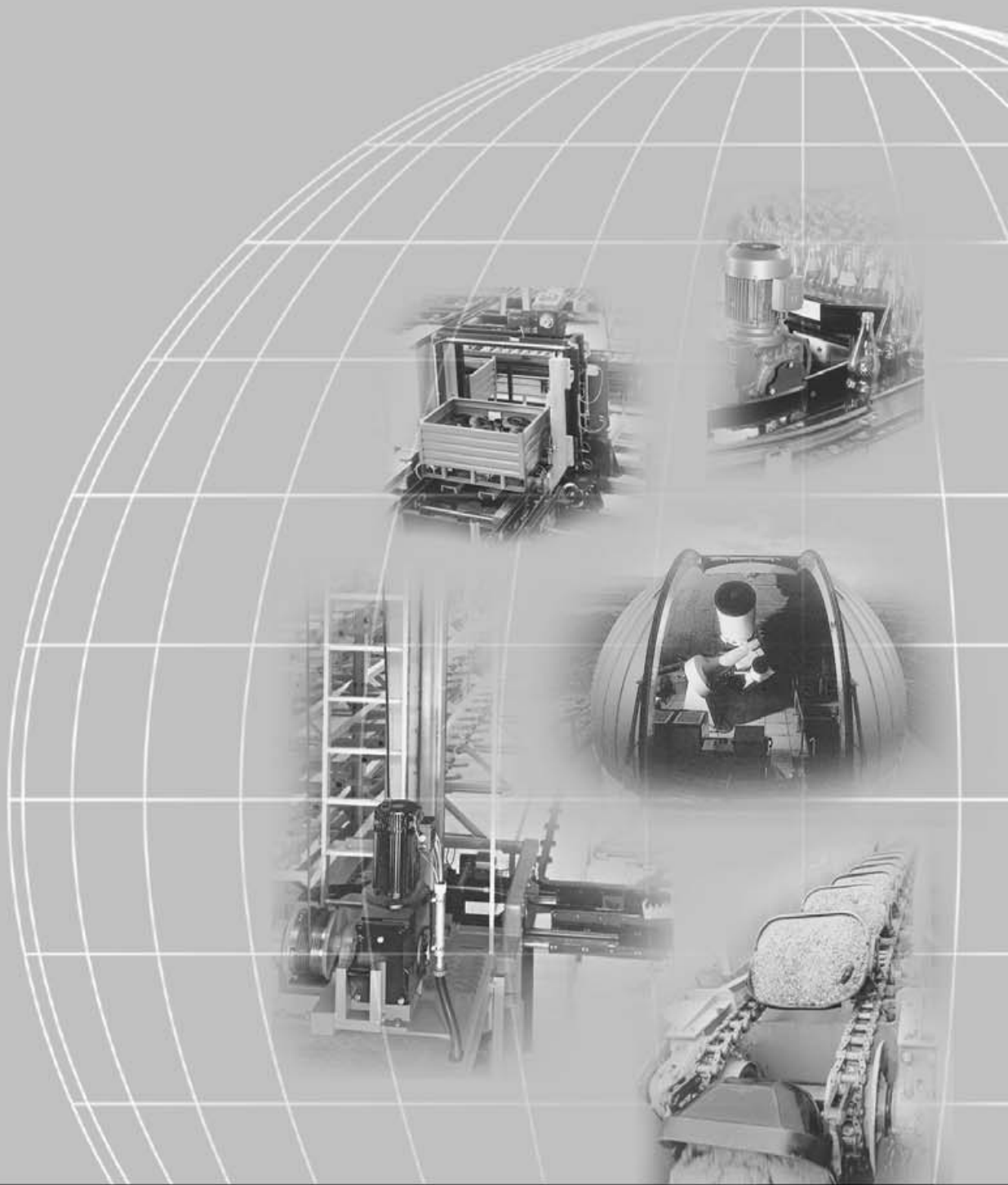
MOVIDRIVE[®] Serial Communication

Edition

11/2001











Manual
1053 1610 / EN



SEW-EURODRIVE





	1 Important Notes.....	4
	2 Introduction	5
	2.1 Overview of serial interfaces.....	5
	2.2 Technical data.....	8
	2.3 MOVILINK® and system bus.....	9
	3 Installation	12
	3.1 System bus (SBus) installation	12
	3.2 RS-485 interface installation	14
	3.3 RS-232 interface installation	16
	4 RS-485 Communication	17
	4.1 Telegrams	17
	4.2 Addressing and transmission process	20
	4.3 Data contents and PDU types.....	29
	5 System Bus (SBus)	37
	5.1 Slave data exchange via MOVILINK®.....	37
	5.2 Setting parameters via the CAN bus.....	42
	5.3 Master data exchange via MOVILINK®.....	47
	5.4 Master/slave operation via the SBus.....	50
	5.5 Data exchange via variable telegrams.....	51
	5.6 Project planning example for SBus.....	62
	6 Operation and Service	64
	6.1 Startup problems with the SBus.....	64
	6.2 Return codes for parameter setting.....	65
	7 Parameter List	67
	7.1 Explanation of the table header	67
	7.2 Complete parameter list, sorted by parameter numbers.....	68
	7.3 Quantity and conversion index.....	84
	8 Index.....	87



1 Important Notes



- **This manual does not replace the detailed operating instructions!**
- **Installation and startup only by trained personnel observing applicable accident prevention regulations and the MOVIDRIVE® operating instructions!**

Documentation

- Read through this manual carefully before you commence installation and startup of MOVIDRIVE® drive inverters with a serial communications link (RS-232, RS-485, system bus).
- This manual assumes that the user has access to and is familiar with the MOVIDRIVE® documentation, in particular the MOVIDRIVE® system manual.
- In this manual, cross references are marked with "→". For example, (→ Sec. X.X) means: Further information can be found in section X.X of this manual.
- A requirement of fault-free operation and fulfillment of any rights to claim under guarantee is that the documentation is observed.






Bus systems

General safety notes on bus systems:

This communication system allows you to match the MOVIDRIVE® drive inverter to the specifics of your application to a very high degree. As with all bus systems, there is a danger of invisible, external (as far as the inverter is concerned) modifications to the parameters which give rise to changes in the inverter behavior. This may result in unexpected (not uncontrolled, though!) system behavior.

Safety and warning instructions

Always follow the safety and warning instructions contained in this publication!

	Electrical hazard Possible consequences: Severe or fatal injuries.
	Hazard Possible consequences: Severe or fatal injuries.
	Hazardous situation Possible consequences: Slight or minor injuries.
	Harmful situation Possible consequences: Damage to the unit and the environment.
	Tips and useful information.



2 Introduction

2.1 Overview of serial interfaces

The following serial interfaces are provided as standard with MOVIDRIVE® drive inverters for serial communication:

1. System bus (SBus) = CAN bus to CAN specification 2.0, parts A and B.
2. RS-485 interface to EIA standard

MOVIDRIVE® MD_60A

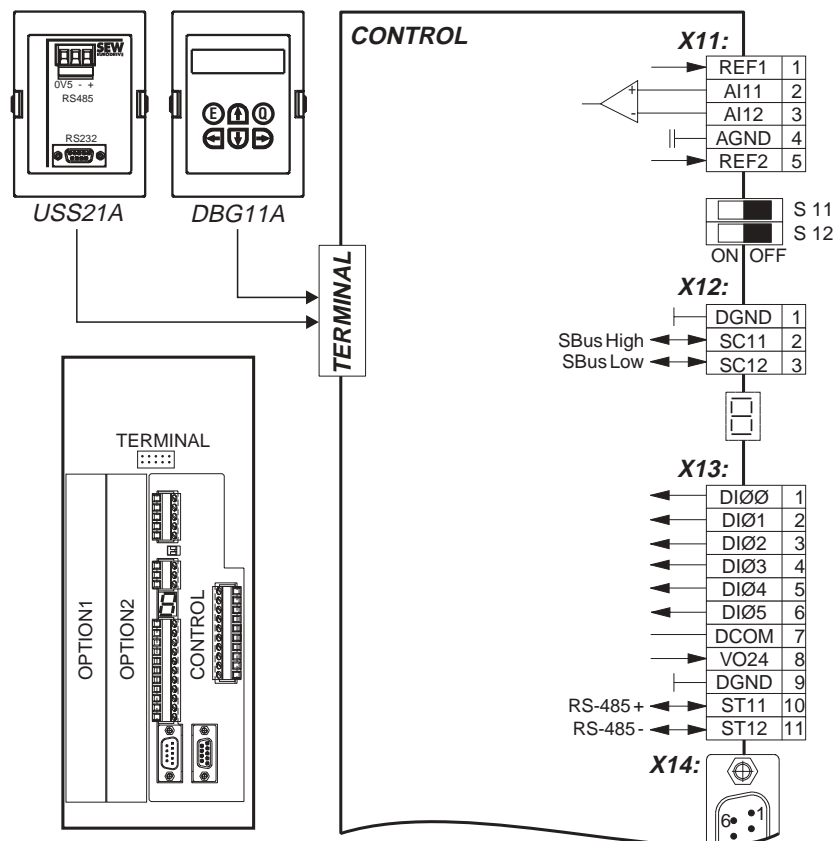
System bus (SBus):

The system bus (SBus) is routed to terminals X12:2/3 in MOVIDRIVE® MD_60A drive inverters.

RS-485 interface:

The RS-485 interface is routed to the TERMINAL option slot and, in parallel, to terminals X13:10/11 in MOVIDRIVE® MD_60A drive inverters.

Either the "DBG11A keypad" or the "USS21A serial interface" can be connected to the TERMINAL option slot.



05274AXX

Fig. 1: Serial interfaces on MOVIDRIVE® MD_60A

X12:1 X12:2 X12:3	DGND: Ref. potential SBus high SBus low	CAN bus to CAN specification 2.0, parts A and B, transmission technology to ISO 11898, max. 64 stations, terminating resistor (120 Ω) can be activated using DIP switches
X13:10 X13:11	ST11: RS-485+ ST12: RS-485-	EIA standard, 9600 baud, max. 32 stations Max. cable length 200 m (660 ft) in total Dynamic terminating resistor with fixed installation



MOVIDRIVE® compact

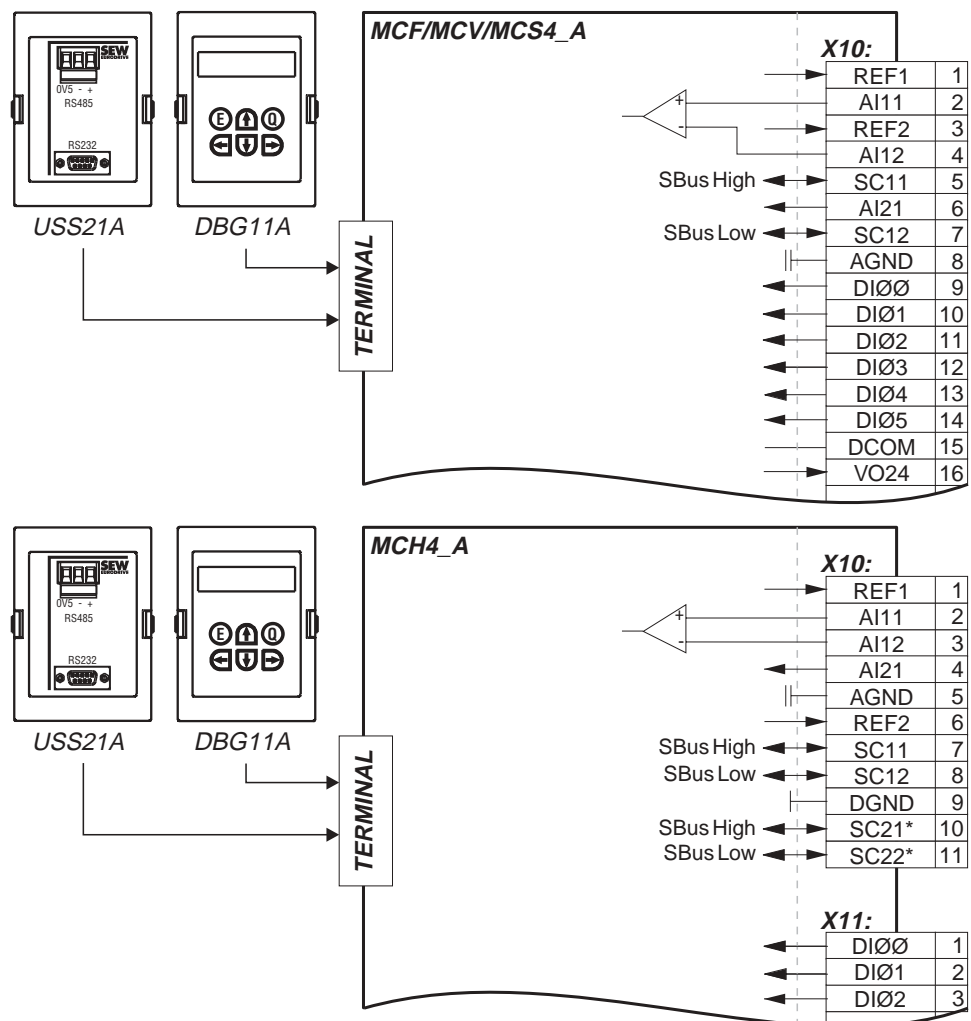
System bus (SBus):

- The system bus (SBus) is routed to terminals X10:5/7 in MOVIDRIVE® compact MCF/MCV/MCS4_A drive inverters.
- The system bus (SBus) is routed to terminals X10:7/8 and X10:10/11 in MOVIDRIVE® compact MCH4_A drive inverters. Terminals X10:7 and X10:10 are electrically connected, as are terminals X10:8 and X10:11.

RS-485 interface:

The RS-485 interface is routed to the TERMINAL option slot in MOVIDRIVE® compact drive inverters.

Either the "DBG11A keypad" or the "USS21A serial interface" can be connected to the TERMINAL option slot.



05275AXX

Fig. 2: Serial interfaces on MOVIDRIVE® compact

* Only use these terminals if S12 = OFF; connect terminating equipment to SC11/SC12.

MOVIDRIVE® compact MCF/MCV/MCS4_A		CAN bus to CAN specification 2.0, parts A and B Transmission system to ISO 11898 max. 64 stations Terminating resistor (120 Ω) can be activated using DIP switches
X10:5	SBus high	
X10:7	SBus low	
MOVIDRIVE® compact MCF/MCV/MCS4_A		
X10:7/10	SBus high	
X10:8/11	SBus low	



USS21A (RS-232 and RS-485)

Startup, operation and service are possible from the PC via the serial interface. The SEW MOVITOOLS software is used for this purpose. It is also possible to transfer parameter settings to several MOVIDRIVE® drive inverters via PC.

MOVIDRIVE® can be equipped with isolated RS-232 and RS-485 interfaces. The RS-232 interface is configured as a 9-pin sub D female connector (EIA standard) and the RS-485 interface as a terminal connection. The interfaces are accommodated in a housing for plugging onto the inverter (TERMINAL option slot). The option can be plugged on during operation. The transmission rate of both interfaces is 9600 baud.

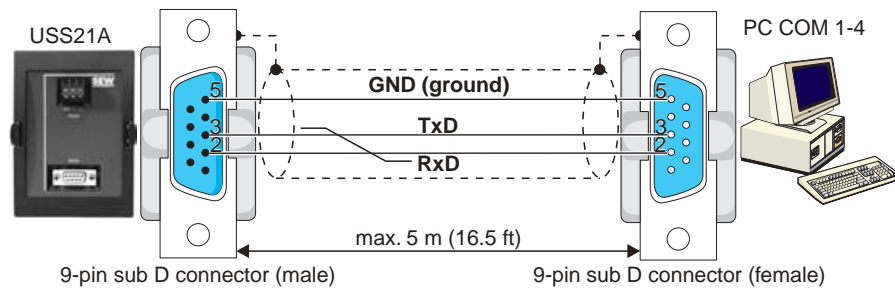


DBG11A and USS21A are connected to the same inverter slot (TERMINAL) and cannot be used at the same time.

RS-232 interface

Use a commercially available serial interface cable (shielded!) for connecting a PC to MOVIDRIVE® with the USS21A option.

Important: 1:1 cabling



02399AEN

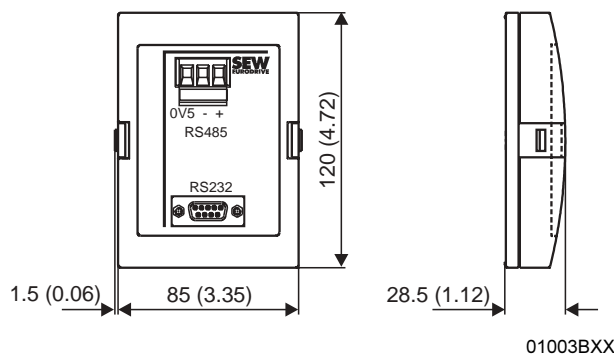
Fig. 3: Connection cable USS21A – PC

RS-485 interface

A maximum of 16 MOVIDRIVE® units can be networked for communications purposes (max. total cable length 200 m (660 ft)) via the RS-485 interface of the USS21A. Dynamic terminating resistors are permanently installed, so do not connect any external terminating resistors.

Unit addresses 0 – 99 are permitted with multipoint connections. In this case, the "peer-to-peer connection" must not be selected in MOVITOOLS. The communications address in MOVITOOLS and the RS-485 address of the MOVIDRIVE® unit (P810) must be identical.

Dimensions



01003BXX

Fig. 4: USS21A dimensions in mm (in)



2.2 Technical data

System bus (SBus)

Standard	CAN specification 2.0 parts A and B
Baud rate	either 125, 250, 500 or 1000 kbaud, factory setting 500 kbaud
ID range	3 – 1020
Address	can be set with parameter P813: 0 – 63
Number of process data words	fixed setting: 3 PD
Line length	depending on the baud rate, max. 320 m
Number of stations	max. 64



Only when P816 "SBus baud rate" = 1000 kbaud:

Do not combine MOVIDRIVE[®] compact MCH42A units with other MOVIDRIVE[®] units in the same system bus combination.

The units are allowed to be mixed at baud rates \neq 1000 kbaud.

RS-485 interface

Standard	RS-485
Baud rate	9.6 kbaud
Start bits	1 start bit
Stop bits	1 stop bit
Data bits	8 data bits
Parity	1 parity bit, supplementing to even parity
Line length	200 m between two stations
Number of stations	1 master and max. 31 slaves

RS-232 interface

Standard	DIN 66020 (V.24)
Baud rate	9.6 kbaud
Start bits	1 start bit
Stop bits	1 stop bit
Data bits	8 data bits
Parity	1 parity bit, supplementing to even parity
Line length	max. 5 m
Number of stations	1 master + 1 slave (peer-to-peer connection)



2.3 MOVILINK® and system bus

MOVILINK® protocol

This document provides a detailed description of the MOVILINK® serial interface protocol for the RS-485 interfaces of MOVIDRIVE® drive inverters. You can control the inverter and set its parameters via the RS-485 interface.

However, please bear in mind that this communications variant is a proprietary communication system for low-end applications.

The low speed of transmission and the significant time and effort needed to implement the various automation systems mean that SEW recommends the following fieldbus systems as the professional method of linking SEW inverters to machine control systems:

- PROFIBUS-DP
- INTERBUS
- INTERBUS with fiber optic cable
- CAN
- CANopen
- DeviceNet

These fieldbus systems are supported by SEW and by all well-known manufacturers of automation systems.

The MOVILINK® protocol for serial interfaces in the new SEW range of inverters, MOVIDRIVE® and MOVIMOT®, enables you to set up a serial bus connection between a higher-level master and several SEW inverters. For example, masters may take the form of programmable logic controllers, PCs or even SEW inverters with PLC functions (IPOS^{plus}®). Generally speaking, the SEW inverters function as slaves in the bus system.

The MOVILINK® protocol allows both of the following applications to be implemented: automation tasks such as control and parameter setting of the drives by means of cyclical data exchange, startup and visualization tasks.

Features

The principal features of the MOVILINK® protocol are:

- Support for the master/slave structure via RS-485 with one master (single master) and at most 31 slave stations (SEW inverters).
- Support for peer-to-peer connection via RS-232.
- User-friendly implementation of the protocol in a simple and reliable telegram structure with fixed telegram lengths and a unique start identifier
- Data interface to the basic unit in accordance with the MOVILINK® profile. This means the user data sent to the drive are transmitted to the inverter in the same way as via the other communications interfaces (PROFIBUS, INTERBUS, CAN, CANopen, DeviceNet, etc.).
- Access to all drive parameters and functions, i.e. it can be used for startup, service, diagnosis, visualization and automation tasks
- Startup and diagnostic tools on the basis of MOVILINK® for PC (e.g. MOVITOOLS/SHELL and MOVITOOLS/SCOPE).



System bus (SBus)

The SBus is a CAN bus in accordance with the CAN specification 2.0, parts A and B. It supports all services in the SEW MOVILINK® unit profile. In addition, you can exchange IPOS^{plus}® variables via the SBus independently of the profile.

The unit behavior of the inverter which forms the basis of CAN operation is referred to as the unit profile. It is independent of any particular fieldbus and is therefore a uniform feature. This provides you, the user, with the opportunity of developing applications regardless of the fieldbus.

MOVIDRIVE® offers digital access to all drive parameters and functions via the SBus. The drive inverter is controlled via high-speed process data. These process data telegrams let the user enter setpoints, such as the setpoint speed, ramp generator time for acceleration/deceleration, etc. and trigger various drive functions such as enable, control inhibit, normal stop, rapid stop, etc. You can also use these telegrams to read back actual values from the drive inverter, such as the actual speed, current, unit status, error number and reference signals.

The exchange of parameter data via the MOVILINK® parameter channel lets you create applications in which all important drive parameters are stored in the programmable master controller. This means there is no need to manually set the parameters on the drive inverter itself, which is frequently a rather time-consuming task. IPOS^{plus}® provides the MOVILNK command for the exchange of parameter data and process data with other MOVILINK® stations. As a result, MOVIDRIVE® can operate as the master via IPOS^{plus}® and control other units.

The process data and the drive parameters can be sent to a synchronization telegram synchronously or asynchronously.

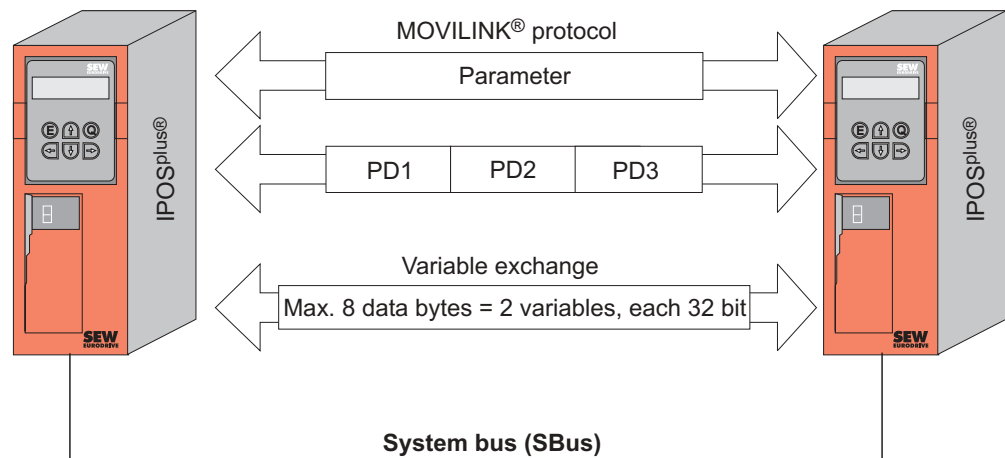


Fig. 5: Variants of SBus communication

02244BEN

Using the SBus requires additional monitoring functions such as time monitoring (SBus timeout delay) or special emergency-off concepts. You can adapt the monitoring functions of MOVIDRIVE® specifically to your application. You can determine which error response the drive inverter should trigger in the event of a timeout. A rapid stop is a good idea for many applications, although this can also be achieved by "freezing" the last setpoints so the drive continues operating with the most recently valid setpoints (e.g. conveyor belt). You can still implement emergency-off concepts which are independent of the bus and use the terminals of the drive inverter because the functions of the control terminals are still active when the SBus is in operation.



The MOVIDRIVE® drive inverter offers you numerous diagnostic options for startup and service purposes. An easy-to-use diagnostics tool is provided in the MOVITOOLS/SHELL PC software. This software makes it possible to call up a detailed display of the bus and unit status as well as setting all drive parameters.

Variable telegrams Not only does the cyclical and acyclical variable exchange function create an interface via which variables can be exchanged between several MOVIDRIVE® units, it is also possible to implement partial functions for specific profiles in external units. These external units may support the CANopen or DeviceNet protocol.



3 Installation

3.1 System bus (SBus) installation

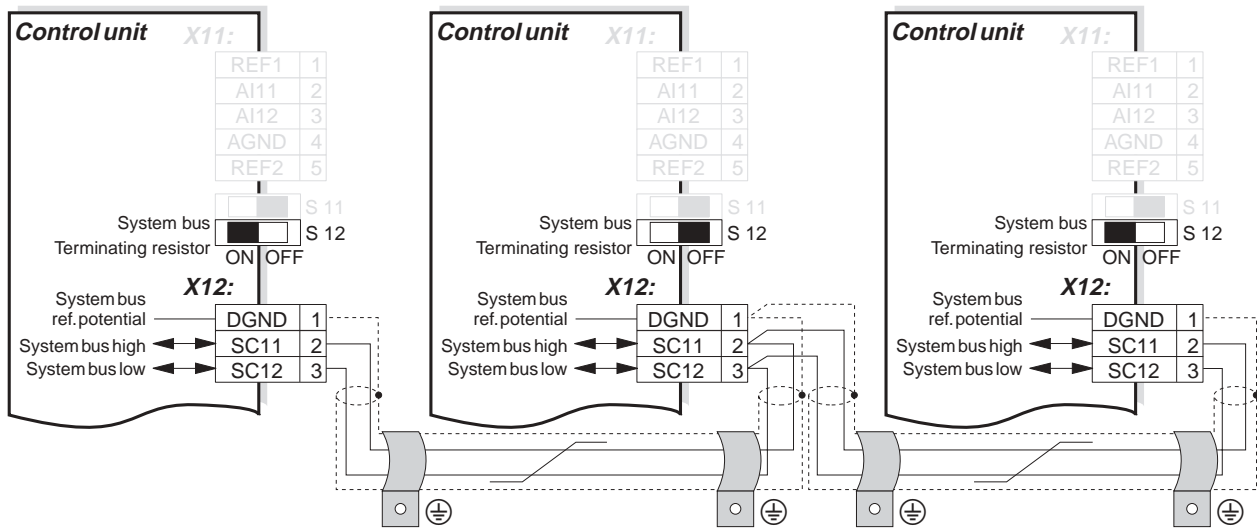


Only when P816 "SBus baud rate" = 1000 kbaud:

Do not combine MOVIDRIVE® compact MCH42A units with other MOVIDRIVE® units in the same system bus combination.

The units are allowed to be mixed at baud rates ≠ 1000 kbaud.

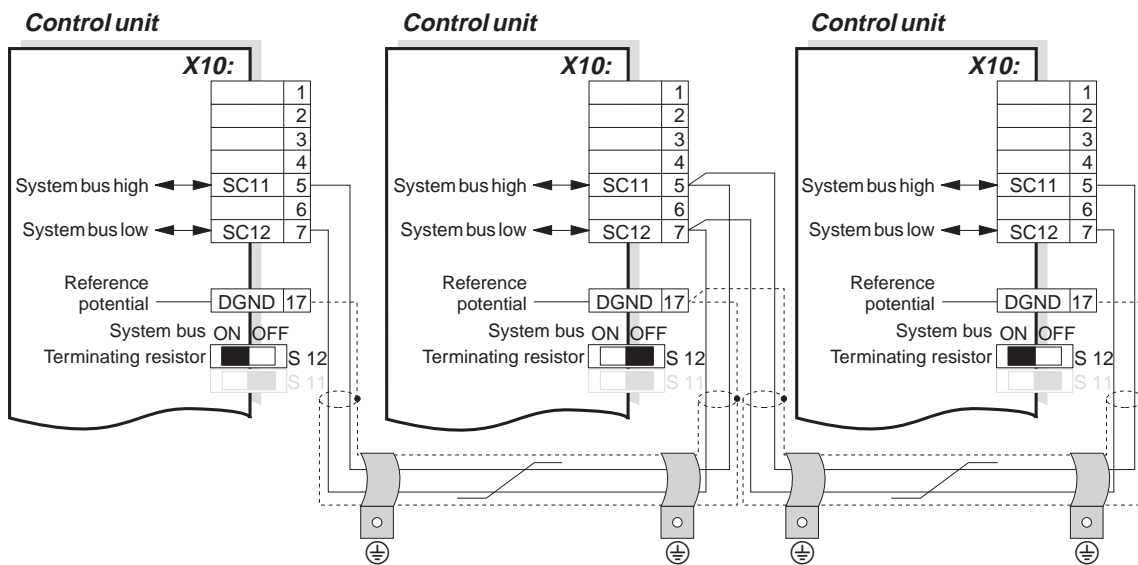
MOVIDRIVE® MD_60A



02205BEN

Fig. 6: System bus connection MOVIDRIVE® MD_60A

MOVIDRIVE® compact MCF/MCV/MCS4_A

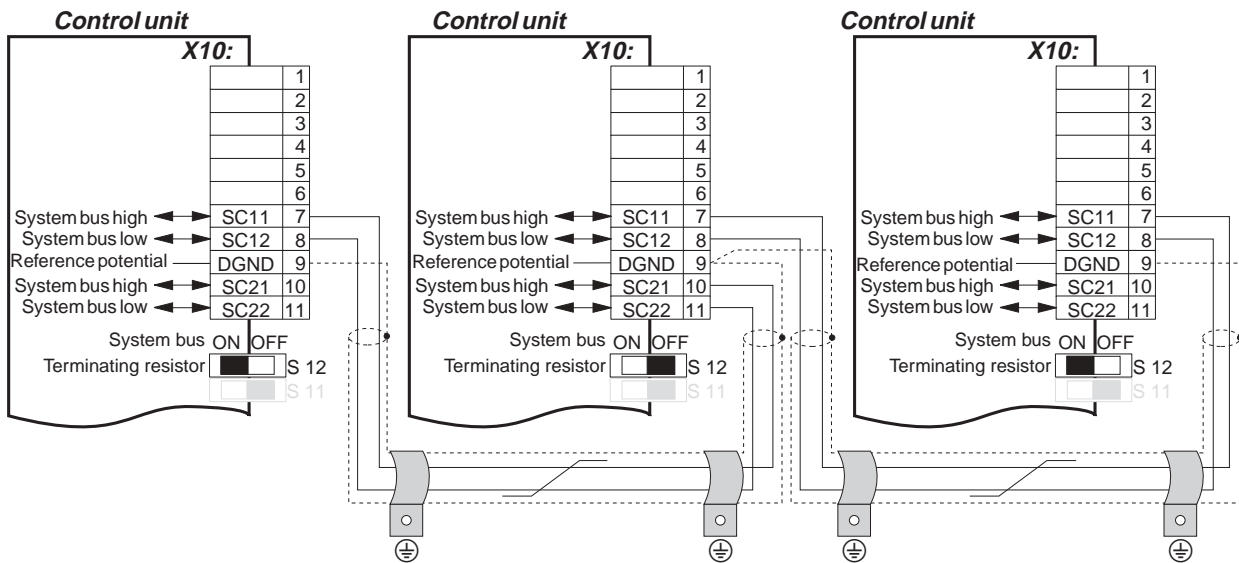


02411AEN

Fig. 7: System bus connection MOVIDRIVE® compact MCF/MCV/MCS4_A



MOVIDRIVE® compact MCHS4_A



05210AEN

Fig. 8: System bus connection MOVIDRIVE® compact MCH4_A

SBus MCH4_A: Connect the terminating equipment to SC11/SC12. SC21/SC22 are only active when S12 = OFF.

Cable specification

- Use a 2-core twisted and shielded copper cable (data transmission cable with shield comprising copper braiding). The cable must meet the following specifications:
 - Conductor cross section 0.75 mm² (AWG 18)
 - Cable resistance 120 Ω at 1 MHz
 - Capacitance per unit length ≤ 40 pF/m (12 pF/ft) at 1 kHz
 Suitable cables are CAN bus or DeviceNet cables, for example.

Shield contact

- Connect the shield at either end to the electronics shield clamp of the inverter or the master control and ensure the shield is connected over a large area. Also connect the ends of the shield to DGND.

Line length

- The permitted total line length depends on the baud rate setting of the SBus (P816):
 - 125 kbaud → 320 m (1056 ft)
 - 250 kbaud → 160 m (528 ft)
 - **500 kbaud → 80 m (264 ft)**
 - 1000 kbaud → 40 m (132 ft)

Terminating resistor

- Switch on the system bus terminating resistor (S12 = ON) at the start and finish of the system bus connection. Switch off the terminating resistor on the other units (S12 = OFF).



- There must not be any potential displacement between the units which are connected together using the SBus. Take suitable measures to avoid a potential displacement, e.g. by connecting the unit ground connectors using a separate lead.

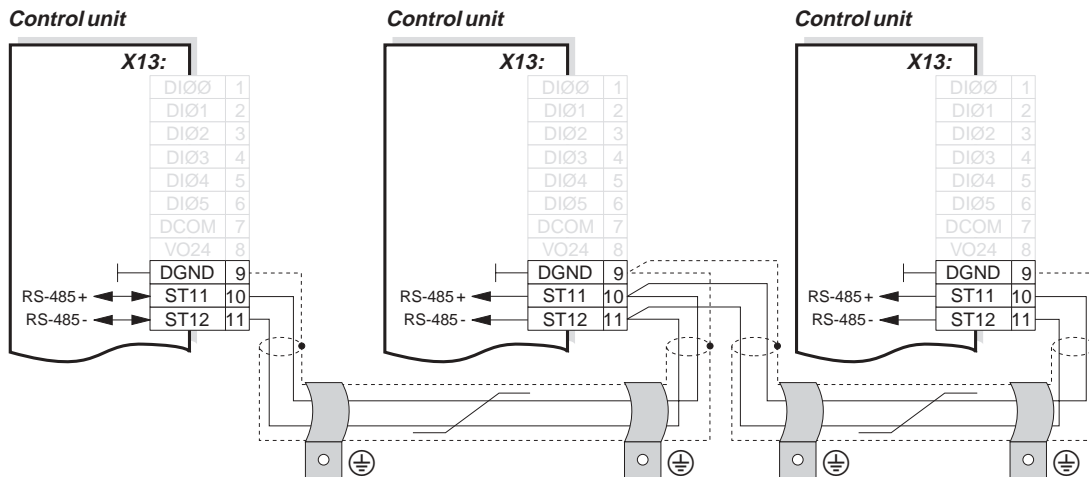


3.2 RS-485 interface installation

MOVIDRIVE® MD_60A

The RS-485 interface is routed to terminals X13:10/11 and, in parallel, to the TERMINAL option slot. The RS-485 interface can only be accessed via the TERMINAL option slot when the "serial interface type USS21A" option is attached.

RS-485 connection via terminals X13:10/11



02206AEN

Fig. 9: RS-485 connection via X13:10/11

Cable specification

- Use a 2-core twisted and shielded copper cable (data transmission cable with shield comprising copper braiding). The cable must meet the following specifications:

- Conductor cross section 0.5 – 0.75 mm² (AWG 20 – 18)
- Cable resistance 100 – 150 Ω at 1 MHz
- Capacitance per unit length ≤ 40 pF/m (12 pF/ft) at 1 kHz

The following cable is suitable, for example:

- BELDEN (www.belden.com), data cable type 3105A

Shield contact

- Connect the shield at either end to the electronics shield clamp of the inverter or the machine control and ensure the shield is connected over a large area. Also connect the ends of the shield to DGND.

Line length

- The permitted total line length is 200 m (660 ft).

Terminating resistor

- Dynamic terminating resistors are fitted. Do not connect **any external terminating resistors!**



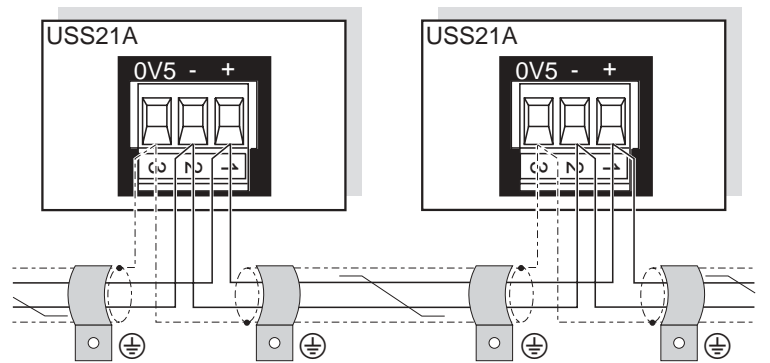
- There must not be any potential displacement between the units which are connected via RS-485. Take suitable measures to avoid a potential displacement, e.g. by connecting the unit ground connectors using a separate lead.



USS21A serial interface

- With MOVIDRIVE® MD_60A drive inverters, the RS-485 interface can also be accessed using the "serial interface type USS21A" option.
- With MOVIDRIVE® compact drive inverters, the RS-485 interface can only be accessed using the "serial interface type USS21A" option.

RS-485 connection via USS21A



00997CXX

Fig. 10: RS-485 interface of the USS21A

Cable specification

- Use a 2-core twisted and shielded copper cable (data transmission cable with shield comprising copper braiding). The cable must meet the following specifications:
 - Conductor cross section 0.5 – 0.75 mm² (AWG 20 – 18)
 - Cable resistance 100 – 150 Ω at 1 MHz
 - Capacitance per unit length ≤ 40 pF/m (12 pF/ft) at 1 kHz
- The following cable is suitable, for example:
 - BELDEN (www.belden.com), data cable type 3105A

Shield contact

- Connect the shield at either end to the electronics shield clamp of the inverter and ensure the shield is connected over a large area. Also connect the ends of the shield to DGND.

EIA standard

- Max. transmission rate 9600 baud
- Max. 32 stations (each unit with USS21A counts as two stations)
- Max. cable length 200 m (660 ft) in total
- Dynamic terminating resistor with fixed installation



3.3 RS-232 interface installation

With MOVIDRIVE® MD_60A and MOVIDRIVE® *compact*, the RS-232 interface can only be accessed using the "serial interface type USS21A" option.

RS-232 connection

- Use a shielded standard interface cable for connecting to the RS-232 interface.

Important: 1:1 cabling

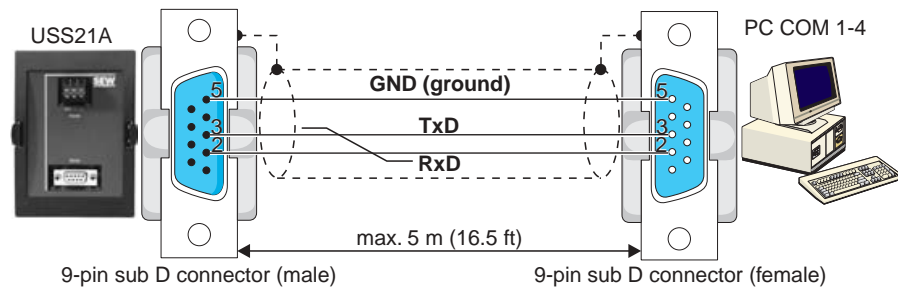


Fig. 11: PC connection via RS-232

02399AENdf



4 RS-485 Communication

4.1 Telegrams

Telegram traffic Both cyclical and acyclical data exchange are used in drive engineering. Cyclical telegrams via the serial interface are used in automation applications, particularly for drive control. The master station must ensure cyclical data exchange in this case.

Cyclical data exchange Cyclical data exchange is used predominantly for controlling the inverters via the serial interface. In this process, the master continuously sends telegrams containing setpoints (request telegrams) to an inverter (slave) and then waits for a response telegram with actual values from the inverter. After a request telegram has been sent to an inverter, the master expects the response telegram within a defined length of time (response delay time). The inverter only sends back a response telegram if it has received a request telegram sent to its slave address without any errors. The inverter monitors whether the data communication fails during the cyclical data exchange. If communication does fail, the inverter triggers a timeout response if it does not receive a new request telegram from the master within an adjustable time.

MOVILINK[®] also offers the opportunity to perform acyclical service and diagnostic tasks even during cyclical communication without changing the type of telegram.

Acyclical data exchange Acyclical data exchange is principally used for startup and diagnostics. The inverter does not monitor the communications link in this case. The master can send telegrams to the inverter at irregular intervals in acyclical mode.



Telegram structure

The entire data exchange is performed using only two types of telegram. It involves the master sending a request containing data to the inverter, in the form of a request telegram. The inverter answers with a response telegram. When word information (16-bit) is sent within the user data, the high byte is always sent first and the low byte last. In the case of double word information (32-bit), the high word is sent first and the low word last. Coding of the user data is not part of the protocol. The content of the user data is explained in detail in the MOVIDRIVE® Fieldbus Unit Profile manual.

Request telegram structure

Fig. 12 shows the structure of the request telegram which the master sends to the inverter. Each telegram starts with an idle time on the bus, referred to as the start pause, followed by a start character. Different start characters are used so that it is possible to clearly differentiate between request and response telegrams. The request telegram starts with the start character SD1 = 02_{hex}, followed by the slave address and the PDU type. The PDU type is followed by the Protocol data unit (PDU) and ends with the Block check character (BCC).

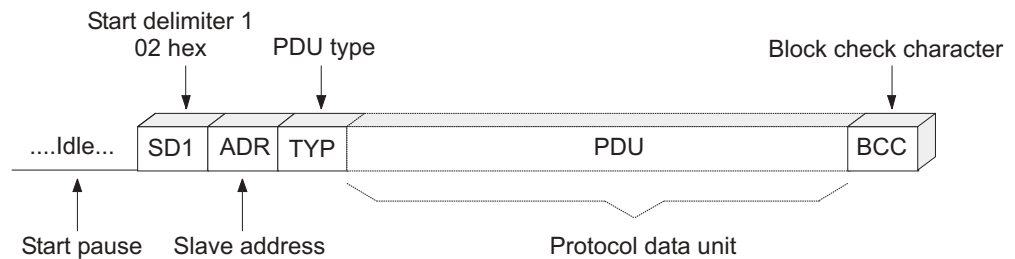


Fig. 12: Structure of the request telegram

01485BEN

Response telegram structure

Fig. 13 shows the structure of the response telegram by means of which the inverter (slave) responds to a request sent by the master. In turn, each response telegram starts with a start pause, followed by a start character. The response telegram starts with the start character SD2 = 1D_{hex}, followed by the slave address and the PDU type so that it is possible to clearly differentiate between request and response telegrams. The PDU type is followed by the Protocol data unit (PDU) and ends with the Block check character (BCC).

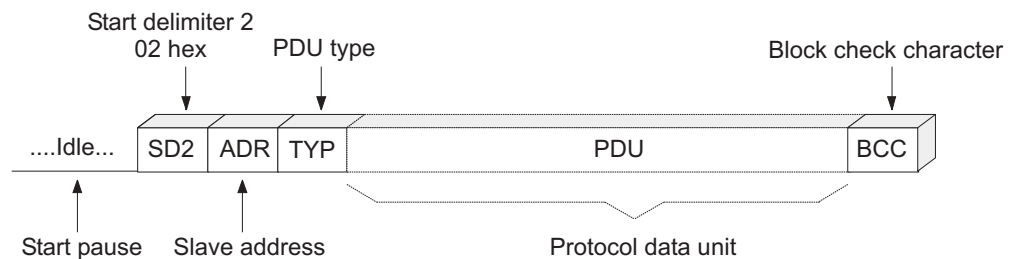


Fig. 13: Structure of the response telegram

01487BEN



Start pause (idle)

The master must observe a start pause of at least 3.44 ms before sending the start character SD1 (02_{hex}) so that the inverter can definitively identify the start of a request telegram. This pause prevents the bit combination 02_{hex}, which may also occur in the user data, from being erroneously interpreted as the start character. As a result, the start pause forms part of the start character. After it has received a valid request telegram, the inverter waits for an idle time of at least 3.44 ms before sending back the response telegram with the start character SD2 (1D_{hex}). This enables the master to clearly identify the start character of a response telegram as well. In case the transmission of a valid request telegram is canceled by the master, a new request telegram cannot be sent until at least two start pauses (6.88 ms) have elapsed.

Start character (SD1 / SD2)

The start character and the preceding start pause detect the commencement and the data direction of a new telegram. The following table shows the allocation of the start character to the data direction.

SD1	02 _{hex}	Request telegram	Master → inverter
SD2	1D _{hex}	Response telegram	Inverter → master



4.2 Addressing and transmission process

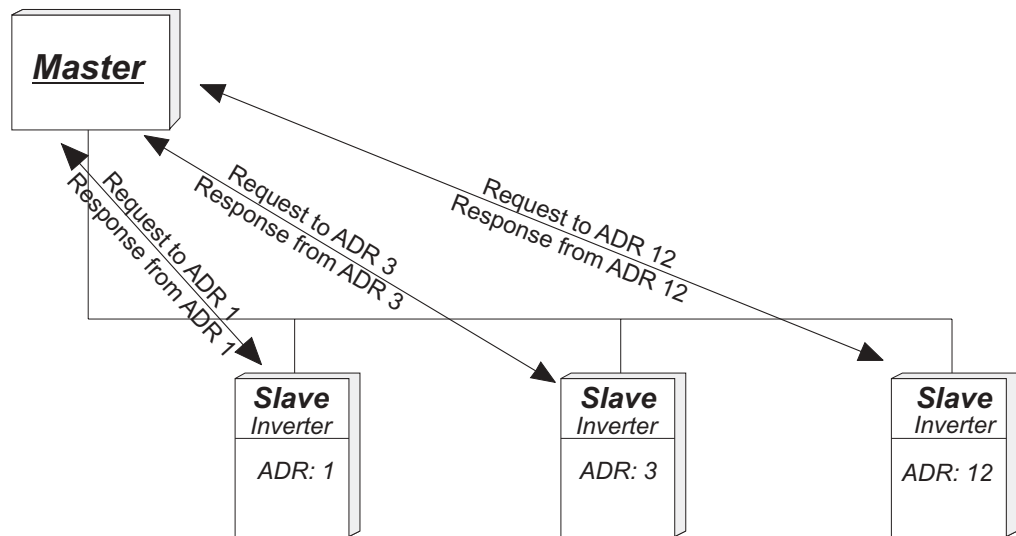
Address byte (ADR)

The address byte always specifies the slave address regardless of the data direction. Therefore, the ADR character in a request telegram specifies the address of the inverter which is to receive the request. In the opposite direction, the master can tell from which inverter the response telegram was sent. Generally speaking, there is only one master in the system. This means the master is not addressed. In addition to individual addressing, the MOVILINK[®] protocol also offers further addressing options. The following table shows the address areas and what they mean.

ADR	Meaning
0 – 99	Individual addressing within an RS-485 bus
100 – 199	Group addressing (multicast) Special case of group address 100: Means "Not assigned to any group", i.e. ineffective
253	Local address: Only effective in conjunction with IPOS ^{plus} [®] as master and the MOVILINK command. For communication within the unit.
254	Universal address for peer-to-peer communication
255	Broadcast address

Individual addressing

Each inverter can be addressed directly via addresses 0 – 99. Each request telegram from the master is answered by a response telegram from the inverter.



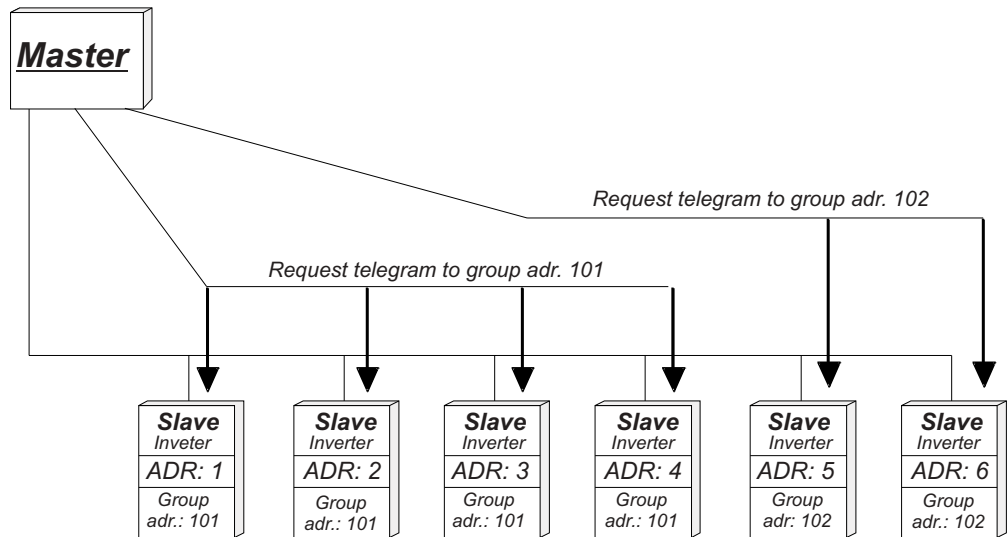
01488BEN

Fig. 14: Individual addressing via unit address 232/485



Group addressing (multicast)

Each inverter possesses an adjustable group address in addition to its individual address. This setup enables the user to form groups with various stations and then address the individual stations in a group simultaneously using the group address. No response telegram is sent back to the master in the case of group addressing. This means it is not possible to request data from the inverter. Also, there is no response when data are written. You can create up to 99 groups.



01489BEN

Fig. 15: Addressing individual groups

Universal addressing for peer-to-peer connection

Every inverter can be addressed via the universal address 254 regardless of the individual address which has been set for it. The advantage of this method is that peer-to-peer connections can be established via the RS-232 interface without necessarily knowing the currently set individual address. Every inverter station is addressed with this universal address, which means this method must not be used in multipoint connections (e.g. RS-485 bus). Otherwise, there would be data collisions on the bus because every inverter would send a response telegram after receiving the request telegram.



01490BEN

Fig. 16: Addressing in peer-to-peer connections with universal address 254



Broadcast address The broadcast address 255 permits a broadcast to all inverter stations. The request telegram sent out by the master to broadcast address 255 is received by all inverters, but they do not reply. Consequently, this addressing variant is predominantly used for transferring setpoints. The master can send broadcast telegrams with a minimum time interval of 25 ms, i.e. an idle time of at least 25 ms must be observed between the last character sent in a request telegram (BCC) and the start of a new request telegram (SD1).

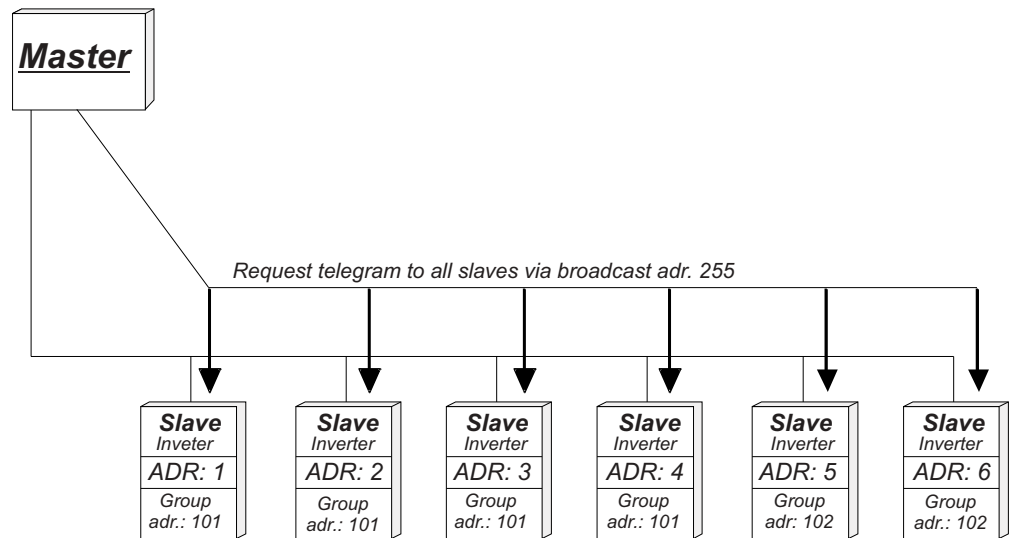


Fig. 17: Addressing individual groups

01491BEN



Structure and length of user data

PDU type (TYP)

The TYP byte describes the structure and the length of the user data which succeed it (protocol data unit or PDU). Fig. 18 shows the structure of the type byte.

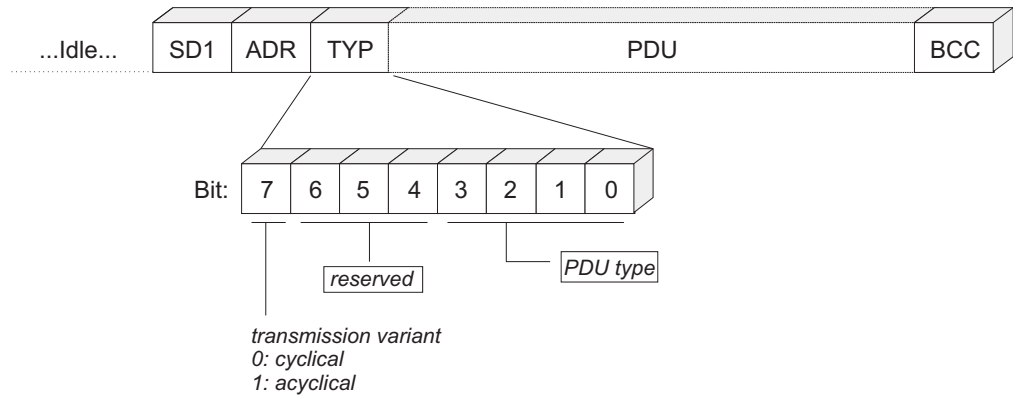


Fig. 18: Structure of the TYP byte

01492BEN

Bit 7 of the TYP byte is used to differentiate between cyclical or acyclical transmission of user data. A request telegram with the cyclical transmission variant signals to the inverter that the data sent by the master will be updated cyclically. Consequently, a response monitoring function can be activated in the inverter. This means a timeout response is triggered if the inverter does not receive a new cyclical request telegram within an adjustable timeout delay.

The following tables show the PDU types for cyclical and acyclical transmission. However, not all PDU types are supported (depending on the type of inverter). The special PDU types are not significant for general serial communication, and are thus not included in the operator documentation. The length of the telegram depends on the related PDU type and is always calculated as follows:

$$\text{Telegram length} = \text{PDU length} + 4.$$

CYCLICAL transmission

PDU types in CYCLICAL transmission:

TYP byte		PDU name	Description	PDU length in bytes	Telegram length in bytes
00 _{hex}	0 _{dec}	PARAM + 1PD	8 bytes parameter channel + 1 process data word	10	14
01 _{hex}	1 _{dec}	1PD	1 process data word	2	6
02 _{hex}	2 _{dec}	PARAM + 2PD	8 bytes parameter channel + 2 process data words	12	16
03 _{hex}	3 _{dec}	2PD	2 process data words	4	8
04 _{hex}	4 _{dec}	PARAM + 3PD	8 bytes parameter channel +3 process data words	14	18
05 _{hex}	5 _{dec}	3PD	3 process data words	6	10
06 _{hex}	6 _{dec}	PARAM + 0PD	8 bytes parameter channel without process data	8	12



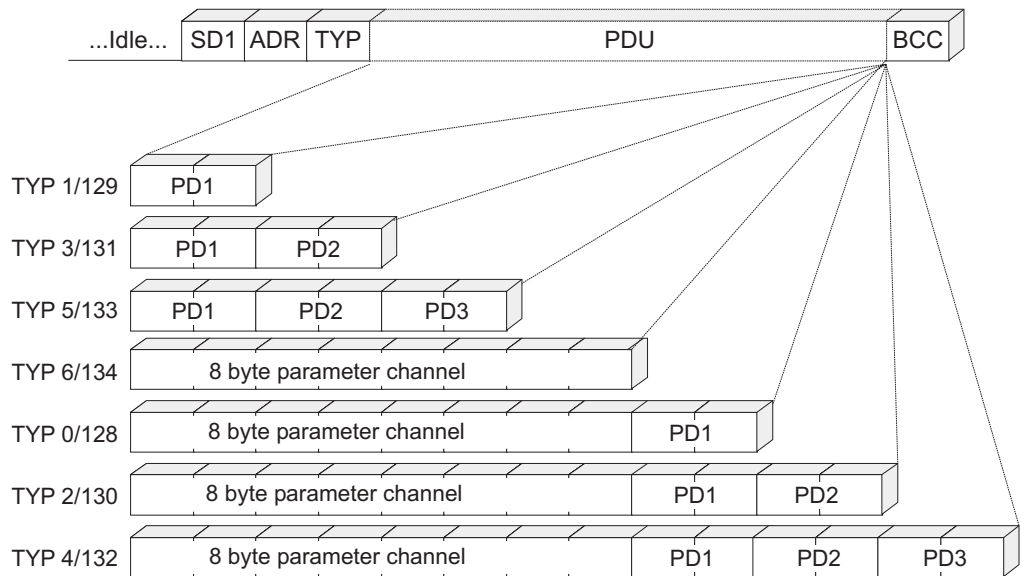
ACYCLICAL transmission

PDU types in ACYCLICAL transmission:

TYP byte		PDU name	Description	PDU length in bytes	Telegram length in bytes
80 _{hex}	128 _{dec}	PARAM + 1PD	8 bytes parameter channel + 1 process data word	10	14
81 _{hex}	129 _{dec}	1PD	1 process word	2	6
82 _{hex}	130 _{dec}	PARAM + 2PD	8 bytes parameter channel + 2 process data words	12	16
83 _{hex}	131 _{dec}	2PD	2 process data words	4	8
84 _{hex}	132 _{dec}	PARAM + 3PD	8 bytes parameter channel +3 process data words	14	18
85 _{hex}	133 _{dec}	3PD	3 process data words	6	10
86 _{hex}	134 _{dec}	PARAM + 0PD	8 bytes parameter channel without process data	8	12

The standard PDU types are made up of the MOVILINK[®] parameter channel and a process data channel. Please refer to the MOVIDRIVE[®] Fieldbus Unit Profile for coding of the parameter channel and the process data.

Fig. 19 shows the structure of a request telegram with the standard PDU types. The corresponding response telegram has the same structure, except for the start character SD2.



01493BEN

Fig. 19: Structure of the request telegram with the standard PDU types



Block check character BCC

Transmission reliability

The transmission reliability of the MOVILINK[®] protocol is improved by the combination of character parity and block parity. This involves setting the parity bit for each character of the telegram in such a way that the number of binary ones, including the parity bit, is even. This means supplementing by the parity bit results in even character parity.

Block parity offers extra security. In this case, the telegram is supplemented by an additional block check character (BCC). Each single bit of the block check character is set in such a way that the telegram character is set to even parity again for all equivalent information bits. Block parity is implemented in the program structure by an EXOR logic operation of all telegram characters. The result is transmitted at the end of the telegram in the BCC. The block check character itself is also safeguarded by means of even character parity.

Creating the block check character

By way of example, the following table shows how the block check character is created for a PDU type 5 cyclical telegram with 3 process data words. The EXOR logic operation on the characters SD1 – PD3_{low} results in the value 57_{hex} as the block check character BCC. This BCC is sent as the last character in the telegram. Once the receiver has received the individual characters, it performs a character parity check. Following this step, the block check character is created from the received characters SD1 – PD3_{low} in accordance with the procedure below. The telegram has been correctly transmitted if the calculated and received BCCs are identical and there is no character parity error. Otherwise, a transmission error has occurred.

	Stop	Parity									Start
SD1: 02 hex	1	0	0	0	0	0	0	0	1	0	
ADR: 01 hex	1	0	0	0	0	0	0	0	0	1	
TYP: 05 hex	0	0	0	0	0	0	0	1	0	1	
PD1 high: 00 hex	0	0	0	0	0	0	0	0	0	0	
PD1 low: 06 hex	0	0	0	0	0	0	0	1	1	0	
PD2 high: 3A hex	0	0	0	1	1	1	0	1	0		
PD2 low: 98 hex	1	1	0	0	1	1	0	0	0		
PD3 high: 01 hex	1	0	0	0	0	0	0	0	0	1	
PD3 low: F4 hex	1	1	1	1	1	0	1	0	0		
calculated BCC: 57 hex	1	0	1	0	1	0	1	1	1	1	

Fig. 20: Creating the block check character BCC

01494BEN



Transmission process

An asynchronous serial transmission procedure is used. This is supported by the UART components of digital technology which are generally and commonly employed. This means the MOVILINK[®] protocol can be implemented on almost all controls and master modules.

Character frame

Each character in the MOVILINK[®] protocol consists of 11 bits and has the following structure:

- 1 start bit
- 8 data bits
- 1 parity bit, supplementing to even parity
- 1 stop bit

Each transmitted character starts with a start bit (always logical 0). This is followed by 8 data bits and the parity bit. The parity bit is set in such a way that the number of logical ones in the data bits, including the parity bit, is an even number. The character is completed by a stop bit which is always set to the logical level 1. This level remains on the transmission medium until a new start bit signals the start of a new character transmission.

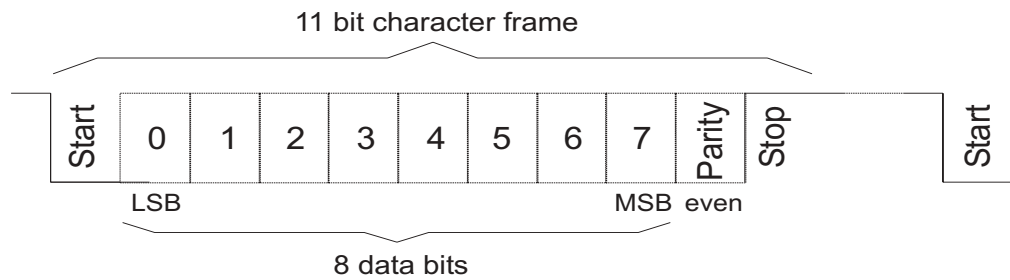


Fig. 21: Character frame

01495BEN



Transmission rate and transmission mechanisms

The transmission rate is 9600 baud. The communication link is monitored by the master and the inverter themselves. The master monitors the response delay time; the inverter monitors the receipt of cyclical request telegrams from the master.

Response delay time of the master

A response delay time is generally programmed on the higher-level master system. The response delay time is the time interval between the last character in the request telegram (BCC) being sent and the start of the response telegram (SD2). The maximum permitted response delay time is 50 ms. There has been a transmission error if the inverter does not respond within this time. Check the interface cable or the coding of the transmitted request telegram. For application reasons, the request telegram should now be repeated again or the next inverter should be addressed.

Character delay time

The time gap between the transmission of the characters in a request telegram must be shorter than the start pause. Otherwise, the inverter might interpret a character it receives containing 02_{hex} or 1D_{hex} as a start character.

RS-485 timeout delay of the inverter

The maximum permitted time interval between two cyclical request telegrams is set in MOVIDRIVE[®] using parameter P812 "RS485 timeout delay". A valid request telegram must be received within this time interval. Otherwise, the inverter triggers an RS-485 timeout error and performs a defined error response.

The MOVIDRIVE[®] is kept in a safe status until the first request telegram is received once the power is switched on or an error has been performed. "t" (= timeout active) appears on the 7-segment display of an enabled inverter; the enable setting does not have any effect. The enable takes effect once the telegram has been received and the drive starts moving.

If the inverter is controlled via the RS-485 interface (P100 "Setpoint source" = RS-485 / P101 "Control signal source" = RS-485) and an error response involving a warning has been programmed, the process data most recently received take effect following an RS-485 timeout and re-establishment of communication.



The RS-485 timeout acts jointly on both RS-485 interfaces. Timeout monitoring of the second interface is ineffective when the DBG11A keypad is connected, because the DBG11A continuously sends request telegrams to the inverter, thereby triggering the timeout mechanism.



Processing the request/response telegrams

The inverter only processes request telegrams which have been received without errors and are correctly addressed. The following reception errors can be recognized:

- Parity error
- Character frame error
- Character delay time exceeded with request telegram
- Address incorrect
- PDU type incorrect
- BCC incorrect
- Response delay time elapsed (master)
- → Possible send repeat
- RS-485 timeout occurred (inverter)
- → Triggering timeout response

The inverter does not respond to incorrectly received request telegrams! These reception errors have to be evaluated on the master end in order to safeguard data transmission.



4.3 Data contents and PDU types

Data contents

The data content of the request and response telegram is structured with a process data area and a parameter channel in accordance with the MOVILINK[®] communication and unit profile. Refer to the documentation for the particular type of inverter for information about the coding of the process data. The individual items of process input and output data are interpreted in accordance with the fieldbus unit profile, and are not explained further in this specification.

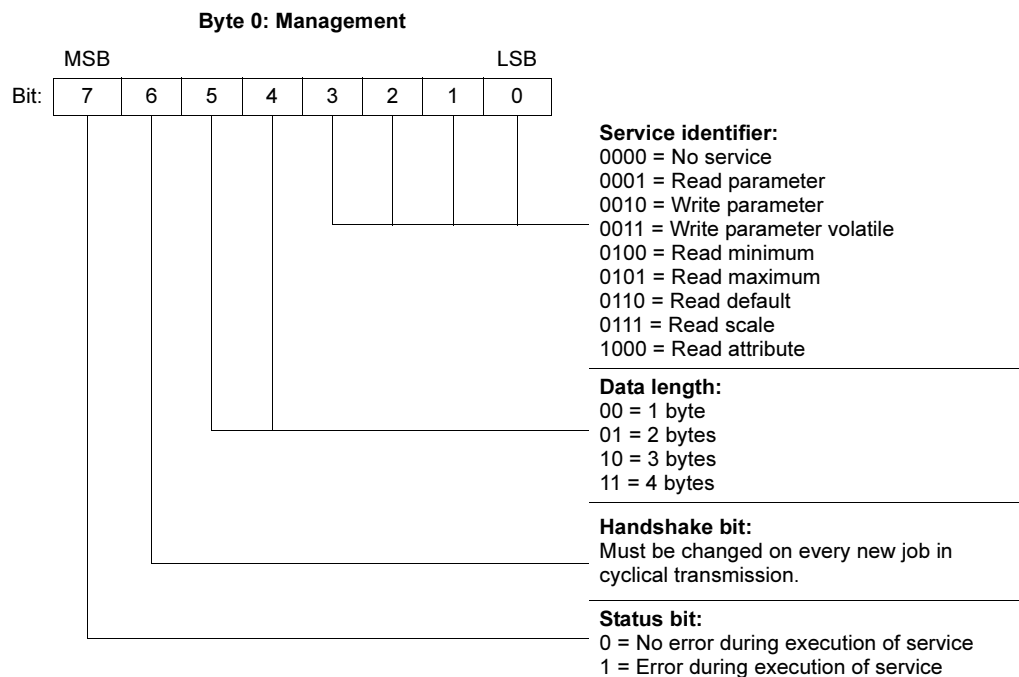
Structure of the MOVILINK[®] parameter channel

The MOVILINK[®] parameter channel affords access to all drive parameters of the drive inverters, regardless of the bus. Special services are available within this parameter channel in order to permit various items of parameter information to be read. In principle, it is made up of a management byte, a reserved byte, an index word and four data bytes.

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Managmnt.	Reserved	Index high	Index low	Data MSB	Data	Data	Data LSB
Parameter list				4 byte data			

Management of the parameter channel

The entire parameter setting sequence is coordinated with byte 0 "Management." This byte is used for providing important service parameters such as service identifier, data length, version and status of the service performed. Bits 0 – 3 contain the service identifier, i.e. they define which service will be performed. Bit 4 and bit 5 specify the data length in bytes for the write service. This should be set to 4 bytes for all SEW drive inverters.





The handshake bit (bit 6) is used in cyclical transmission variant as an acknowledgement bit between the control and the inverter. In this variant, the parameter channel is transmitted cyclically, with the process data if necessary. For this reason, implementation of the service in the inverter has to be triggered by edge control using the handshake bit 6. To permit this, the value of this bit is altered for each new service to be performed (toggle). The inverter uses the handshake bit to signal whether the service was performed or not. The service has been performed as soon as the handshake bit received in the control corresponds to the one which was sent. Status bit 7 indicates whether it was possible to carry out the service properly or if there were errors.

Index addressing

Byte 2 "Index high" and byte 3 "Index low" determine the parameter to be read or written via the fieldbus system. The parameters of the inverter are addressed using the same index in all communications interfaces. Byte 1 should be viewed as reserved and must always be set to 0x00.

Data range

The data are located in byte 4 to byte 7 of the parameter channel. This means up to 4 bytes of data can be transmitted per service. The data are always entered with right-justification, i.e. byte 7 contains the least significant data byte (data-LSB) whereas byte 4 is the most significant data byte (data-MSB).

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Managmnt.	Reserved	Index high	Index low	Data MSB	Data	Data	Data LSB
				High byte 1	Low byte 1	High byte 2	Low byte 2
				High word		Low word	
				Double word			

Incorrect performance of service

The status bit in the management byte is set to signal that a service has been performed incorrectly. The service was performed by the inverter if the received handshake bit is the same as the sent handshake bit. If the status bit now signals an error, the error code is entered in the data range of the parameter telegram. Bytes 4 – 7 send back the return code in a structured format.

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Managmnt.	Reserved	Index high	Index low	Error class	Error code	Add. code high	Add. code low



Status bit = 1: Incorrect performance of service



Description of the parameter services	Bits 0 – 3 of the management byte define the individual parameter services. The following parameter services are possible, although they are not supported by all inverters.
<i>No service</i>	This coding signals that there is no parameter service.
<i>Read parameter</i>	A drive parameter is read in this parameter service.
<i>Write parameter</i>	A drive parameter is written to the permanent memory in this parameter service. The written parameter value is stored in the non-volatile memory (e.g. in an EEPROM). This service should not be used for cyclical write access, because the memory modules only permit a limited number of write cycles.
<i>Write parameter volatile</i>	A drive parameter is written to the volatile memory in this parameter service, providing the parameter permits this. The written parameter value is only stored in the non-permanent RAM of the inverter; it is lost when the inverter is switched off. The value written last with the Write parameter is still available when the inverter is switched back on.
<i>Read minimum</i>	This service makes it possible to determine the smallest value (minimum) which can be set for a drive parameter. Coding is in the same way as for the parameter value.
<i>Read maximum</i>	This service makes it possible to determine the largest value (maximum) which can be set for a drive parameter. Coding is in the same way as for the parameter value.
<i>Read default</i>	This service makes it possible to determine the factory setting (default) of a drive parameter. Coding is in the same way as for the parameter value.
<i>Read scale</i>	This service makes it possible to determine the scaling of a parameter. When it is performed, the inverter supplies what are referred to as a quantity index and a conversion index.

Byte 4	Byte 5	Byte 6	Byte 7
Data MSB	Data	Data	Data LSB
Reserved		Quantity index	Conversion index

Quantity index:

The quantity index is used for coding physical quantities. This index provides a communication partner with information about which physical quantity is involved with the corresponding parameter value. Coding is performed in line with the sensor/actuator systems profile of the Profibus User Organization (PNO). The entry FF_{hex} means that no quantity index is specified. You can also refer to the list of parameters of the inverter for the quantity index.

**Conversion index:**

The conversion index is used for converting the transmitted parameter value into a basic SI unit. Coding is performed in line with the sensor/actuator systems profile of the Profibus User Organization (PNO).

Example:

Drive parameter: P131 Ramp t11 UP CW
 Quantity index: 4 (= time with the unit "second")
 Conversion index: -3 (10^{-3} = milli)
 Transmitted numerical value: 3000_{dec}

The drive inverter interprets the numerical value it received via the bus as follows:
 $3000 \text{ s} \times 10^{-3} = 3 \text{ s}$

Read attribute

This service makes it possible to read the access attributes and the index of the next parameter.

Byte 4	Byte 5	Byte 6	Byte 7
Data MSB	Data	Data	Data LSB
Next available index		Access attributes	

The access attributes are coded for specific devices and can be found by referring to the list of parameters for the corresponding families of inverters.

Reading a parameter

The handshake bit must be changed in the cyclical transmission variant so that the service processing (performance of the READ service) will be activated. When acyclical PDU types are used, the inverter processes each request telegram and thereby always performs the parameter channel.

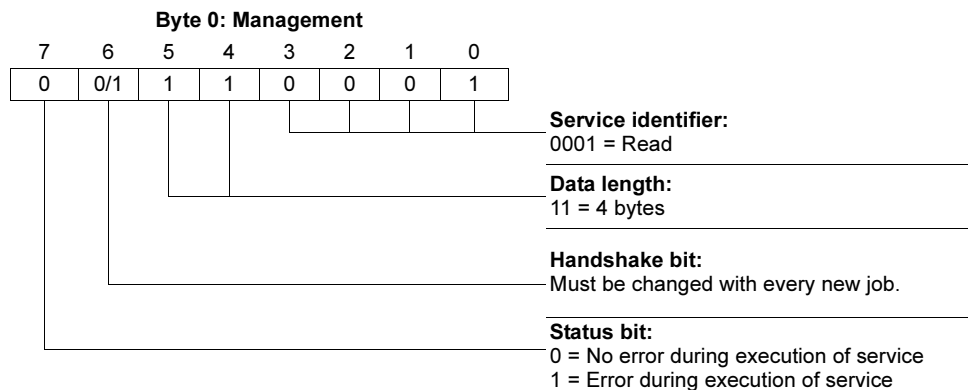
Parameter setting is performed as follows:

1. Enter the index of the parameter to be read in byte 2 (Index high) and byte 3 (Index low).
2. Enter the service identifier for the read service in the management byte (byte 0).
3. In cyclical PDU types, only transfer the read service to the inverter by changing the handshake bit. The parameter channel is always evaluated in acyclical PDU types.

Since this is a read service, the sent data bytes (bytes 4 – 7) and the data length (in the management byte) are ignored and consequently do not need to be set.



The inverter now processes the read service and sends the service confirmation back by setting the handshake bit to an equal value.



X = Not relevant
0/1 = Bit value is changed

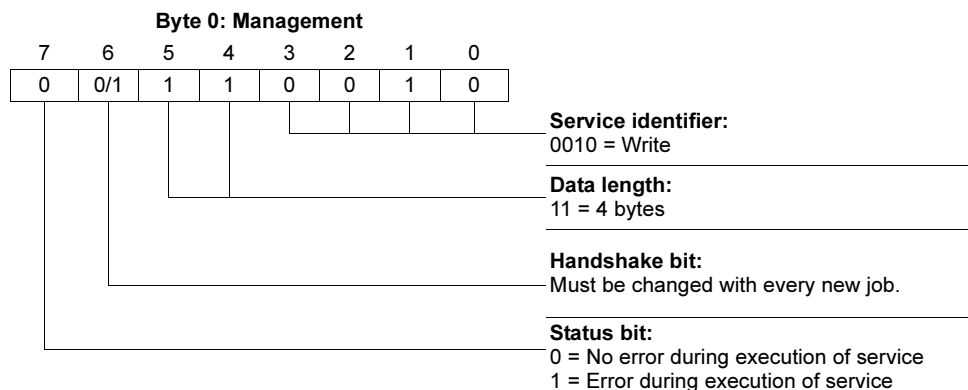
Writing a parameter

The handshake bit must be changed in the cyclical transmission variant so that the service processing (performance of the WRITE service) will be activated. When acyclical PDU types are used, the inverter processes each request telegram and thereby always performs the parameter channel.

Parameter setting is performed as follows:

1. Enter the index of the parameter to be written in byte 2 (Index high) and byte 3 (Index low).
2. Enter the data to be written in bytes 4 – 7.
3. Enter the service identifier and the data length for the write service in the management byte (byte 0).
4. In cyclical PDU types, only transfer the WRITE service to the inverter by changing the handshake bit. The parameter channel is always evaluated in acyclical PDU types.

The inverter now processes the write service and sends the service confirmation back by setting the handshake bit to an equal value.



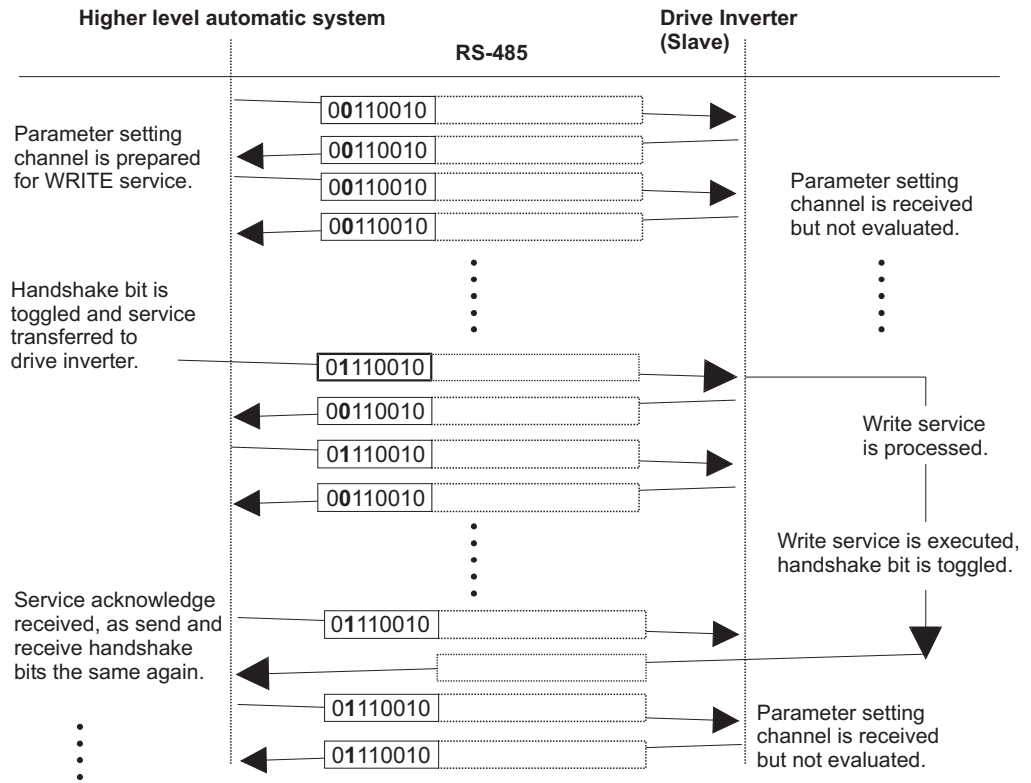
0/1 = Bit value is changed

The data length is 4 bytes for all parameters in SEW drive inverters.



Parameter setting with cyclical PDU types

Fig. 22 takes the example of the WRITE service to illustrate a parameter setting procedure between the control and the inverter concerning a cyclical PDU type. To simplify the sequence, only the management byte of the parameter channel is shown.



00152BEN

Fig. 22: Sequence of parameter setting with handshake bit

The parameter channel is only received and returned by the drive inverter while the master is preparing the parameter channel for the write service. The service is not activated until the moment the handshake bit is changed (in this example, when it changes from 0 to 1). The drive inverter now interprets the parameter channel and processes the write service; however, it continues to respond to all telegrams with handshake bit = 0. Confirmation that the service has been performed occurs when the handshake bit in the response telegram of the drive inverter is set to the same value. The master now detects that the received handshake bit is once again the same as the one which was sent. It can now prepare another parameter setting procedure.



Sample application

Control via three process data words

In this example, the inverter (e.g. MOVIMOT®) is controlled via three process data words with PDU type 5 (3PD acyclical). The master sends three process output data words (PO) to the inverter. The inverter responds with three process input data words (PI). The coding of the process data can be interpreted as follows in this example:

Request telegram from master to inverter:

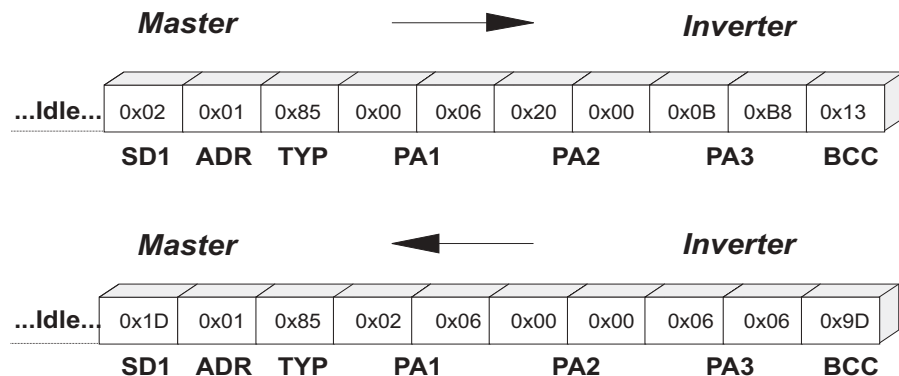
PO1: 0006_{hex} (e.g. control word 1 = enable)
 PO2: 2000_{hex} (e.g. speed [%] setpoint = 50 %)
 PO3: 0BB8_{hex} (e.g. ramp = 3 s)

Response telegram from inverter to master:

PI1: 0206_{hex} (e.g. status word 1)
 PI2: 0000_{hex} (e.g. speed [%] actual value = 0 %)
 PI3: 0606_{hex} (e.g. status word 2)

Structure of the request and response telegram

The following figure shows the structure of the request and response telegram.



01514BEN

Fig. 23: Structure of the request telegram with standard PDU types

This example shows the acyclical transmission, i.e. no timeout monitoring is active in the inverter. The cyclical transmission can be implemented by entering TYP = 5. In this case, the request telegrams must be sent by the master within the timeout delay of the inverter.



Please refer to the information in the "IPOS^{plus}® Positioning and Sequence Control System" manual for applications with IPOS^{plus}® as the master.

**PDU types**

List of PDU types supported by MOVIDRIVE® inverters:

PDU type		Name
00 _{hex}	0 _{dec}	PARAM + 1PD cyclical
01 _{hex}	1 _{dec}	1PD cyclical
02 _{hex}	2 _{dec}	PARAM + 2PD cyclical
03 _{hex}	3 _{dec}	2PD cyclical
04 _{hex}	4 _{dec}	PARAM + 3PD cyclical
05 _{hex}	5 _{dec}	3PD cyclical
06 _{hex}	6 _{dec}	PARAM + 0PD cyclical
80 _{hex}	128 _{dec}	PARAM + 1PD acyclical
81 _{hex}	129 _{dec}	1PD acyclical
82 _{hex}	130 _{dec}	PARAM + 2PD acyclical
83 _{hex}	131 _{dec}	2PD acyclical
84 _{hex}	132 _{dec}	PARAM + 3PD acyclical
85 _{hex}	133 _{dec}	3PD acyclical

All inverters support the READ parameter service in PDU type PARAM + 0PD acyclical (86_{hex}/134_{dec}).



5 System Bus (SBus)

5.1 Slave data exchange via MOVILINK®

Communication via MOVILINK® incorporates the data exchange of parameter and process data telegrams. The MOVIDRIVE® unit can communicate as master or slave in this process.

As master, the unit can actively initiate a data exchange of parameter and process data telegrams using an IPOS^{plus}® program with the MOVLNK command.

As slave, the unit can receive and answer parameter and process data telegrams via the SBus.

Various types of telegrams have been defined for communication with a master control. These types can be divided into three categories:

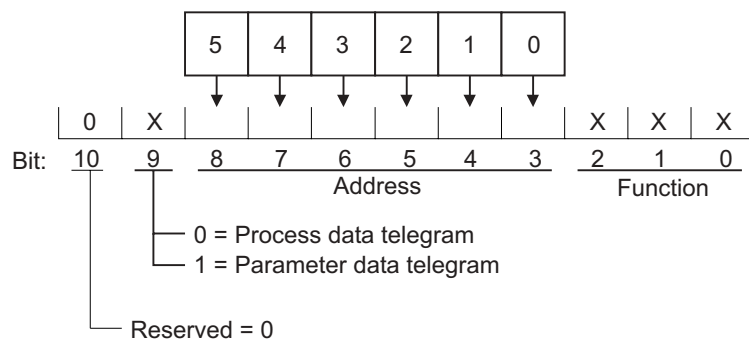
- Synchronization telegram
- Process data telegrams
- Parameter telegrams

CAN bus identifier

On the SBus, it is necessary to differentiate between these various types of telegram by means of the identifier (ID). As a result, the ID of an SBus telegram is made up of the type of telegram and the SBus address which is set using either parameter P813 (SBus address) or parameter P814 (SBus group address).

The CAN bus identifier is 11 bits in length because only standard identifiers are used. The 11 bits of the identifier are divided into three groups.

- Function (bits 0 – 2)
- Address (bits 3 – 8)
- Process data/parameter data switch (bit 9)



02250BEN

Fig. 24: CAN identifier for SBus via MOVILINK®

Bit 9 is used for differentiating between process data and parameter data telegrams. Bit 10 is reserved and must be 0. The address of parameter and process data telegrams contains the SBus address (P813) of the unit to which a request is being sent, while the address of group parameter and group process data telegrams contains the SBus group address (P814).



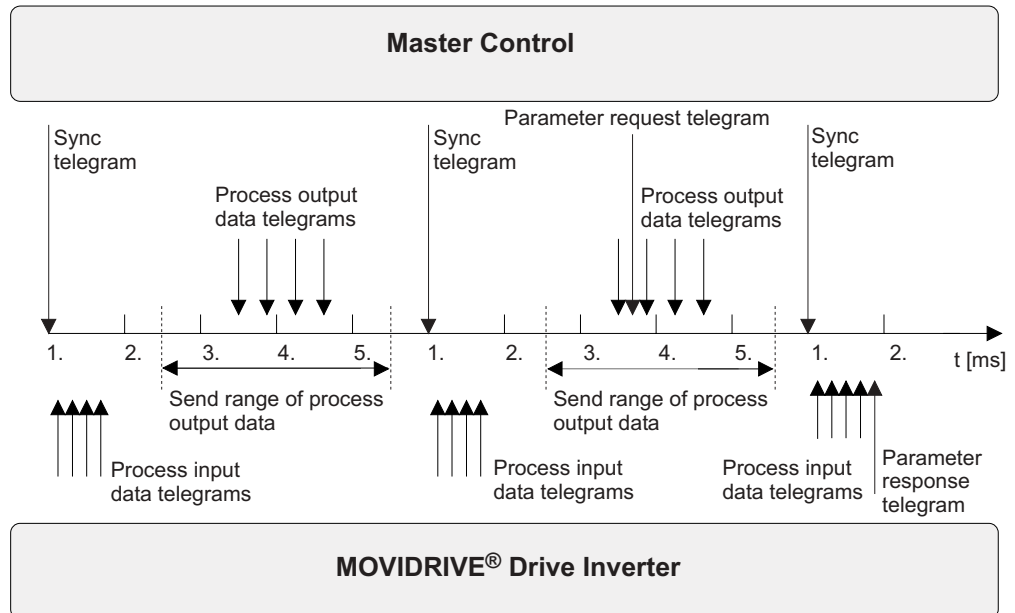
Creating the identifiers

The following table shows the relationship between the type of telegram and the address when the identifiers for SBus MOVILINK® telegrams are created:

Identifier	Telegram type
$8 \times \text{SBus address} + 3$	Process output data telegram (PO)
$8 \times \text{SBus address} + 4$	Process input data telegram (PI)
$8 \times \text{SBus address} + 5$	Process output data telegram synchronizable (PO-sync)
$8 \times \text{SBus group address} + 6$	Group process output data telegram (GPO)
$8 \times \text{SBus address} + 512 + 3$	Parameter request telegram
$8 \times \text{SBus address} + 512 + 4$	Parameter response telegram
$8 \times \text{SBus group address} + 512 + 6$	Group parameter request telegram

Synchronization telegram

A fixed time base of 5 milliseconds must be specified for transferring process data and parameter data. For this purpose, a synchronization telegram must be sent from the master control to the connected drive inverters in the first millisecond of a cycle.



01020BEN

Fig. 25: The bus time is divided into bus cycles

The synchronization telegram is a broadcast telegram. As a result, all drive inverters receive this telegram. The identifier of this telegram is set to zero in the factory setting. Any value between 0 and 2047 can be selected, but there must not be any overlap with the identifiers of the process or parameter data telegrams.



Process data telegrams

Process data telegrams are comprised of a process output and a process input data telegram. The process output data telegram is sent from the master to a slave; it contains the setpoint values for the slave. The process input data telegram is sent from the slave to the master and contains the actual values of the slave.

The setting for the number of process data items is fixed at the value "3 process data words."

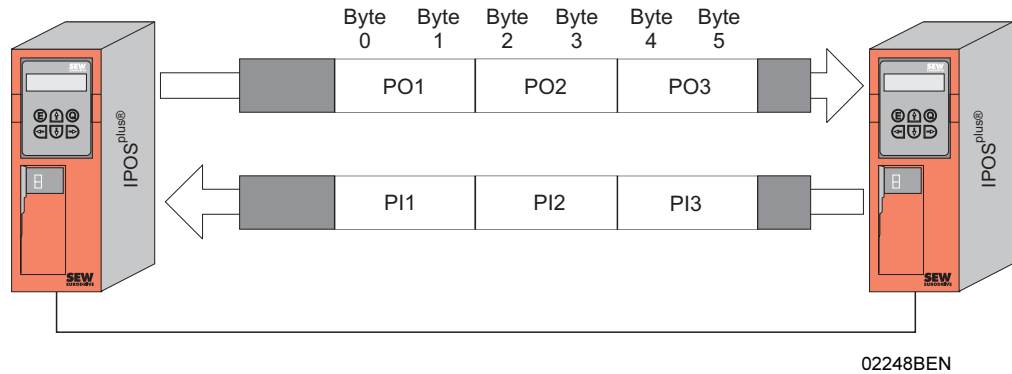


Fig. 26: Process data telegrams

The process data are sent at certain times within the fixed time base of five milliseconds. The system distinguishes between synchronous and asynchronous process data.

Synchronous process data are sent at specific times within the time base. As such, the process output data must be sent by the master control at the earliest 500 µs after the second millisecond and at the latest 500 µs before the first millisecond (→ Fig. 25). The process input data are sent by MOVIDRIVE® as a response in the first millisecond.

Asynchronous process data are not sent within the time base. The process output data can be sent by the master control at any time and they are answered by the MOVIDRIVE® unit within one millisecond by a process input data telegram.

Group parameter telegram

The group parameter telegram is sent from the master to one or more slaves with the same SBus group address. Its structure is the same as the parameter request telegram. This telegram can only be used for writing parameters to the slave units; the slaves do not respond to the telegram.

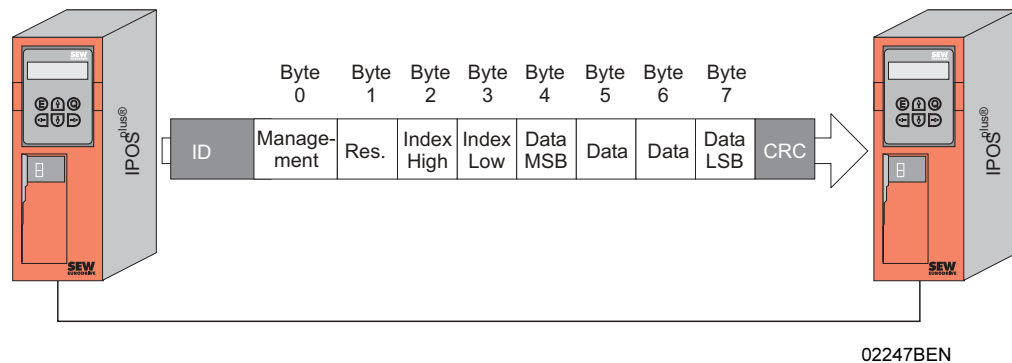


Fig. 27: Group parameter telegram



Parameter telegrams

Parameter telegrams are made up of a parameter request telegram and a parameter response telegram.

The parameter request telegram is sent by the master in order to read or write a parameter value. It consists of the:

- Management byte
- Index high byte
- Index low byte
- Four data bytes

The management byte specifies which service should be performed. The index specifies for which parameter the service should be performed and the four data bytes contain the numerical value which should be read or written (→ "Fieldbus Unit Profile" manual).

The parameter response telegram is sent by the slave in response to the parameter request telegram from the master. The request and response telegrams have the same structure.

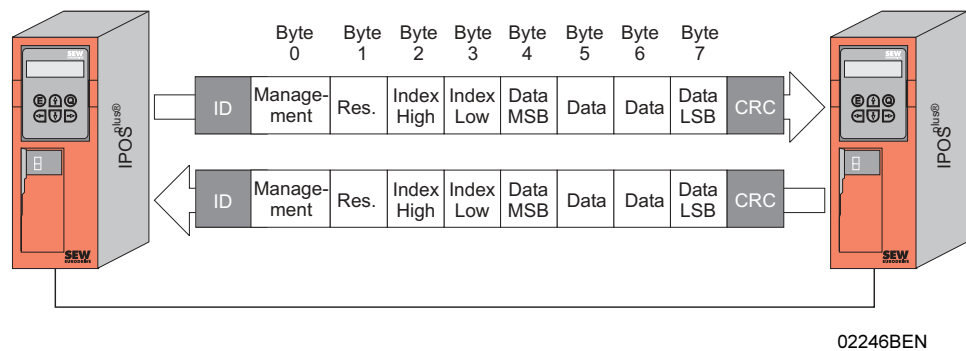


Fig. 28: Parameter telegrams

In the case of parameter telegrams, the system also distinguishes between synchronous and asynchronous telegrams. Synchronous parameter telegrams are answered within the time base of five milliseconds. The response telegram is sent in the first millisecond. Asynchronous parameter telegrams are answered regardless of the time base.

Group process data telegram

The group process data telegram is sent from the master to one or more slaves with the same SBus group address. Its structure is the same as the process output data telegram. This telegram can be used for sending the same setpoint values to several slaves which share the same SBus group address. The slaves do not respond to the telegram.

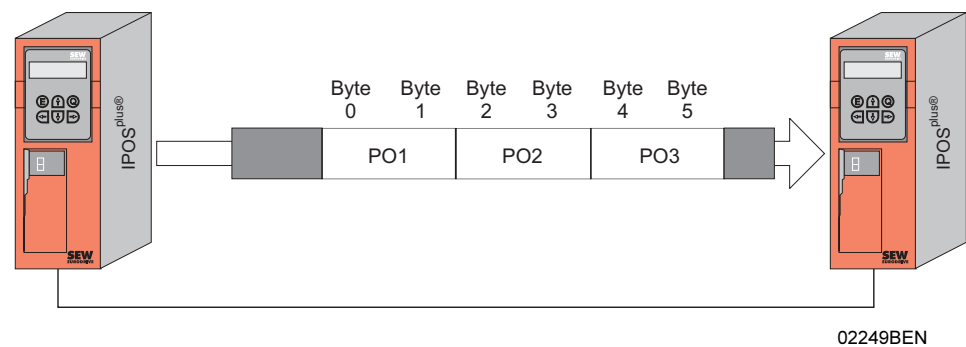


Fig. 29: Group process data telegram



Parameter settings

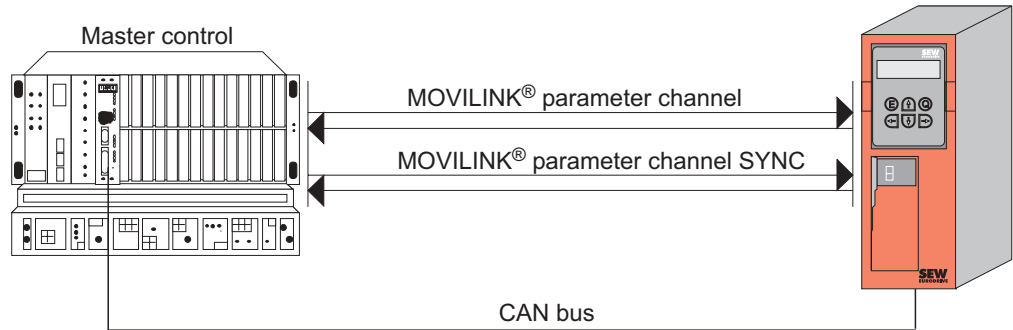
The following parameters have to be set for communication via the SBus:

Par.	Name	Setting	Meaning
100	Setpoint source	SBus	The inverter gets its setpoint from the SBus.
101	Control signal source	SBus	The inverter gets its control commands from the SBus.
813	SBus address	0 – 63	Setting the SBus address by which the parameter and process data are exchanged.
814	SBus group address	0 – 63	Setting the SBus group address by which the group parameter data and group process data can be received.
815	SBus timeout delay	0 – 650 s	Monitoring time for data transmission via the SBus. MOVIDRIVE® performs the error response set in P836 if there is no data traffic on the SBus within this time. Data transmission monitoring for the SBus is deactivated if P815 is set to 0 or 650 s.
816	SBus baud rate	125/250/500/1000 kbaud	The transmission speed of the SBus is set.
817	SBus synchronization ID	0 – 2047	A synchronization between the drives can take place for transmitting process data and parameter data via the SBus. To do this, the master control has to send a synchronization telegram to the connected inverters at specific intervals. P817 is used for setting the identifier (address) of the synchronization signal in the inverter for the SBus. Make sure there is no overlap between the identifiers for the process data or parameter data telegrams.
836	Response SBus TIMEOUT	Factory setting: EMERG.STOP/ ERROR	This programs the error response which is triggered by the system bus timeout monitoring.
870	Setpoint description PO1	Factory setting: CTRL. WORD 1	The content of the process output data words PO1/PO2/PO3 is defined. This is necessary so MOVIDRIVE® can allocate the appropriate setpoints.
871	Setpoint description PO2	SPEED	
872	Setpoint description PO3	NO FUNCTION	
873	Actual value description PI1	Factory setting: STATUS WORD1	The content of the process input data words PI1/PI2/PI3 is defined. This is necessary so MOVIDRIVE® can allocate the appropriate actual values. Furthermore, the process data must be enabled for the unit to adopt the setpoints.
874	Actual value description PI2	SPEED	
875	Actual value description PI3	NO FUNCTION	
876	PO data enable	ON	



5.2 Setting parameters via the CAN bus

The MOVIDRIVE® drive inverter supports the "MOVILINK® parameter channel" and the "MOVILINK® parameter channel-SYNC" with the SBus.



01025BEN

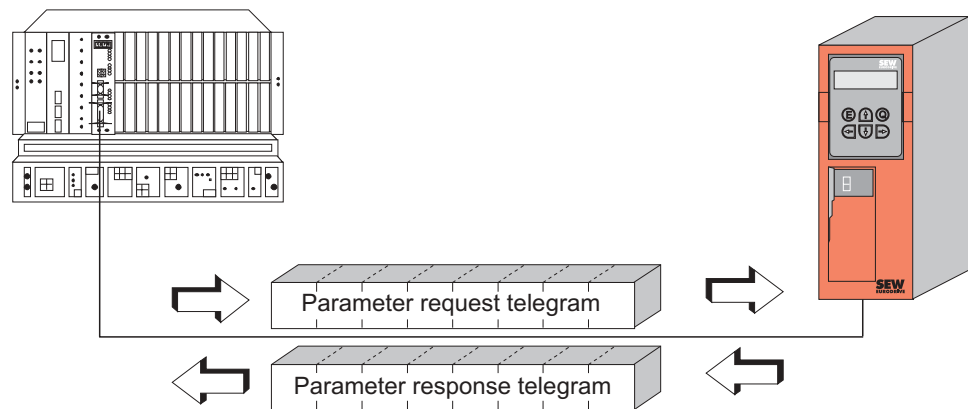
Fig. 30: Services via the CAN bus

Services

The drive inverter parameters are read and written via the SBus using the "MOVILINK® parameter channel" and "MOVILINK® parameter channel synchronized" services of the application layer (layer 7).

Structure of the parameter telegram

In order to set the parameters of peripheral units via fieldbus systems which do not provide an application layer, it is necessary to recreate the most important functions and services such as READ and WRITE for reading and writing parameters. A parameter telegram is defined for this purpose, e.g. for CAN. This parameter telegram is described by an identifier which is dependent on the set SBus synchronization ID. Parameter data can be exchanged via the parameter telegram (→ Fig. 31).



01026BEN

Fig. 31: Parameter telegram for CAN

The following table shows the structure of the parameter telegram. In principle, it is made up of a management byte, an index word, a reserved byte and four data bytes.

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Managmnt.	Reserved	Index high	Index low	Data MSB	Data	Data	Data LSB
Parameter list				4 byte data			

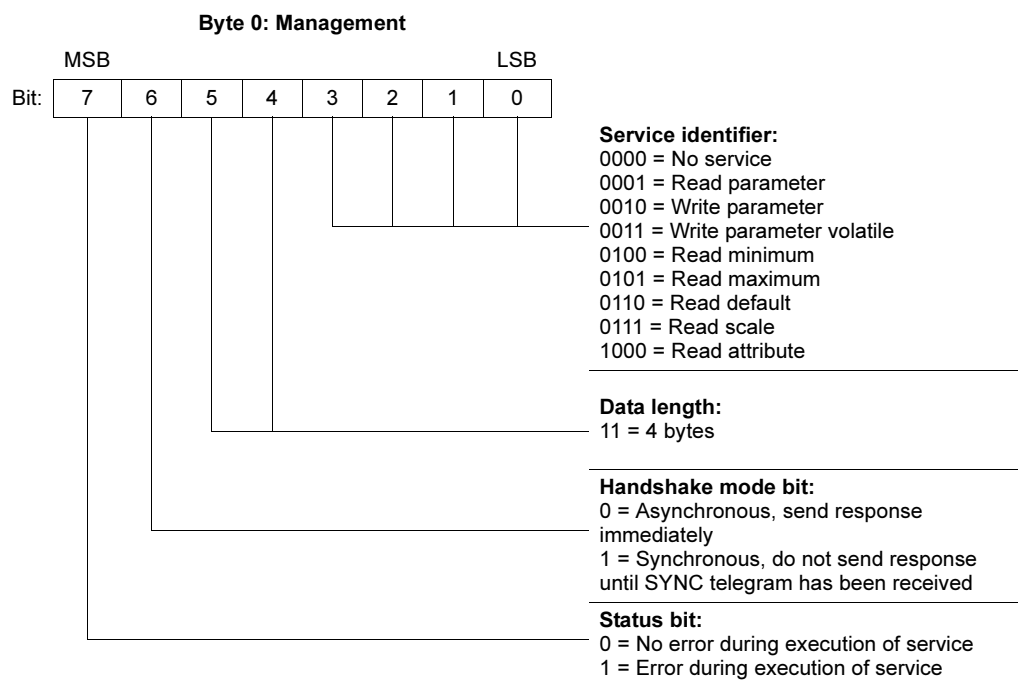


Management of the parameter telegram

The entire parameter setting sequence is coordinated with byte 0: "Management" byte. This byte is used for providing important service parameters such as service identifier, data length, version and status of the service performed. The following table shows that bits 0 – 3 contain the service identifier, and consequently they define which service is performed. Bit 4 and bit 5 specify the data length in bytes for the WRITE service. The data length should be set to 4 bytes for all SEW drive inverters.

- Handshake mode bit = 0: Asynchronous response to the synchronization telegram
- Handshake mode bit = 1: Synchronous response to the synchronization telegram in the first millisecond

Status bit 7 indicates whether it was possible to carry out the service properly or if there were errors.



Index addressing

Byte 2: Index high and byte 3: Index low determine the parameter which is to be read or written via the fieldbus system. The parameters of a drive inverter are addressed with a uniform index regardless of the connected fieldbus system.

Byte 1 should be viewed as reserved and must always be set to 0x00.

*Data range*

The data are located in byte 4 to byte 7 of the parameter telegram. This means up to 4 bytes of data can be transmitted per service. The data are always entered with right-justification, i.e. byte 7 contains the least significant data byte (data-LSB) whereas byte 4 is the most significant data byte (data-MSB).

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Managmnt.	Reserved	Index high	Index low	Data MSB	Data	Data	Data LSB
				High byte 1	Low byte 1	High byte 2	Low byte 2
				High word		Low word	
				Double word			

Incorrect performance of service

The status bit in the management byte is set to signal that a service has been performed incorrectly. If the status bit now indicates an error, the error code is entered in the data range of the parameter telegram. Bytes 4 – 7 send back the return code in a structured format.

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Managmnt.	Reserved	Index high	Index low	Error class	Error code	Add. code high	Add. code low



Status bit = 1: Incorrect performance of service

MOVILINK® parameter channel

The MOVILINK® parameter channel is described in detail in the "MOVIDRIVE® Fieldbus Unit Profile" manual.

It is important to distinguish between synchronized and non-synchronized parameters in connection with the CAN bus.

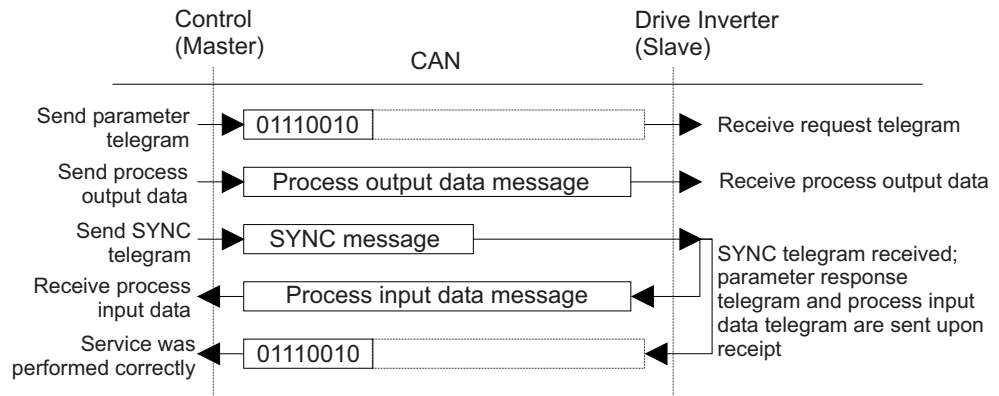
- With **non-synchronized** parameter telegrams, the acknowledgement is independent of the sync telegram.
- With **synchronized** parameter telegrams, the acknowledgement is not sent until after the sync telegram has been sent in the first millisecond.



Procedure for setting parameters with CAN

Taking the example of the WRITE-SYNC service, Fig. 32 represents a process of setting parameters between the control and the drive inverter via CAN. To simplify the sequence, Fig. 16 only shows the management byte of the parameter telegram.

While the control is preparing the parameter telegram for the WRITE-SYNC service, the drive inverter receives sync telegrams and receives and sends back process data telegrams. The service is activated after the parameter request telegram has been received. The drive inverter now interprets the parameter telegram and processes the WRITE-SYNC service. At the same time, it responds to all process data telegrams. The parameter response telegram is not sent until after the SYNC telegram has been received.



01028BEN

Fig. 32: Procedure for setting parameters with CAN

Parameter file format

When parameters are set via the SBus interface, the same parameter coding is used as in RS-232 or RS-485 serial interfaces.

The data formats and ranges of values for the individual parameters are listed in the "MOVIDRIVE® Fieldbus Unit Profile" manual.



Return codes for parameter setting

In the event of an incorrect parameter setting, the drive inverter sends back various return codes to the master which set the parameters. These codes provide detailed information about the cause of an error. All of these return codes are structured in accordance with DIN 19245 P2. The system distinguishes between the following elements:

- Error class
- Error code
- Additional code

These return codes are described in the "MOVIDRIVE® Fieldbus Unit Profile" manual.

Special cases

The fieldbus software describes parameter setting errors which cannot be identified either by layer 7 or the system software of the drive inverter.

This situation refers to the following special cases:

- **Errors in parameter settings**

An incorrect code was entered in the management byte during the implementation of a read or write service via the CAN bus.

	Code (dec)	Meaning
Error class:	5	Service
Error code:	5	Incorrect value
Add. code high:	0	-
Add. code low:	0	-



5.3 Master data exchange via MOVILINK®

MOVILINK® commands can be sent via IPOS^{plus}® in order to send parameter or process data telegrams to other stations. The command for implementing a parameter and/or process data exchange is MOVLNK.



The Movilink (...) command is not multimaster-capable. Only master is permitted within the whole network during the entire implementation time of a MOVLNK command! Use the IPOS^{plus}® SCOM command if you want to use a multimaster-capable protocol.

The Movilink (...) command has the variable H as its argument. This variable refers to a command structure. All information required for communication must be entered into this command structure.

The command structure contains several variables which select the interface defining the type of transmission and the data. The command structure is explained in the following table:

MOVLNK MI
_Movilink (...)

Variable no.	Name	Values	Meaning
H	Ml.BusType	0 = RESERVED 1 = S0 (RS-485 #1) 2 = S1 (RS-485 #2) 3 = RESERVED 4 = RESERVED 5 = SBus	This selects the interface to be used for transmitting a MOVILINK® command.
H + 1	Ml.Address	0 – 63 (SBus) 100 – 163 (SBus)	– SBus address is required for standard services. – 100 must be added to the SBus group address in order to calculate the group address.
H + 2	Ml.Format	0 = Param + 1PD 1 = 1PD 2 = Param + 2PD 3 = 2PD 4 = Param + 3PD 5 = 3PD 6 = Param (without PD)	Description of the telegram structure, e.g. Frametype = 4: Parameters and 3 process data items are transmitted with one MOVLNK command.
H + 3	Ml.Service	1 = Read 2 = Write 3 = Write without saving	– Read a parameter via parameter telegram. – Write with saving to non-volatile memory. – Write without saving.
H + 4	Ml.Index	Index number of a parameter	Index number of the parameter to be modified or read
H + 5	Ml.DPointer	Variable number	Number of the variable H' where the read data are stored or from where the data to be written are obtained.
H + 6	Ml.Result	Error code or 0	Contains the error code after implementation of the service, or contains zero if no error has occurred.

The DPointer points to a data structure which is explained in the following table:

MLDATA Mld

Variable no.	Name ¹⁾	Meaning
H'	Mld.Write Par	Contains the data for a parameter write service.
H' + 1	Mld.Read Par	Contains the data read by a parameter service.
H' + 2	Mld.PO1	First process output data word sent from the master unit to the slave unit.
H' + 3	Mld.PO2	Second process output data word sent from the master unit to the slave unit.
H' + 4	Mld.PO3	Third process output data word sent from the master unit to the slave unit.
H' + 5	Mld.PI1	First process input data word sent from the slave unit to the master unit.
H' + 6	Mld.PI2	Second process input data word sent from the slave unit to the master unit.
H' + 7	Mld.PI3	Third process input data word sent from the slave unit to the master unit.

1) The name is not displayed.


IPOS^{plus}® sample program

The command structure is initialized first. The following table shows a sample command structure:

MOVLNK MI

Name	Value	Meaning
MI.BusType	5	Use of the SBus
MI.Address	1	Slave address
MI.Format	4	Parameter data and 3 process data items
MI.Service	1	Read parameter data
MI.Index	8318	Index of the parameter: Speed
MI.DPointer	H20	Data structure starting with variable H20

The DPointer points to the data structure that can be initialized with the following values:

MOVLNK Mid

Name ¹⁾	Value	Meaning
Mid.WritePar	0	Can contain any values because a read service is performed
Mid.ReadPar	1000000	This value is sent in the parameter telegram from the slave and represents the speed for parameter 8318 (notation with 3 decimal places!)
Mid.PO1	6	The process output data word 1 is programmed to control word 1 on the slave. The slave is enabled by the value 6.
Mid.PO2	5000	The process output data word 2 is programmed to speed on the slave. The value 5000 specifies a speed of 1000 rpm for the slave (5000/5 rpm)
Mid.PO3	15000	The process output data word 3 is programmed to max. speed on the slave. The value 15000 specifies a maximum speed of 3000 rpm for the slave (15000/5 rpm)
Mid.PI1	7	The process input data word 1 is programmed to status word 1 on the slave. The slave sends its status with the value 7.
Mid.PI2	5011	The process input data word 2 is programmed to speed on the slave. The slave sends its actual speed with the value 5011. It is 1002.2 rpm (5011/5 rpm)
Mid.PI3	0	The process input data word 3 is programmed to no function on the slave. The value 0 is sent.

1) The name is not displayed.



IPOS^{plus}® program sequence

The screenshot shows the IPOSplus® COMPILER interface with a C program for MOVILINK data exchange. The program includes headers for constants and I/O, defines MOVILINK command and data structures, and implements a main function with initialization and a main loop. The Variables Watch window displays the current values of various MOVILINK-related variables.

Identifier	Value
H420 Ml.BusType	5
H421 Ml.Address	1
H422 Ml.Format	4
H423 Ml.Service	1
H424 Ml.Index	8318
H425 Ml.DPointer	427
H426 Ml.Result	0
H427 Mld.WritePar	0
H428 Mld.ReadPar	0
H429 Mld.PO1	6
H430 Mld.PO2	5000
H431 Mld.PO3	0
H432 Mld.PI1	514
H433 Mld.PI2	0
H434 Mld.PI3	0

Fig. 33: Master data exchange via MOVILINK®

05300AXX

The parameter response and process input data are updated after the Movilink (...) command has been performed, provided there was no error during the transfer. If an error did occur, this event is signaled in the return code by a value other than zero.



5.4 Master/slave operation via the SBus



Please also read the description of parameter group P75_ "Master-Slave function" in the MOVIDRIVE® System Manual for more information about master/slave operation.

Master/slave operation can be performed via the SBus. It involves the master sending the setpoint value and the control word to the slaves every millisecond using an SBus group telegram.

The master and the slaves must share the same SBus group address setting (P814). Valid addresses are in the range 0 – 63. Permitted baud rates for master/slave operation are 500 kbaud and 1000 kbaud (P816). Make sure that the baud rate settings on the master and the slaves are identical.

SBus group telegrams are used by the master for master/slave communication. Consequently, the master cannot be controlled by other stations using SBus group telegrams since this would disrupt arbitration on the CAN bus.

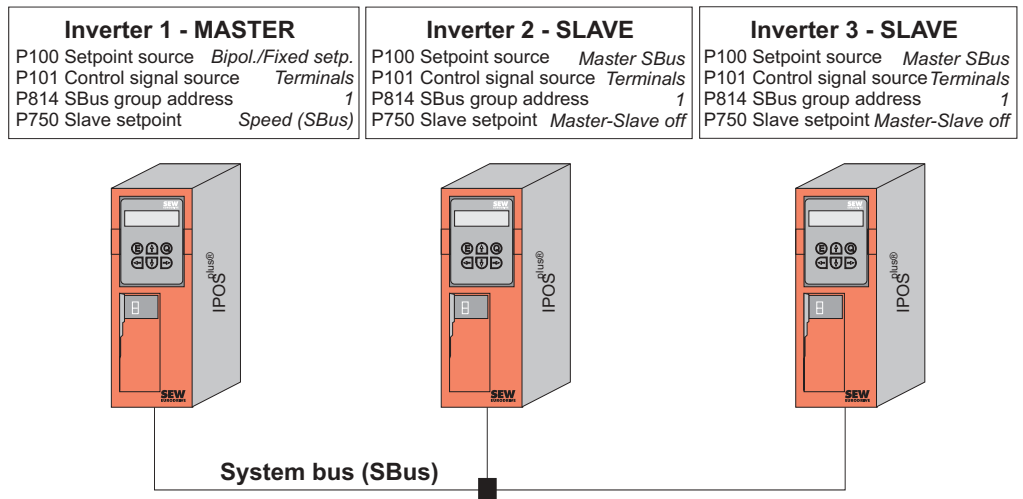


Fig. 34: Example of master/slave operation

02253BEN



5.5 Data exchange via variable telegrams

General information

Unit parameters and process data can be exchanged between units via MOVILINK®. Both the transfer frame and the identifiers are defined on the CAN bus.

Variable telegrams were introduced in order to create an open CAN bus interface. With the variable telegrams, there is a free choice of the identifier with which the telegrams are sent and the 8 bytes of data on the CAN bus are used for the content of two variables.

This provides an interface with direct access to layer 2 of the CAN bus. Consequently, the maximum processing speed is achieved for transmission of variables via the CAN bus.

The CAN bus is multimaster-capable which means every station can send a telegram. All bus stations always listen actively to see which telegrams are being sent on the bus. Each station filters out the telegrams which are important for itself and makes the data available to the application.

These characteristics make for an object-oriented approach. The stations send objects and those stations which want to process these objects receive them.



Each station can send and receive objects. However, it is important to note that each transmit object is only allowed to be sent by one station in order to avoid arbitration errors on the CAN bus.

CAN bus identifiers are reserved for the SBus MOVILINK® telegrams. The following rules apply:

1. A particular identifier may only be sent by one station. This means the identifiers used in the MOVILINK® protocol for sending telegrams are no longer available for the exchange of variables.
2. An SBus identifier may only be used once within a unit. This means the identifiers which are used for the SBus MOVILINK® protocol in a unit are no longer available for the transmission of variables.



**Example for
assignment of the
CAN identifiers**

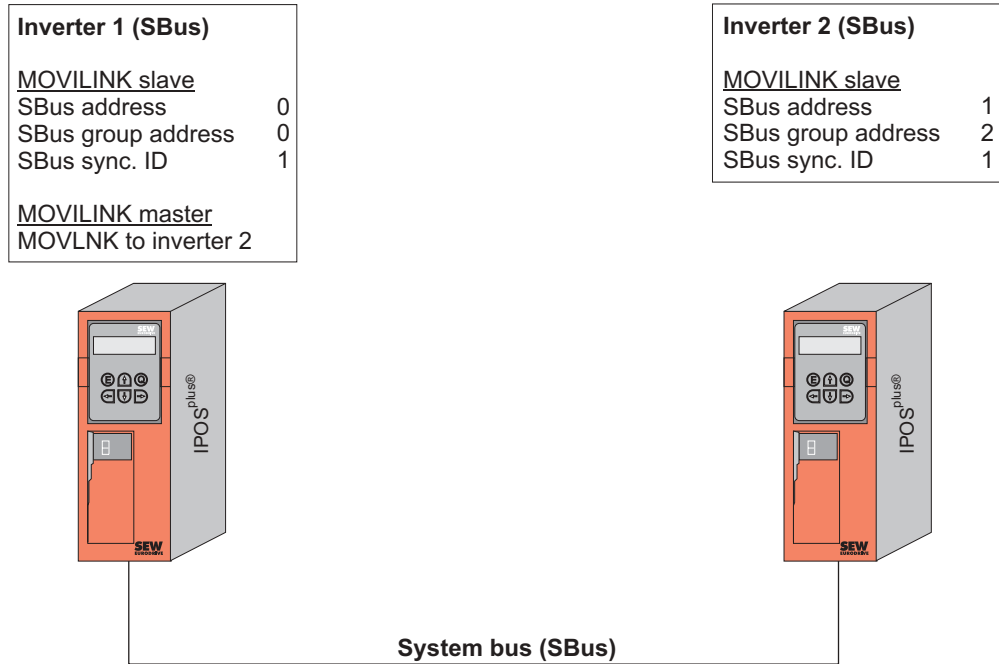


Fig. 35: Example for assignment of the CAN identifiers

02254BEN

The following table shows the assignment of the identifiers:

Type	Unit 1	ID	Unit 2	ID
SBus address	0		---	
Master/slave	Slave		---	
Parameter request ID	Receive	515	---	
Parameter response ID	Transmit	516	---	
PO async.	Receive	3	---	
PI	Transmit	4	---	
PO sync.	Receive	5	---	
SBus group address	0		---	
Master/slave	Slave		---	
Parameter request ID	Receive	518	---	
PO	Receive	6	---	
SBus address	1		1	
Master/slave	Master		Slave	
Parameter request ID	Transmit	523	Receive	523
Parameter response ID	Receive	524	Transmit	524
PO async.	Transmit	11	Receive	11
PI	Receive	12	Transmit	12
PO sync.	Transmit	13	Receive	13
SBus group address	2		2	
Master/slave	Master		Slave	
Parameter request ID	Transmit	534	Receive	534
PO	Transmit	22	Receive	22



The following relationship results from the previous table:

1. The identifiers shown on a gray background in the table (transmit identifiers) are no longer allowed to be used for variable transmit objects within this bus line.
2. The identifiers used within a unit (→ Unit column) are no longer allowed to be used for the variable objects (transmit and receive).

The following variable objects are not allowed to be used in the example (unit-specific!):

Unit no.	Transmit			Receive		
Unit 1	515	516	3	515	516	3
	4	5	518	4	5	518
	6	523	524	6	523	524
	11	12	13	11	12	13
	534	22	531	534	22	531
	532	19	29	532	19	29
	21			21		
Unit 2	523	524	11	523	524	11
	12	13	534	12	13	534
	22	516	4			

Two different IPOS^{plus}® commands are provided for variable transfer:

1. `_SBusCommDef (...)`: Transfer of variable telegrams
2. `_SBusCommOn ()`: Initialization and start of variable telegram transfer

The following services are available for the SCOM command:

1. `SCD_TRCYCL`: Cyclical sending of variable telegrams
2. `SCD_TRACYCL`: Acyclical sending of variable telegrams
3. `SCD_REC`: Receiving variable telegrams

These commands are described below.



Cyclical sending of variable telegrams

Command: `_SBusComDef (SCD_TRCYCL, TrCycle)`

(For a detailed description → MOVIDRIVE® IPOS^{plus}® manual)

This command sets up a cyclical variable service which sends a variable telegram with a set identifier at cyclical intervals.

Cyclical data transmission is started using the `_SBusCommOn ()` command and interrupted with a program stop.

The number of transmit objects which can be set up depends on the cycle time of the transmit objects:

Cycle time	Maximum number of transmit objects
1 – 9 ms	5
10 – 65530 ms	10

SCOM TRANSMIT CYCLIC contains a variable pointer as its second argument. The specified variable refers to a command structure.

SCTRCYCL TrCycl

Variable no.	Name	Value	Meaning
H	ObjectNo	0 – 2047	Describes the object number (CAN bus identifier)
H + 1	Cycletime	1 – 9 10 – 65530	Cycle time: 1 – 9 ms Step 1 ms 10 – 65530 ms Step 10 ms
H + 2	Offset	0 – 65534 0 – 65530	Time offset: 0 – 65534 ms Step 1 ms for cycle times < 10 ms 0 – 65530 ms Step 10 ms for cycle times ≥ 10 ms
H + 3	Len	0h – 8h 100h – 108h	Number of data bytes and data format. The length is specified in bits 0 – 3. The data format is specified in bit 8.
H + 4	DPointer	e.g. 20	The data structure starts with variable H20.
H + 5	Result	≥ 0 -1 -2 -3	Return code of initialization: <ul style="list-style-type: none"> • Free bus capacity in % • Incorrect cycle time • Too many objects set up • Bus overload

The DPointer refers to the data structure, in this case variable H20.

Variable no.	Name	Meaning
H'	long ITrDataLow	Contains the data of the first variable
H' + 1	long ITrDataHigh	Contains the data of the second variable

The second variable is used only with a set length of more than 4 bytes.



Receiving variable telegrams

Command: **_SBusComDef (SCD_REC, Rec)**

(For a detailed description → MOVIDRIVE® IPOS^{plus}® manual)

This command sets up a variable read service which receives a variable telegram with a set identifier.

Initialization of the objects is started using the `_SBusCommOn ()` command and interrupted with a program stop.

The number of receive objects which can be set up is restricted to 32 objects.

SCOM RECEIVE contains a variable pointer as its second argument. The specified variable refers to a command structure.

SCREC Rec

Variable no.	Name	Value	Meaning
H	ObjectNo	0 – 2047	Describes the object number (CAN bus identifier)
H + 1	Len	0h – 8h 100h – 108h	Number of data bytes and data format. The length is specified in bits 0 – 3. The data format is specified in bit 8.
H + 2	DPointer	e.g. 20	Data structure starting with variable H20.

The DPointer refers to the data structure, in this case variable H20.

Variable no.	Name	Meaning
H'	long IRecDataLow	Contains the data of the first variable
H' + 1	long IRecDataHigh	Contains the data of the second variable

The second variable is used only with a set length of more than 4 bytes.



Sending acyclical variable telegrams

Command: `_SBusComDef (SCD_TRACYCL, TrAcycl)`

(For a detailed description → MOVIDRIVE[®] IPOS^{plus}[®] manual)

This command sets up an acyclical variable write service which sends a variable telegram with a set identifier within an IPOS^{plus}[®] program cycle.

Initialization of the objects and sending of the object is undertaken immediately after the `SCD_TRACYCL` command has been carried out.

`SCD_TRACYCL` contains a variable pointer as its second argument. The specified variable refers to a command structure.

SCTRACYCL TrAcycl

Variable no.	Name	Value	Meaning
H10	ObjectNo.	0 – 2047	Describes the object number (CAN bus identifier)
H11	Len	0h – 8h 100h – 108h	Number of data bytes and data format. The length is specified in bits 0 – 3. The data format is specified in bit 8.
H12	DPointer	e.g. 20	Data structure starting with variable H20.
H13	Return code	0 1 2 3	Ready Transmission in progress Transmission successful Sending faulty

The DPointer refers to the data structure, in this case variable H20.

Variable no.	Name	Meaning
H20	long ITrAcyclDataLow	Contains the data of the first variable.
H21	long ITrAcyclDataHigh	Contains the data of the second variable.

The second variable is used only with a set length of more than 4 bytes.



Transfer formats and examples

MOTOROLA
format

Telegram 1:

The screenshot shows the IPOSplus COMPILER interface with the following components:

- Source Code (SCOM1.IPC):**

```

/*-----*/
      IPOS Source File
/*-----*/
#include <const.h>
#include <io.h>

// Transmit SCOM command structure
SCTRCYCL TrCycl;
// SCOM data
long lTrDataLow;
long lTrDataHigh;

// Receive SCOM command structure
SCREC Rec;
// SCOM data
long lReDataLow;
long lReDataHigh;
/*-----*/

      Main Function (IPOS Entry Function)
/*-----*/
main ()
{
  /*-----*/
      Initialisation
/*-----*/
  TrCycl.ObjectNo = 1024;
  TrCycl.CycleTime = 100;
  TrCycl.Offset = 0;
  TrCycl.Format = 0x8;
  TrCycl.DPointer = sizeof(lTrDataLow);
  lTrDataLow = 0x11223344;
  lTrDataHigh = 0x55667788;
  _SBusCmdDef( SCD_TRCYCL,TrCycl );
  Rec.ObjectNo = 1025;
  Rec.Format = 0x8;
  Rec.DPointer = sizeof(lReDataLow);
  _SBusCmdDef( SCD_REC,Rec );
  _SBusCommOn( );
  /*-----*/
      Main Loop
/*-----*/
  while (1)
  {
    // ...
  }
}

```
- Variables Watch:**

Identifier	Value
H420 TrCycl.ObjectNo	1024
H421 TrCycl.CycleTime	100
H422 TrCycl.Offset	0
H423 TrCycl.Format	00000008 h
H424 TrCycl.DPointer	426
H425 TrCycl.Result	99
H426 lTrDataLow	11223344 h
H427 lTrDataHigh	55667788 h
H428 Rec.ObjectNo	1025
H429 Rec.Format	00000008 h
H430 Rec.DPointer	431
H431 lReDataLow	44332211 h
H432 lReDataHigh	00776655 h
- Status Bar:** Online | ADDR: 0 | stored | 27:23

Fig. 36: Example of SCOM with MOTOROLA format, telegram 1

05301AXX



Telegram 2:

The screenshot shows the IPOSplus COMPILER interface with the following components:

- Source Code (SCOM2.IPC):**

```

IPOS Source File
-----+
#include <const.h>
#include <io.h>

// Transmit SCOM command structure
SCTRCYCL TrCycl;
// SCOM data
long lTrDataLow;
long lTrDataHigh;

// Receive SCOM command structure
SCREC Rec;
// SCOM data
long lReDataLow;
long lReDataHigh;
-----+

Main Function (IPOS Entry Function)
-----+
main()
{
    /*
    Initialisation
    -----+
    TrCycl.ObjectNo = 1025;
    TrCycl.CycleTime = 120;
    TrCycl.Offset = 0;
    TrCycl.Format = 0x8;
    TrCycl.DPointer = numof(lTrDataLow);
    lTrDataLow = 0x44332211;
    lTrDataHigh = 0x88776655;
    _SDusCommDef( SCD_TRCYCL, TrCycl );
    Rec.ObjectNo = 1024;
    Rec.Format = 0x8;
    Rec.DPointer = numof(lReDataLow);
    _SBusCommDef( SCD_REC, Rec );
    _SDusCommOn( );
    -----+

    Main Loop
    -----+
    while (1)
    {
        // ...
    }
}

```
- Variables Watch:**

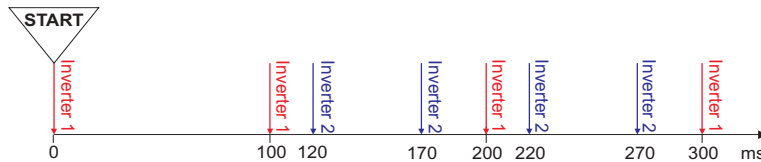
Identifier	Value
H420 TrCycl.ObjectNo	1025
H421 TrCycl.CycleTime	120
H422 TrCycl.Offset	0
H423 TrCycl.Format	00000008 h
H424 TrCycl.DPointer	426
H425 TrCycl.Result	99
H426 lTrDataLow	44332211 h
H427 lTrDataHigh	88776655 h
H428 Rec.ObjectNo	1024
H429 Rec.Format	00000008 h
H430 Rec.DPointer	431
H431 lReDataLow	11223344 h
H432 lReDataHigh	55667700 h

Fig. 37: Example of SCOM with MOTOROLA format, telegram 2

05387AXX



In the example, a variable telegram (= telegram 1) with a length of 8 bytes (i.e. 2 variables) and object number 1 is sent from unit 1 every 100 ms and received by unit 2. A second variable telegram (= telegram 2) with a length of 8 bytes (i.e. 2 variables) and object number 2 is sent from unit 2 every 50 ms with a time offset of 120 ms and received by unit 1.



02256BEN

Fig. 38: Time relationships with the SCOM command



The MOTOROLA format is characterized by the fact that the high byte is transmitted first and the low byte is transmitted last.

IPOS variables	
...	
20	11223344h
21	55667788h

CAN bus telegram									
Byte	0	1	2	3	4	5	6	7	CRC
Signific.	MSB			LSB	MSB			LSB	
Value	55h	66h	77h	88h	11h	22h	33h	44h	
Variable	H21 = DPointer + 1				H20 = DPointer				



INTEL format

Telegram 1:

The screenshot displays the IPOSplus@ COMPILER interface. The main window shows the source code for 'SCOM1_IntelIPC'. The code includes headers, defines transmit and receive structures, and a main function with initialization and a main loop. The 'Variables Watch' window on the right shows the current state of variables.

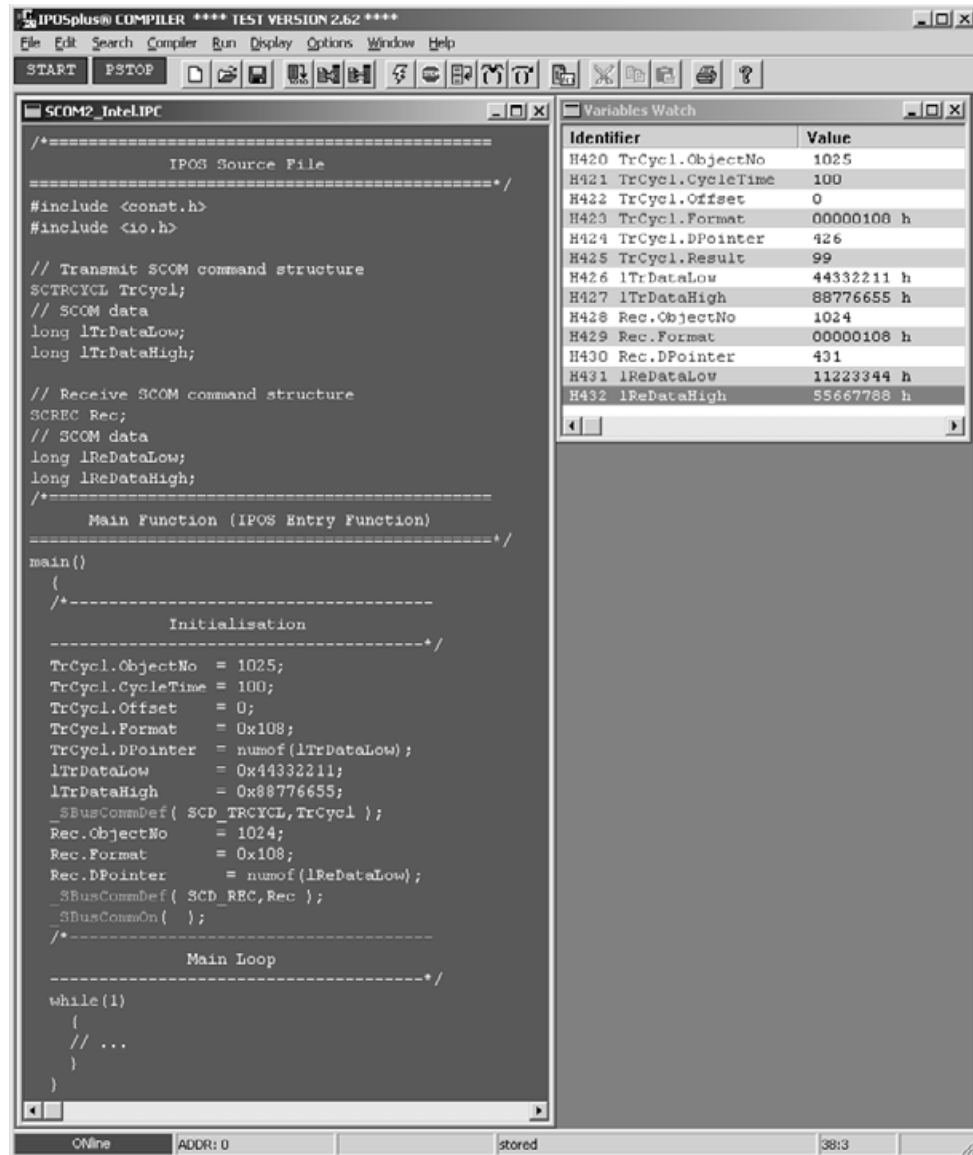
Identifier	Value
H420 TrCycl.ObjectNo	1024
H421 TrCycl.CycleTime	100
H422 TrCycl.Offset	0
H423 TrCycl.Format	00000108 h
H424 TrCycl.DPointer	426
H425 TrCycl.Result	99
H426 lTrDataLow	11223344 h
H427 lTrDataHigh	55667700 h
H428 Rec.ObjectNo	1025
H429 Rec.Format	00000108 h
H430 Rec.DPointer	431
H431 lReDataLow	44332211 h
H432 lReDataHigh	00776655 h

Fig. 39: Example of SCOM with INTEL format, telegram 1

02258BDE



Telegram 2:



05388AXX

Fig. 40: Example of SCOM with INTEL format, telegram 2

In the example, a variable telegram (= telegram 1) with a length of 8 bytes (i.e. 2 variables) and object number 1 is sent from unit 1 every 100 ms and received by unit 2. A second variable telegram with a length of 8 bytes (i.e. 2 variables) and object number 2 is sent from unit 2 every 100 ms and received by unit 1.



The INTEL format is characterized by the fact that the low byte is transmitted first and the high byte is transmitted last.

IPOS variables	
...	
20	11223344h
21	55667788h

		CAN bus telegram								
Byte	Identifier	0	1	2	3	4	5	6	7	CRC
Signific.		LSB			MSB	LSB			MSB	
Value		44h	33h	22h	11h	88h	77h	66h	55h	
Variable		H21 = DPointer				H20 = DPointer + 1				

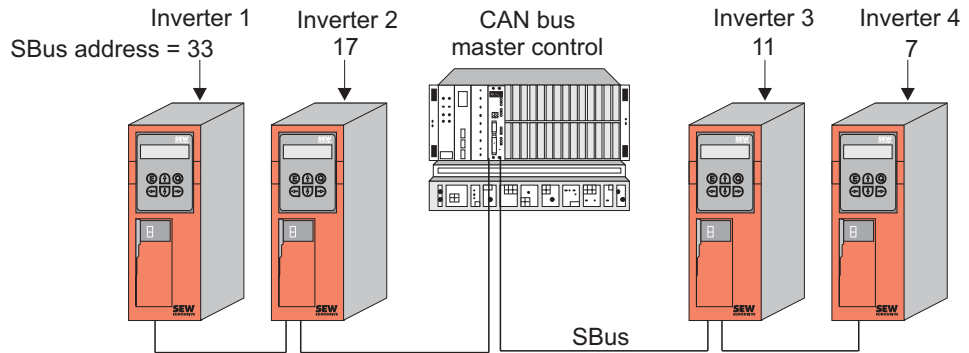


5.6 Project planning example for SBus

Settings

The following settings have to be made:

- Number of drive inverters: 4
- Process data length: 3
- Baud rate: 500 kbit/s



02260BEN

Fig. 41: Sample for project planning

Settings

Use parameter P816 to set the required SBus baud rate on all drive inverters. The same baud rate must be set on all drive inverters. The factory setting for the baud rate is 500 kbaud. The setting for the number of process data items on the SBus is fixed at the value "3 process data words."

Then use parameter P813 to set the SBus address on each drive inverter. In this example, the SBus address is assigned to the drive inverters in accordance with the following table:

Drive inverter	SBus address
1	33 _{dec}
2	17 _{dec}
3	11 _{dec}
4	7 _{dec}

You can see during this process that there is no need to follow a particular sequence when setting the SBus address. However, the addresses cannot be assigned several times, i.e. two drive inverters must not share the same SBus address.

In addition, the terminating resistor must also be switched on at the cable ends. Do this by switching on the S12 DIP switches on inverters 1 and 4. The terminating resistors must be switched off on all other stations, e.g. inverters 2 and 3 and the CAN bus master control.



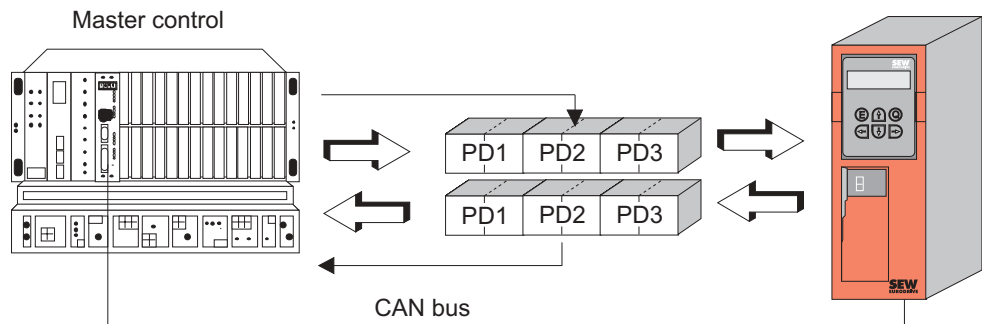
IDs on the CAN bus

In this combination, the IDs listed below are occupied on the CAN bus:

Drive inverter	SBus address	ID	Telegram type
1	33 _{dec}	267	Process output data telegram (PO)
		268	Process input data telegram (PI)
		269	Process output data telegram synchronized (PO-sync)
		779	Parameter request telegram
		780	Parameter response telegram
2	17 _{dec}	139	Process output data telegram (PO)
		140	Process input data telegram (PI)
		141	Process output data telegram synchronized (PO-sync)
		651	Parameter request telegram
		652	Parameter response telegram
3	11 _{dec}	91	Process output data telegram (PO)
		92	Process input data telegram (PI)
		93	Process output data telegram synchronized (PO-sync)
		603	Parameter request telegram
		604	Parameter response telegram
4	7 _{dec}	59	Process output data telegram (PO)
		60	Process input data telegram (PI)
		61	Process output data telegram synchronized (PO-sync)
		571	Parameter request telegram
		572	Parameter response telegram

The IDs are calculated on the basis of the set SBus address (P813) and the offset which describes the corresponding telegram.

Process data length = 3 means that the drive inverter receives exactly three process output data words and sends three process input data words to the master control.



01024BEN

Fig. 42: Programming a process data word

The content of the process data words is defined using the process output data description parameters 1 – 3 and the process input data description parameters 1 – 3.



6 Operation and Service

6.1 Startup problems with the SBus

- **No communication on the SBus: Timeout or no response**
 1. SBus cable incorrect or not connected.
 2. System bus high and system bus low reversed.
 3. Not all stations are communicating at the same baud rate: Setting with parameter P816.
 4. Terminating resistors were not switched on or not switched on correctly (S12).
 5. Collisions on the SBus because several stations are sending a transmit telegram with the same ID.
 6. SBus cable too long (max. 80 m at 500 kbaud).
 7. Incorrect setting of SBus address or SBus group address.

- **Unit error F47: TIMEOUT SBUS**
 1. The unit is enabled and no process data are received via the SBus. → **Solution:** Send process data.
 2. The unit is enabled and no process data are received via the SBus within the SBus timeout time, and the SBus timeout response has been programmed for a response with malfunction. → **Solution:** Increase SBus timeout delay.
 3. → No communication on the SBus.

- **The MOVILINK[®] transfer is being disrupted by the exchange of variables via IPOS^{plus}[®].**
 Identifiers used for the variable transfer have already been used for the MOVILINK[®] transfer.

- **Some variable objects cannot be sent or received.**
 Identifiers used for the variable transfer have already been used for the MOVILINK[®] transfer.

- **The data content of the variable objects is represented incorrectly.**
 Check the data format. Note the difference between the MOTOROLA and the INTEL formats.

- **Acyclical variable transfer does function although cyclical variable transfer does not.**
 The cyclical variable objects and the receive objects must be started with the SCOMON command.

- **Unit cannot be enabled and displays status "t."**
 The setpoint source and/or the control signal source are set to SBus and no process values are being received via the SBus. Either set the SBus timeout delay (P815) to 0 (this deactivates timeout monitoring) or send process data via the SBus.



6.2 Return codes for parameter setting

In the event of an incorrect parameter setting, the drive inverter sends back various return codes to the master which set the parameters. These codes provide detailed information about the cause of the error. All of these return codes are structured in accordance with EN 50170. The system distinguishes between the following elements:

- Error class
- Error code
- Additional code

These return codes apply to all MOVIDRIVE® communications interfaces.

Error class

The error class element classifies the type of error more precisely. EN 50170 defines the following different error classes.

Class (hex)	Name	Meaning
1	vfd-state	Status error of the virtual field unit
2	application-reference	Error in application program
3	definition	Definition error
4	resource	Resource error
5	service	Error when performing service
6	access	Access error
7	ov	Error in object list
8	other	Other error (see additional code)

The error class is generated by the communications software of the fieldbus interface in case there is an error in communication. This does not apply to "Error class 8 = Other error." Return codes sent from the drive inverter system are all in "Error class 8 = Other error." The error can be more precisely identified using the additional code element.

Error code

The error code element provides a means for more precisely identifying the cause of the error within the error class. It is generated by the communications software of the fieldbus card in the event of an error in communication. Only error code 0 ("Other error code") is defined for "Error class 8 = Other error." In this case, detailed identification is available by using the additional code.

**Additional code**

The additional code contains the SEW-specific return codes dealing with incorrect parameter settings of the drive inverter. They are sent back to the master in "Error class 8 = Other error." The following table shows all possible codings for the additional code.

Add. code high (hex)	Add. code low (hex)	Meaning
00	00	No error
00	10	Invalid parameter index
00	11	Function/parameter not implemented
00	12	Rread access only
00	13	Parameter lock is active
00	14	Factory setting is active
00	15	Parameter value too large
00	16	Parameter value too small
00	17	Option card required for this function/parameter is missing
00	18	Error in system software
00	19	Parameter access via RS-485 process interface on X13 only
00	1A	Parameter access via RS-485 diagnostic interface only
00	1B	Parameter has access protection
00	1C	Controller inhibit required
00	1D	Incorrect parameter value
00	1E	Factory setting was activated
00	1F	Parameter was not saved in EEPROM
00	20	Parameter cannot be changed with enabled output stage

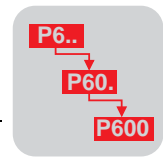
"Internal communications error" special case

The return code listed in the following table is sent back in case a communications error has occurred between the option card and the inverter system. The parameter adjustment service transferred via the fieldbus may not have been performed and should be repeated. If this error reoccurs, it is necessary to switch off the drive inverter completely and then back on again so it is re-initialized.

	Code (dec)	Meaning
Error class:	6	Access
Error code:	2	Hardware error
Add. code high:	0	-
Add. code low:	0	-

Correcting the error

Repeat the read or write service. If this error occurs again, switch the drive inverter off completely and back on again. Contact the SEW Electronics Service for advice if this error occurs continuously.



7 Parameter List

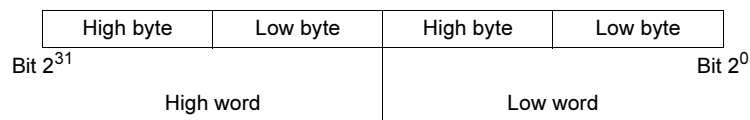
7.1 Explanation of the table header

The meanings of the entries in the table header are as follows:

Par. no.	= Parameter number used in MOVITOOLS/SHELL or the DBG11A.	
Parameter	= Parameter name	
Index	= 16-bit index for addressing the parameter via interfaces. Notation in decimal (= dec) and hexadecimal (= hex) format.	
Unit/index	= Unit index according to the sensor/actuator profile of the PNO.	Abbr. = Abbreviation of the unit of measurement Sz. = Size index (e.g. 11 = Speed) Cv. = Conversion index (e.g. -3 = 10 ⁻³)
Access	= Access attributes	S = Store even with parameter lock RO = Read only R = Controller inhibit must be active when writing RW = Read/write N = The value is written from the EEPROM into the RAM during a restart
Default	= Factory setting	
Meaning/ value range	= Meaning and value range of the parameter	

Data format

All parameters are treated as 32-bit values. Representation in Motorola format.



MOVILINK[®] parameters

The parameters are arranged so they are in the proprietary area of the drive profiles (DRIVECOM-INTERBUS, CANopen, etc.). Consequently, the area for the indices of the MOVILINK[®] parameters is as follows:

$$2000_{\text{hex}} (= 8192_{\text{dec}}) - 5FFF_{\text{hex}} (= 24575_{\text{dec}})$$

Start index	Number of indices	Contents
8300	700	Drive parameters / Display values / Scope parameters
10000	100	Error responses (max. 255 error codes)
10300	40	Motor table current (ID)
10600	40	Motor table flux
11000	512	IPOS variables (11000 + IPOS variable number)
16000	2048	IPOS program code
20000	513	Curve points for the electronic cam
24575	-	End

The Ctrl + F1 key combination calls up the index number of every selected parameter in MOVITOOLS/SHELL.



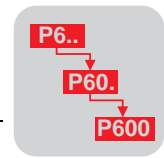
7.2 Complete parameter list, sorted by parameter numbers



Note that the following parameter list applies to MOVIDRIVE® MD_60A and MOVIDRIVE® compact drive inverters.

- Parameters which are valid for MOVIDRIVE® MD_60A only are indicated by • after the parameter number (Par. no.).

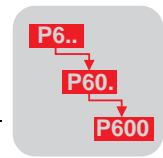
Par. no.	Parameter	Index		Unit/index			Access	Default	Meaning / value range
		Dec	Hex	Abbr.	Sz.	Cv.			
0.. Display values									
00. Process values									
000	Speed [rpm]	8318	207E	rps	11	66	RO	0	
001	User display []	8501	2135		0	-3	RO	0	
002	Frequency [Hz]	8319	207F	Hz	28	-3	RO	0	
003	Actual position [Inc]	8320	2080		0	0	RO	0	
004	Output current [%In]	8321	2081	%	24	-3	RO	0	
005	Active current [%In]	8322	2082	%	24	-3	RO	0	
006	Motor utilization 1 [%]	8323	2083	%	24	-3	N/RO	0	
007	Motor utilization 2 [%]	8324	2084	%	24	-3	N/RO	0	
008	DC link voltage [V]	8325	2085	V	21	-3	RO	0	
009	Output current [A]	8326	2086	A	22	-3	RO	0	
01. Status displays									
010	Inverter status	8310	2076		0	0	RO	0	Low word coded as status word 1
011	Operational status	8310	2076		0	0	RO	0	
012	Error status	8310	2076		0	0	RO	0	
013	Active parameter set	8310	2076		0	0	RO	0	
014	Heat sink temperature [°C]	8327	2087	°C	17	100	RO	0	
015	Mains ON operation time [h]	8328	2088	s	4	70	N/RO	0	
016	Operating time (enabled) [h]	8329	2089	s	4	70	N/RO	0	
017	Electrical energy [kWh]	8330	208A	Ws	8	5	N/RO	0	
02. Analog setpoints									
020	Analog input AI1 [V]	8331	208B	V	21	-3	RO	0	
021	Analog input AI2 [V]	8332	208C	V	21	-3	RO	0	
022	External current limit [%]	8333	208D	%	24	-3	RO	0	
03. Binary inputs basic unit									
030	Binary input DI00	8334	208E		0	0	R/RO	0	
031	Binary input DI01	8335	208F		0	0	N/R/RW	2	0 – 25, step 1
032	Binary input DI02	8336	2090		0	0	N/R/RW	3	0 – 25, step 1
033	Binary input DI03	8337	2091		0	0	N/R/RW	1	0 – 25, step 1
034	Binary input DI04	8338	2092		0	0	N/R/RW	4	0 – 25, step 1
035	Binary input DI05	8339	2093		0	0	N/R/RW	5	0 – 25, step 1
036	Binary inputs DI00..DI05	8334	208E		0	0	N/R/RW	0	Bit 0 = DI00 – Bit 5 = DI05
04. Binary inputs option									
040	Binary input DI10	8340	2094		0	0	N/R/RW	0	0 – 25, step 1
041	Binary input DI11	8341	2095		0	0	N/R/RW	0	0 – 25, step 1
042	Binary input DI12	8342	2096		0	0	N/R/RW	0	0 – 25, step 1
043	Binary input DI13	8343	2097		0	0	N/R/RW	0	0 – 25, step 1
044	Binary input DI14	8344	2098		0	0	N/R/RW	0	0 – 25, step 1
045	Binary input DI15	8345	2099		0	0	N/R/RW	0	0 – 25, step 1
046	Binary input DI16	8346	209A		0	0	N/R/RW	0	0 – 25, step 1
047	Binary input DI17	8347	209B		0	0	N/R/RW	0	0 – 25, step 1
048	Binary inputs DI10..DI17	8348	209C		0	0	RO	0	Bit 0 = DI10 – Bit 7 = DI17
05. Binary outputs basic unit									
050	Binary output DB00	8349	209D		0	0	RO	0	
051	Binary output DO01	8350	209E		0	0	N/RW	2	0 – 22, step 1
052	Binary output DO02	8351	209F		0	0	N/RW	1	0 – 22, step 1
053	Binary outputs DB00, DO01, DO02	8349	209D		0	0	RO	0	Bit 0 = DB00, Bit 1 = DO01, Bit 2 = DO02



Par. no.	Parameter	Index		Unit/index			Access	Default	Meaning / value range
		Dec	Hex	Abbr.	Sz.	Cv.			
06. Binary outputs option									
060	Binary output DO10	8352	20A0		0	0	N/RW	0	0 – 22, step 1
061	Binary output DO11	8353	20A1		0	0	N/RW	0	0 – 22, step 1
062	Binary output DO12	8354	20A2		0	0	N/RW	0	0 – 22, step 1
063	Binary output DO13	8355	20A3		0	0	N/RW	0	0 – 22, step 1
064	Binary output DO14	8356	20A4		0	0	N/RW	0	0 – 22, step 1
065	Binary output DO15	8357	20A5		0	0	N/RW	0	0 – 22, step 1
066	Binary output DO16	8358	20A6		0	0	N/RW	0	0 – 22, step 1
067	Binary output DO17	8359	20A7		0	0	N/RW	0	0 – 22, step 1
068	Binary outputs DO10..DO17	8360	20A8		0	0	RO	0	Bit 0 = DO10 – Bit 7 = DO17
07. Unit data									
070	Unit type	8301	206D		0	0	RO	0	
071	Unit rated current [A]	8361	20A9	A	22	-3	RO	0	
072	Option 1	8362	20AA		0	0	RO	0	0 = SHORT CIRCUIT 1 = Invalid 2 = FIELDBUS 3 = DPI/DPA 4 = DRS 5 = AIO 6 = Invalid 7 = DIO 8 = Invalid 9 = NONE
073	Option 2	8363	20AB		0	0	RO	0	See menu no. 072 or index 8362
074	Firmware option 1	8364	20AC		0	0	RO	0	Example:
075	Firmware option 2	8365	20AD		0	0	RO	0	822609711 = 822 609 7.11
076	Firmware basic unit	8300	206C		0	0	RO	0	1822609011 = 822 609 X.11
077	Technology function	8878	22AE		0	0	RW	0	0 = Standard 1 = Electronic cam 2 = ISync 3 = Auto ASR
08. Error memory time t-x									
080	Error t-0	8366	20AE		0	0	N/RO	0	
	Input terminals 1..6	8371	20B3		0	0	N/RO	0	Bit 0 = DI00 – Bit 5 = DI05
	Input terminals (opt.) 1..8	8376	20B8		0	0	N/RO	0	Bit 0 = DI10 – Bit 7 = DI17
	Output terminals 1..3	8381	20BD		0	0	N/RO	0	Bit 0 = DB00, Bit 1 = DO01, Bit 2 = DO02
	Output term. (optional) 1..8	8386	20C2		0	0	N/RO	0	Bit 0 = DO10 – Bit 7 = DO17
	Operational status	8391	20C7		0	0	N/RO	0	Low word coded as status word 1
	Heat sink temperature [°C]	8396	20CC	°C	17	100	N/RO	0	
	Speed [rpm]	8401	20D1	rps	11	66	N/RO	0	
	Output current [%]	8406	20D6	%	24	-3	N/RO	0	
	Active current [%]	8411	20DB	%	24	-3	N/RO	0	
	Unit utilization [%]	8416	20E0	%	24	-3	N/RO	0	
	DC link voltage [V]	8421	20E5	V	21	-3	N/RO	0	
	Mains ON operation time [h]	8426	20EA	s	4	70	N/RO	0	
	Operating time (enabled) [h]	8431	20EF	s	4	70	N/RO	0	
	Parameter set	8391	20C7		0	0	N/RO	0	
	Motor utilization 1 [%]	8441	20F9	%	24	-3	N/RO	0	
	Motor utilization 2 [%]	8446	20FE	%	24	-3	N/RO	0	



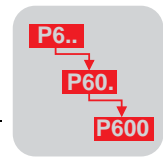
Par. no.	Parameter	Index		Unit/index			Access	Default	Meaning / value range	
		Dec	Hex	Abbr.	Sz.	Cv.				
081	Error t-1		8367	20AF		0	0	N/RO	0	
	Input terminals 1..6		8372	20B4		0	0	N/RO	0	Bit 0 = DI00 – Bit 5 = DI05
	Input terminals (opt.) 1..8		8377	20B9		0	0	N/RO	0	Bit 0 = DI10 – Bit 7 = DI17
	Output terminals 1..3		8382	20BE		0	0	N/RO	0	Bit 0 = DB00, Bit 1 = DO01, Bit 2 = DO02
	Output term. (optional) 1..8		8387	20C3		0	0	N/RO	0	Bit 0 = DO10 – Bit 7 = DO17
	Operational status		8392	20C8		0	0	N/RO	0	Low word coded as status word 1
	Heat sink temperature [°C]		8397	20CD	°C	17	100	N/RO	0	
	Speed [rpm]		8402	20D2	rps	11	66	N/RO	0	
	Output current [%]		8407	20D7	%	24	-3	N/RO	0	
	Active current [%]		8412	20DC	%	24	-3	N/RO	0	
	Unit utilization [%]		8417	20E1	%	24	-3	N/RO	0	
	DC link voltage [V]		8422	20E6	V	21	-3	N/RO	0	
	Mains ON operation time [h]		8427	20EB	s	4	70	N/RO	0	
	Operating time (enabled) [h]		8432	20F0	s	4	70	N/RO	0	
	Parameter set		8392	20C8		0	0	N/RO	0	
	Motor utilization 1 [%]		8442	20FA	%	24	-3	N/RO	0	
Motor utilization 2 [%]		8447	20FF	%	24	-3	N/RO	0		
082	Error t-2		8368	20B0		0	0	N/RO	0	
	Input terminals 1..6		8373	20B5		0	0	N/RO	0	Bit 0 = DI00 – Bit 5 = DI05
	Input terminals 1..8		8378	20BA		0	0	N/RO	0	Bit 0 = DI10 – Bit 7 = DI17
	Output terminals 1..3		8383	20BF		0	0	N/RO	0	Bit 0 = DB00, Bit 1 = DO01, Bit 2 = DO02
	Output term. (optional) 1..8		8388	20C4		0	0	N/RO	0	Bit 0 = DO10 – Bit 7 = DO17
	Operational status		8393	20C9		0	0	N/RO	0	Low word coded as status word 1
	Heat sink temperature [°C]		8398	20CE	°C	17	100	N/RO	0	
	Speed [rpm]		8403	20D3	rps	11	66	N/RO	0	
	Output current [%]		8408	20D8	%	24	-3	N/RO	0	
	Active current [%]		8413	20DD	%	24	-3	N/RO	0	
	Unit utilization [%]		8418	20E2	%	24	-3	N/RO	0	
	DC link voltage [V]		8423	20E7	V	21	-3	N/RO	0	
	Mains ON operation time [h]		8428	20EC	s	4	70	N/RO	0	
	Operating time (enabled) [h]		8433	20F1	s	4	70	N/RO	0	
	Parameter set		8393	20C9		0	0	N/RO	0	
	Motor utilization 1 [%]		8443	20FB	%	24	-3	N/RO	0	
Motor utilization 2 [%]		8448	2100	%	24	-3	N/RO	0		
083	Error t-3		8369	20B1		0	0	N/RO	0	
	Input terminals 1..6		8374	20B6		0	0	N/RO	0	Bit 0 = DI00 – Bit 5 = DI05
	Input terminals (opt.) 1..8		8379	20BB		0	0	N/RO	0	Bit 0 = DI10 – Bit 7 = DI17
	Output terminals 1..3		8384	20C0		0	0	N/RO	0	Bit 0 = DB00, Bit 1 = DO01, Bit 2 = DO02
	Output terminals 1..8		8389	20C5		0	0	N/RO	0	Bit 0 = DO10 – Bit 7 = DO17
	Operational status		8394	20CA		0	0	N/RO	0	Low word coded as status word 1
	Heat sink temperature [°C]		8399	20CF	°C	17	100	N/RO	0	
	Speed [rpm]		8404	20D4	rps	11	66	N/RO	0	
	Output current [%]		8409	20D9	%	24	-3	N/RO	0	
	Active current [%]		8414	20DE	%	24	-3	N/RO	0	
	Unit utilization [%]		8419	20E3	%	24	-3	N/RO	0	
	DC link voltage [V]		8424	20E8	V	21	-3	N/RO	0	
	Mains ON operation time[h]		8429	20ED	s	4	70	N/RO	0	
	Operating time (enabled)[h]		8434	20F2	s	4	70	N/RO	0	
	Parameter set		8394	20CA		0	0	N/RO	0	
	Motor utilization 1 [%]		8444	20FC	%	24	-3	N/RO	0	
Motor utilization 2 [%]		8449	2101	%	24	-3	N/RO	0		



Par. no.	Parameter	Index		Unit/index			Access	Default	Meaning / value range
		Dec	Hex	Abbr.	Sz.	Cv.			
084	Error t-4	8370	20B2		0	0	N/RO	0	
	Input terminals 1..6	8375	20B7		0	0	N/RO	0	Bit 0 = DI00 – Bit 5 = DI05
	Input terminals (opt.) 1..8	8380	20BC		0	0	N/RO	0	Bit 0 = DI10 – Bit 7 = DI17
	Output terminals 1..3	8385	20C1		0	0	N/RO	0	Bit 0 = DO00, Bit 1 = DO01, Bit 2 = DO02
	Output term. (optional) 1..8	8390	20C6		0	0	N/RO	0	Bit 0 = DO10 – Bit 7 = DO17
	Operational status	8395	20CB		0	0	N/RO	0	Low word coded as status word 1
	Heat sink temperature [°C]	8400	20D0	°C	17	100	N/RO	0	
	Speed [rpm]	8405	20D5	rps	11	66	N/RO	0	
	Output current [%]	8410	20DA	%	24	-3	N/RO	0	
	Unit utilization [%]	8420	20E4	%	24	-3	N/RO	0	
	DC link voltage [V]	8425	20E9	V	21	-3	N/RO	0	
	Mains ON operation time [h]	8430	20EE	s	4	70	N/RO	0	
	Operating time (enabled) [h]	8435	20F3	s	4	70	N/RO	0	
	Parameter set	8395	20CB		0	0	N/RO	0	
	Motor utilization 1 [%]	8445	20FD	%	24	-3	N/RO	0	
Motor utilization 2 [%]	8450	2102	%	24	-3	N/RO	0		
09. Bus diagnostics									
090	PD configuration	8451	2103		0	0	N/S/RO	4	0 = PARAM + 1PD 1 = 1PD 2 = PARAM + 2PD 3 = 2PD 4 = PARAM + 3PD 5 = 3PD 6 = PARAM + 6PD 7 = 6PD 8 = PARAM + 10PD 9 = 10PD
091	Fieldbus type	8452	2104		0	0	S/RO	0	0 = NO FIELDBUS 1 = PROFIBUS FMS/DP 2 = INTERBUS 3 = Reserved 4 = CAN 5 = PROFIBUS DP
092	Fieldbus baud rate	8453	2105		0	-3	S/RW	0	0 – FFFFFFFFh, step 1
093	Fieldbus address	8454	2106		0	0	S/RW	0	0 – 65535, step 1
094	PO1 setpoint [hex]	8455	2107		0	0	S/RO	0	
095	PO2 setpoint [hex]	8456	2108		0	0	S/RO	0	
096	PO3 setpoint [hex]	8457	2109		0	0	S/RO	0	
097	PI1 actual value [hex]	8458	210A		0	0	RO	0	
098	PI2 actual value [hex]	8459	210B		0	0	RO	0	
099	PI3 actual value [hex]	8460	210C		0	0	RO	0	



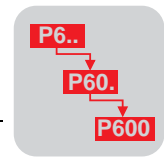
Par. no.	Parameter	Index		Unit/index			Access	Default	Meaning / value range
		Dec	Hex	Abbr.	Sz.	Cv.			
1.. Setpoints/ramp generators									
10. Setpoint selection									
100	Setpoint source	8461	210D		0	0	N/R/RW	0	0 = BIPOL./FIX.SETPT 1 = UNIPOL/FIX.SETPT 2 = RS-485 3 = FIELDBUS 4 = MOTOR POT 5 = MOT.POT +AI1 6 = FIX SETP+AI1 7 = FIX SETP*AI1 8 = MASTER-SBus. 9 = MASTER-RS-485 10 = SBus
101	Control signal source	8462	210E		0	0	N/R/RW	0	0 = TERMINALS 1 = RS-485 2 = FIELDBUS 3 = SBus
11. Analog input AI1									
110	AI1 scaling	8463	210F		0	-3	N/RW	1000	-10000 – -0, step 10 0 – 10000, step 10
111	AI1 offset [mV]	8464	2110	V	21	-3	N/RW	0	-500 – -0, step 1 0 – 500, step 1
112	AI1 operation mode	8465	2111		0	0	N/RW	1	0 = Ref. 3000 rpm 1 = Ref. N-MAX 2 = U-Off., N-MAX 3 = N-Off., N-MAX 4 = Expert charact. 5 = N-MAX, 0-20mA 6 = N-MAX, 4-20mA
113	AI1 voltage offset [V]	8466	2112	V	21	-3	N/RW	0	-10000 – -0, step 10 0 – 10000, step 10
114	AI1 speed offset [rpm]	8467	2113	rps	11	66	N/RW	0	-5000000 – -0, step 200 0 – 5000000, step 200
115	Filter setpoint [ms]	8468	2114	s	4	-6	N/RW	5000	0 – 1000, step 1000 1000 – 20000, step 10 20000 – 50000, step 100 50000 – 100000, step 1000
12. Analog inputs (optional)									
120	AI2 operation mode (optional)	8469	2115		0	0	N/R/RW	0	0 = NO FUNCTION 1 = 0. +/-10V+Setp.1 2 = 0..10V I-limit 3 = ACT.VAL.CONTROL.
13. Speed ramps 1									
130	Ramp t11 UP CW [s]	8470	2116	s	4	-3	N/RW	2000	0 – 1000, step 10 1000 – 10000, step 100 10000 – 100000, step 1000 100000 – 2000000, step 10000
131	Ramp t11 DOWN CW [s]	8471	2117	s	4	-3	N/RW	2000	0 – 1000, step 10 1000 – 10000, step 100 10000 – 100000, step 1000 100000 – 2000000, step 10000
132	Ramp t11 up CCW [s]	8472	2118	s	4	-3	N/RW	2000	0 – 1000, step 10 1000 – 10000, step 100 10000 – 100000, step 1000 100000 – 2000000, step 10000
133	Ramp t11 down CCW [s]	8473	2119	s	4	-3	N/RW	2000	0 – 1000, step 10 1000 – 10000, step 100 10000 – 100000, step 1000 100000 – 2000000, step 10000
134	Ramp t12 UP=DOWN [s]	8474	211A	s	4	-3	N/RW	10000	0 – 1000, step 10 1000 – 10000, step 100 10000 – 100000, step 1000 100000 – 2000000, step 10000



Par. no.	Parameter	Index		Unit/index			Access	Default	Meaning / value range
		Dec	Hex	Abbr.	Sz.	Cv.			
135	S pattern t12	8475	211B		0	0	N/RW	0	0 = 0 1 = 1 2 = 2 3 = 3
136	Stop ramp t13 [s]	8476	211C	s	4	-3	N/RW	2000	0 – 1000, step 10 1000 – 10000, step 100 10000 – 20000, step 1000
137	Emergency ramp t14 [s]	8477	211D	s	4	-3	N/RW	2000	0 – 1000, step 10 1000 – 10000, step 100 10000 – 20000, step 1000
138	Ramp limit	8794	225A		0	0	N/RW	0	0 = NO / 1 = YES
14. Speed ramps 2									
140	Ramp t21 UP CW [s]	8478	211E	s	4	-3	N/RW	2000	0 – 1000, step 10 1000 – 10000, step 100 10000 – 100000, step 1000 100000 – 2000000, step 10000
141	Ramp t21 DOWN CW [s]	8479	211F	s	4	-3	N/RW	2000	0 – 1000, step 10 1000 – 10000, step 100 10000 – 100000, step 1000 100000 – 2000000, step 10000
142	Ramp t21 up CCW [s]	8480	2120	s	4	-3	N/RW	2000	0 – 1000, step 10 1000 – 10000, step 100 10000 – 100000, step 1000 100000 – 2000000, step 10000
143	Ramp t21 down CCW [s]	8481	2121	s	4	-3	N/RW	2000	0 – 1000, step 10 1000 – 10000, step 100 10000 – 100000, step 1000 100000 – 2000000, step 10000
144	Ramp t22 UP=DOWN [s]	8482	2122	s	4	-3	N/RW	10000	0 – 1000, step 10 1000 – 10000, step 100 10000 – 100000, step 1000 100000 – 2000000, step 10000
145	S pattern t22	8483	2123		0	0	N/RW	0	See menu no. 135 or index 8475
146	Stop ramp t23 [s]	8484	2124	s	4	-3	N/RW	2000	0 – 1000, step 10 1000 – 10000, step 100 10000 – 20000, step 1000
147	Emergency ramp t24 [s]	8485	2125	s	4	-3	N/RW	2000	0 – 1000, step 10 1000 – 10000, step 100 10000 – 20000, step 1000
15. Motorized potentiometer									
150	Ramp t3 UP [s]	8486	2126	s	4	-3	N/RW	20000	200 – 1000, step 10 1000 – 10000, step 100 10000 – 50000, step 1000
151	Ramp t3 DOWN [s]	8487	2127	s	4	-3	N/RW	20000	200 – 1000, step 10 1000 – 10000, step 100 10000 – 50000, step 1000
152	Save last setpoint	8488	2128		0	0	N/RW	0	0 = OFF 1 = ON



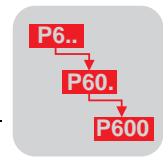
Par. no.	Parameter	Index		Unit/index			Access	Default	Meaning / value range
		Dec	Hex	Abbr.	Sz.	Cv.			
16. Fixed setpoints 1									
160	Internal setpoint n11 [rpm]	8489	2129	rps	11	66	N/RW	150000	-5000000 – -0, step 200 0 – 5000000, step 200
161	Internal setpoint n12 [rpm]	8490	212A	rps	11	66	N/RW	750000	-5000000 – -0, step 200 0 – 5000000, step 200
162	Internal setpoint n13 [rpm]	8491	212B	rps	11	66	N/RW	1500000	-5000000 – -0, step 200 0 – 5000000, step 200
	Internal setpoint n11 [%]	8489	2129	rps	11	66	N/RW	150000	-5000000 – -0, step 200 0 – 5000000, step 200
	Internal setpoint n12 [%]	8490	212A	rps	11	66	N/RW	750000	-5000000 – -0, step 200 0 – 5000000, step 200
	Internal setpoint n13 [%]	8491	212B	rps	11	66	N/RW	1500000	-5000000 – -0, step 200 0 – 5000000, step 200
17. Fixed setpoints 2									
170	Internal setpoint n22 [rpm]	8492	212C	rps	11	66	N/RW	150000	-5000000 – -0, step 200 0 – 5000000, step 200
171	Internal setpoint n22 [rpm]	8493	212D	rps	11	66	N/RW	750000	-5000000 – -0, step 200 0 – 5000000, step 200
172	Internal setpoint n23 [rpm]	8494	212E	rps	11	66	N/RW	1500000	-5000000 – -0, step 200 0 – 5000000, step 200
	Internal setpoint n21 [%]	8492	212C	rps	11	66	N/RW	150000	-5000000 – -0, step 200 0 – 5000000, step 200
	Internal setpoint n22 [%]	8493	212D	rps	11	66	N/RW	750000	-5000000 – -0, step 200 0 – 5000000, step 200
	Internal setpoint n23 [%]	8494	212E	rps	11	66	N/RW	1500000	-5000000 – -0, step 200 0 – 5000000, step 200



Par. no.	Parameter	Index		Unit/index			Access	Default	Meaning / value range
		Dec	Hex	Abbr.	Sz.	Cv.			
2.. Controller parameters									
20. Speed control									
200	P gain speed controller	8495	212F		0	-3	N/RW	2000	100 – 32000, step 10
201	Time constant n-control.[ms]	8496	2130	s	4	-6	N/RW	10000	0 – 1000, step 1000 1000 – 20000, step 10 20000 – 50000, step 100 50000 – 200000, step 1000 200000 – 300000, step 2000 300000 – 1000000, step 20000 1000000 – 3000000, step 200000
202	Gain accel. precontrol	8497	2131		0	-3	N/RW	0	0 – 32000, step 1
203	Filter accel. precontrol [ms]	8498	2132	s	4	-6	N/RW	0	0 – 1000, step 1000 1000 – 20000, step 10 20000 – 50000, step 100 50000 – 100000, step 1000
204	Filter speed actual value[ms]	8499	2133	s	4	-6	N/RW	0	0 – 1000, step 1000 1000 – 20000, step 10 20000 – 32000, step 100
205	Load precontrol	8436	20F4	%	24	-3	N/RW	0	-150000 – 0 – 150000, step 1000
206	Sample time n-control.	8437	20F5		0	0	N/RW	0	0 = 1.0 / 1 = 0.5
207	Load precontrol VFC	8786	2252	%	24	-3	N/RW	200000	-200000 – 0 – 200000, step 1000
21. Hold controller									
210	P gain hold controller	8500	2134		0	-3	N/RW	500	100 – 32000, step 10
22. Synchr. oper. control									
220•	P-gain (DRS)	8509	213D		0	-3	N/RW	10000	1000 – 200000, step 1000
221•	Master gear ratio factor	8502	2136		0	0	N/RW	1	1 – 3999999999, step 1
222•	Slave gear ratio factor	8503	2137		0	0	N/RW	1	1 – 3999999999, step 1
223•	Mode selection	8504	2138		0	0	N/RW	0	0 = MODE 1 1 = MODE 2 2 = MODE 3 3 = MODE 4 4 = MODE 5 5 = MODE 6 6 = MODE 7 7 = MODE 8
224•	Slave counter [Inc]	8505	2139		0	0	N/RW	10	-99999999 – -10, step 1 10 – 99999999, step 1
225•	Offset 1 [Inc]	8506	213A		0	0	N/RW	10	-32767 – -10, step 1 10 – 32767, step 1
226•	Offset 2 [Inc]	8507	213B		0	0	N/RW	10	-32767 – -10, step 1 10 – 32767, step 1
227•	Offset 3 [Inc]	8508	213C		0	0	N/RW	10	-32767 – -10, step 1 10 – 32767, step 1
228	Precontrol filter (DRS)	8438	20F6	s	4	-6	N/RW	0	0 – 1000, step 1 1000 – 20000, step 10 20000 – 50000, step 100 50000 – 100000, step 1000
23. Synchr. oper. w. sync encoder									
230•	Synchronous encoder	8510	213E		0	0	N/R/RW	0	0 = OFF 1 = EQUAL-RANKING 2 = CHAIN
231•	Factor slave encoder	8511	213F		0	0	N/RW	1	1 – 1000, step 1
232•	Factor slave sync. encoder	8512	2140		0	0	N/RW	1	1 – 1000, step 1
24. Synchr. oper. w. catch up									
240	Synchronization speed [rpm]	8513	2141	rps	11	66	N/RW	1500000	0 – 5000000, step 200
241	Synchronization ramp [s]	8514	2142	s	4	-3	N/RW	2000	0 – 1000, step 10 1000 – 10000, step 100 10000 – 50000, step 1000



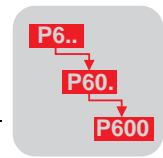
Par. no.	Parameter	Index		Unit/index			Access	Default	Meaning / value range
		Dec	Hex	Abbr.	Sz.	Cv.			
3.. Motor parameters									
30. Limits 1									
300	Start/stop speed 1 [rpm]	8515	2143	rps	11	66	N/RW	60000	0 – 150000, step 200
301	Minimum speed 1 [rpm]	8516	2144	rps	11	66	N/RW	60000	0 – 5500000, step 200
302	Maximum speed 1 [rpm]	8517	2145	rps	11	66	N/RW	1500000	0 – 5500000, step 200
303	Current limit 1 [%In]	8518	2146	%	24	-3	N/RW	150000	0 – 150000, step 1000
304	Torque limit	8688	21F0	%	24	-3	N/RW	0	0 – 150000, step 1000
31. Limits 2									
310	Start/stop speed 2 [rpm]	8519	2147	rps	11	66	N/RW	60000	0 – 150000, step 200
311	Minimum speed 2 [rpm]	8520	2148	rps	11	66	N/RW	60000	0 – 5500000, step 200
312	Maximum speed 2 [rpm]	8521	2149	rps	11	66	N/RW	1500000	0 – 5500000, step 200
313	Current limit 2 [%In]	8522	214A	%	24	-3	N/RW	150000	0 – 150000, step 1000
32. Motor compensat. 1 (asynchr.)									
320	Automatic adjustment 1	8523	214B		0	0	N/RW	1	See menu no. 152 or index 8488
321	Boost 1 [%]	8524	214C	V	21	-3	N/RW	0	0 – 100000, step 1000
322	IxR compensation 1 [%]	8525	214D	V	21	-3	N/RW	0	0 – 100000, step 1000
323	Premagnetizing time 1 [s]	8526	214E	s	4	-3	N/RW	100	0 – 2000, step 1
324	Slip compensation 1 [rpm]	8527	214F	rps	11	66	N/RW	0	0 – 500000, step 200
33. Motor compensat. 2 (asynchr.)									
330	Automatic adjustment 2	8528	2150		0	0	N/RW	1	See menu no. 152 or index 8488
331	Boost 2 [%]	8529	2151	V	21	-3	N/RW	0	0 – 100000, step 1000
332	IxR compensation 2 [%]	8530	2152	V	21	-3	N/RW	0	0 – 100000, step 1000
333	Premagnetizing time 2 [s]	8531	2153	s	4	-3	N/RW	100	0 – 2000, step 1
334	Slip compensation 2 [rpm]	8532	2154	rps	11	66	N/RW	0	0 – 500000, step 200
34. Motor protection									
340	Motor protection 1	8533	2155		0	0	N/RW	0	See menu no. 152 or index 8488
341	Cooling type 1	8534	2156		0	0	N/RW	0	0 = FAN COOLED 1 = FORCED COOLING
342	Motor protection 2	8535	2157		0	0	N/RW	0	See menu no. 152 or index 8488
343	Cooling type 2	8536	2158		0	0	N/RW	0	See menu no. 341 or index 8534
35. Motor sense of rotation									
350	Change direction of rotation 1	8537	2159		0	0	N/R/RW	0	See menu no. 152 or index 8488
351	Change direction of rotation 2	8538	215A		0	0	N/R/RW	0	See menu no. 152 or index 8488



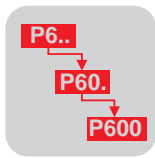
Par. no.	Parameter	Index		Unit/index			Access	Default	Meaning / value range
		Dec	Hex	Abbr.	Sz.	Cv.			
4.. Reference signals									
40. Speed reference signal									
400	Speed reference value [rpm]	8539	215B	rps	11	66	N/RW	1500000	0 – 5000000, step 200
401	Hysteresis [rpm]	8540	215C	rps	11	66	N/RW	100000	0 – 500000, step 1000
402	Delay time [s]	8541	215D	s	4	-3	N/RW	1000	0 – 9000, step 100
403	Signal = "1" if:	8542	215E		0	0	N/RW	0	0 = n < n ref 1 = n > n ref
41. Speed window signal									
410	Window center [rpm]	8543	215F	rps	11	66	N/RW	1500000	0 – 5000000, step 200
411	Range width [rpm]	8544	2160	rps	11	66	N/RW	0	0 – 5000000, step 200
412	Delay time [s]	8545	2161	s	4	-3	N/RW	1000	0 – 9000, step 100
413	Signal = "1" if:	8546	2162		0	0	N/RW	0	0 = INSIDE 1 = OUTSIDE
42. Speed setp./act. val. comp.									
420	Hysteresis [rpm]	8547	2163	rps	11	66	N/RW	100000	1000 – 300000, step 1000
421	Delay time [s]	8548	2164	s	4	-3	N/RW	1000	0 – 9000, step 100
422	Signal = "1" if:	8549	2165		0	0	N/RW	1	0 = n <> n setpt 1 = n = n setpt
43. Current reference signal									
430	Current reference value [%In]	8550	2166	%	24	-3	N/RW	100000	0 – 150000, step 1000
431	Hysteresis [%In]	8551	2167	%	24	-3	N/RW	5000	0 – 30000, step 1000
432	Delay time [s]	8552	2168	s	4	-3	N/RW	1000	0 – 9000, step 100
433	Signal = "1" if:	8553	2169		0	0	N/RW	0	0 = I < I ref 1 = I > I ref
44. Imax signal									
440	Hysteresis [%In]	8554	216A	%	24	-3	N/RW	5000	5000 – 50000, step 1000
441	Delay time [s]	8555	216B	s	4	-3	N/RW	1000	0 – 9000, step 100
442	Signal = "1" if:	8556	216C		0	0	N/RW	1	0 = I=Imax 1 = I<Imax
5.. Monitoring functions									
50. Speed monitoring									
500	Speed monitoring 1	8557	216D		0	0	N/RW	3	0 = OFF 1 = MOTOR MODE 2 = REGENERAT. MODE 3 = MOT.& REGEN.MODE
501	Delay time 1 [s]	8558	216E	s	4	-3	N/RW	1000	0 – 10000, step 10
502	Speed monitoring 2	8559	216F		0	0	N/RW	3	See menu no. 500 or index 8557
503	Delay time 2 [s]	8560	2170	s	4	-3	N/RW	1000	0 – 10000, step 10
51. Synchr. operation monitoring									
510•	Positioning tol. slave [Inc]	8561	2171		0	0	N/RW	25	10 – 32768, step 1
511•	Prewarning lag error [Inc]	8562	2172		0	0	N/RW	50	50 – 99999999, step 1
512•	Lag error limit [Inc]	8563	2173		0	0	N/RW	4000	100 – 99999999, step 1
513•	Delay lag error signal [s]	8564	2174	s	4	-3	N/RW	1000	0 – 99000, step 100
514•	Counter LED display [Inc]	8565	2175		0	0	N/RW	100	10 – 32768, step 1
515•	Delay in-position signal [ms]	8566	2176	s	4	-3	N/RW	10	5 – 2000, step 1
52. Mains OFF monitoring									
520	Mains OFF response time[s]	8567	2177	s	4	-3	N/RW	0	0 – 5000, step 1
521	Mains OFF response	8753	2231		0	0	N/RW	0	0 = CONTROL.INHIBIT 1 = EMERGENCY STOP



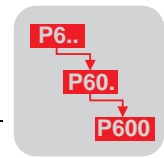
Par. no.	Parameter	Index		Unit/index			Access	Default	Meaning / value range
		Dec	Hex	Abbr.	Sz.	Cv.			
6.. Terminal assignment									
60. Binary inputs basic unit									
600	Binary input DI01	8335	208F		0	0	N/R/RW	2	0 = NO FUNCTION 1 = ENABLE/RAP.STOP 2 = CW/STOP 3 = CCW/STOP 4 = n11/n21 5 = n12/n22 6 = FIX SETPT SW.OV 7 = PAR. SWITCHOVER 8 = RAMP SWITCHOVER 9 = MOTOR POT UP 10 = MOTOR POT DOWN 11 = /EXT. ERROR 12 = ERROR RESET 13 = /HOLD CONTROL 14 = /LIM. SWITCH CW 15 = /LIM. SWITCH CCW 16 = IPOS INPUT 17 = REFERENCE CAM 18 = REF.TRAVEL START 19 = SLAVE FREE RUNN. 20 = SETPOINT HOLD 21 = MAINS ON 22 = DRS SET ZERO PT. 23 = DRS SLAVE START 24 = DRS TEACH IN 25 = DRS MAST.STOPPED
601	Binary input DI02	8336	2090		0	0	N/R/RW	3	See menu no. 600 or index 8335
602	Binary input DI03	8337	2091		0	0	N/R/RW	1	See menu no. 600 or index 8335
603	Binary input DI04	8338	2092		0	0	N/R/RW	4	See menu no. 600 or index 8335
604	Binary input DI05	8339	2093		0	0	N/R/RW	5	See menu no. 600 or index 8335
61. Binary inputs option									
610•	Binary input DI10	8340	2094		0	0	N/R/RW	0	See menu no. 600 or index 8335
611•	Binary input DI11	8341	2095		0	0	N/R/RW	0	See menu no. 600 or index 8335
612•	Binary input DI12	8342	2096		0	0	N/R/RW	0	See menu no. 600 or index 8335
613•	Binary input DI13	8343	2097		0	0	N/R/RW	0	See menu no. 600 or index 8335
614•	Binary input DI14	8344	2098		0	0	N/R/RW	0	See menu no. 600 or index 8335
615•	Binary input DI15	8345	2099		0	0	N/R/RW	0	See menu no. 600 or index 8335
616•	Binary input DI16	8346	209A		0	0	N/R/RW	0	See menu no. 600 or index 8335
617•	Binary input DI17	8347	209B		0	0	N/R/RW	0	See menu no. 600 or index 8335



Par. no.	Parameter	Index		Unit/index			Access	Default	Meaning / value range
		Dec	Hex	Abbr.	Sz.	Cv.			
62. Binary outputs basic unit									
620	Binary output DO01	8350	209E		0	0	N/RW	2	0 = NO FUNCTION 1 = /ERROR 2 = READY 3 = OUTP. STAGE ON 4 = ROT. FIELD ON 5 = BRAKE RELEASED 6 = BRAKE APPLIED 7 = MOTOR STANDSTILL 8 = PARAMETER SET 9 = SPEED REFERENCE 10 = SPEED WINDOW 11 = SP/ACT.VAL.COMP. 12 = CURR. REFERENCE 13 = I _{max} -SIGNAL 14 = /MOTOR UTILIZ.1 15 = /MOTOR UTILIZ.2 16 = /DRS PREWARN. 17 = /DRS LAG ERROR 18 = DRS SLAVE IN POS 19 = IPOS IN POSITION 20 = IPOS REFERENCE 21 = IPOS OUTPUT 22 = /IPOS ERROR
621	Binary output DO02	8351	209F		0	0	N/RW	1	See menu no. 620 or index 8350
63. Binary outputs option									
630•	Binary output DO10	8352	20A0		0	0	N/RW	0	See menu no. 620 or index 8350
631•	Binary output DO11	8353	20A1		0	0	N/RW	0	See menu no. 620 or index 8350
632•	Binary output DO12	8354	20A2		0	0	N/RW	0	See menu no. 620 or index 8350
633•	Binary output DO13	8355	20A3		0	0	N/RW	0	See menu no. 620 or index 8350
634•	Binary output DO14	8356	20A4		0	0	N/RW	0	See menu no. 620 or index 8350
635•	Binary output DO15	8357	20A5		0	0	N/RW	0	See menu no. 620 or index 8350
636•	Binary output DO16	8358	20A6		0	0	N/RW	0	See menu no. 620 or index 8350
637•	Binary output DO17	8359	20A7		0	0	N/RW	0	See menu no. 620 or index 8350
64. Analog outputs optional									
640	Analog output AO1	8568	2178		0	0	N/RW	3	0 = NO FUNCTION 1 = RAMP INPUT 2 = SPEED SETPOINT 3 = ACTUAL SPEED 4 = ACTUAL FREQUENCY 5 = OUTPUT CURRENT 6 = ACTIVE CURRENT 7 = UNIT UTILIZATION 8 = IPOS OUTPUT
641	Scaling AO1	8569	2179		0	-3	N/RW	1000	-10000 – -0, step 10 0 – 10000, step 10
642	Operating mode AO1	8570	217A		0	0	N/RW	1	0 = OFF 1 = -10V..10V 2 = 0..20mA 3 = 4..20mA
643•	Analog output AO2	8571	217B		0	0	N/RW	5	See menu no. 640 or index 8568
644•	Scaling AO2	8572	217C		0	-3	N/RW	1000	-10000 – -0, step 10 0 – 10000, step 10
645•	Operating mode AO2	8573	217D		0	0	N/RW	1	See menu no. 642 or index 8570



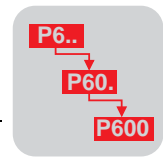
Par. no.	Parameter	Index		Unit/index			Access	Default	Meaning / value range	
		Dec	Hex	Abbr.	Sz.	Cv.				
7.. Control functions										
70. Operating modes										
700	Operating mode 1	8574	217E			0	0	N/R/RW	0	0 = VFC 1 1 = VFC 1 & GROUP 2 = VFC 1 & HOIST 3 = VFC 1 & DC BRAK. 4 = VFC 1 & FLY.START 5 = VFC-n-CONTROL 6 = VFC-n-CTRL&GROUP 7 = VFC-n-CTRL&HOIST 8 = VFC-n-CTRL& SYNC 9 = VFC-n-CTRL& IPOS 10 = VFC-n-CTRL& DPx 11 = CFC 12 = CFC & M-CONTROL 13 = CFC & IPOS 14 = CFC & SYNC. 15 = CFC & DPx 16 = SERVO 17 = SERVO & M-CONTROL 18 = SERVO & IPOS 19 = SERVO & SYNC. 20 = SERVO & DPx
701	Operating mode 2	8575	217F			0	0	N/R/RW	0	0 = VFC 2 1 = VFC 2 & GROUP 2 = VFC 2 & HOIST 3 = VFC 2 & DC BRAK. 4 = VFC 2 & FLY.START
71. Current at standstill										
710	Standstill current 1 [%Imot.]	8576	2180	A	22	-3		N/RW	0	0 – 50000, step 1000
711	Standstill current 2 [%Imot.]	8577	2181	A	22	-3		N/RW	0	0 – 50000, step 1000
72. Setpoint stop function										
720	Setpoint stop function 1	8578	2182			0	0	N/RW	0	See menu no. 152 or index 8488
721	Stop setpoint 1 [rpm]	8579	2183	rps	11	66		N/RW	30000	0 – 500000, step 200
722	Start offset 1 [rpm]	8580	2184	rps	11	66		N/RW	30000	0 – 500000, step 200
723	Setpoint stop function 2	8581	2185			0	0	N/RW	0	See menu no. 152 or index 8488
724	Stop setpoint 2 [rpm]	8582	2186	rps	11	66		N/RW	30000	0 – 500000, step 200
725	Start offset 2 [rpm]	8583	2187	rps	11	66		N/RW	30000	0 – 500000, step 200
73. Brake function										
730	Brake function 1	8584	2188			0	0	N/RW	1	See menu no. 152 or index 8488
731	Brake release time 1 [s]	8749	222D	s	4	-3		N/RW	0	0 – 2000, step 1
732	Brake application time 1[s]	8585	2189	s	4	-3		N/RW	200	0 – 2000, step 1
733	Brake function 2	8586	218A			0	0	N/RW	1	See menu no. 152 or index 8488
734	Brake release time 2 [s]	8750	222E	s	4	-3		N/RW	0	0 – 2000, step 1
735	Brake application time 2[s]	8587	218B	s	4	-3		N/RW	200	0 – 2000, step 1
74. Speed skip										
740	Skip window center 1 [rpm]	8588	218C	rps	11	66		N/RW	1500000	0 – 5000000, step 200
741	Skip width 1 [rpm]	8589	218D	rps	11	66		N/RW	0	0 – 300000, step 200
742	Skip window center 2 [rpm]	8590	218E	rps	11	66		N/RW	1500000	0 – 5000000, step 200
743	Skip width 2 [rpm]	8591	218F	rps	11	66		N/RW	0	0 – 300000, step 200
75. Master-Slave function										
750	Slave setpoint	8592	2190			0	0	N/RW	0	0 = MASTER-SLAVE OFF 1 = SPEED (RS-485) 2 = SPEED (SBus) 3 = SPEED (485+SBus) 4 = TORQUE (RS-485) 5 = TORQUE (SBus) 6 = TORQUE (485+SBus) 7 = LOAD SHAR(RS485) 8 = LOAD SHARE(SBus) 9 = LOAD S(485+SBus)
751	Scaling slave setpoint	8593	2191			0	-3	N/RW	1000	-10000 – -0, step 1 0 – 10000, step 1



Par. no.	Parameter	Index		Unit/index			Access	Default	Meaning / value range
		Dec	Hex	Abbr.	Sz.	Cv.			
8.. Unit functions									
80. Setup									
802	Factory setting	8594	2192		0	0	R/RW	0	0 = NO 1 = YES
803	Parameter lock	8595	2193		0	0	N/S/RW	0	See menu no. 152 or index 8488
804	Reset statistic data	8596	2194		0	0	RW	0	0 = NO 1 = ERROR MEMORY 2 = KWH-METER 3 = OPERATING HOURS
81. Serial communication									
810	RS485 address	8597	2195		0	0	N/RW	0	0 – 99, step 1
811	RS-485 group address	8598	2196		0	0	N/RW	100	100 – 199, step 1
812	RS485 timeout delay [s]	8599	2197	s	4	-3	N/RW	0	0 – 650000, step 10
813	SBus address	8600	2198		0	0	N/RW	0	0 – 63, step 1
814	SBus group address	8601	2199		0	0	N/RW	0	0 – 63, step 1
815	SBus timeout delay [s]	8602	219A	s	4	-3	N/RW	100	0 – 650000, step 10
816	SBus baud rate [kBaud]	8603	219B		0	0	N/RW	2	0 = 125 1 = 250 2 = 500 3 = 1000
817	SBus synchronization ID	8604	219C		0	-3	N/RW	0	0 – 2047000, step 1000
818	CAN synchronization ID	8732	221C		0	-3	N/RW	1000	0 – 2047000, step 1000
819	Fieldbus timeout delay [s]	8606	219E	s	4	-3	N/S/RW	500	0 – 650000, step 10
82. Brake operation									
820	4-quadrant operation 1	8607	219F		0	0	N/RW	1	See menu no. 152 or index 8488
821	4-quadrant operation 2	8608	21A0		0	0	N/RW	1	See menu no. 152 or index 8488
83. Error response									
830	Response EXT. ERROR	8609	21A1		0	0	N/RW	3	0 = NO RESPONSE 1 = DISPLAY ERROR 2 = IMM. STOP/ERROR 3 = EMERG.STOP/ERROR 4 = RAPID STOP/ERROR 5 = IMM. STOP/WARNG 7 = EMERG. STOP/WARNG 7 = RAPID STOP/WARNG
831	Response FIELDBUS TIMEOUT	8610	21A2		0	0	N/RW	4	See menu no. 830 or index 8609
832	Response MOTOR OVERLOAD	8611	21A3		0	0	N/RW	3	See menu no. 830 or index 8609
833	Response RS485 TIMEOUT	8612	21A4		0	0	N/RW	7	See menu no. 830 or index 8609
834	Response DRS LAG ERROR	8613	21A5		0	0	N/RW	3	See menu no. 830 or index 8609
835	Response TF sensor SIGNAL	8616	21A8		0	0	N/RW	0	See menu no. 830 or index 8609
836	Response SBus TIMEOUT	8615	21A7		0	0	N/RW	3	See menu no. 830 or index 8609
84. Reset response									
840	Manual reset	8617	21A9		0	0	S/RW	0	See menu no. 802 or index 8594
841	Auto reset	8618	21AA		0	0	N/RW	0	See menu no. 152 or index 8488
842	Restart time [s]	8619	21AB	s	4	-3	N/RW	3000	1000 – 30000, step 1000
85. Scaling speed actual value									
850	Scaling factor numerator	8747	222B		0	0	N/RW	1	1 – 65535, step 1
851	Scaling factor denominator	8748	222C		0	0	N/RW	1	1 – 65535, step 1
852	User dimension	8772 8773	2244		0	0	N/RW	1768763 185	2 × ASCII characters
86. Modulation									
860	PWM frequency 1 [kHz]	8620	21AC		0	0	N/RW	0	0 = 4 1 = 8 2 = 12 3 = 16
861	PWM frequency 2 [kHz]	8621	21AD		0	0	N/RW	0	See menu no. 860 or index 8620
862	PWM fix 1	8751	222F		0	0	N/RW	0	See menu no. 152 or index 8488
863	PWM fix 2	8752	2230		0	0	N/RW	0	See menu no. 152 or index 8488



Par. no.	Parameter	Index		Unit/index			Access	Default	Meaning / value range	
		Dec	Hex	Abbr.	Sz.	Cv.				
87. Process data description										
870	Setpoint description PO1	8304	2070			0	0	N/RW	9	0 = NO FUNCTION 1 = SPEED 2 = CURRENT 3 = POSITION LO 4 = POSITION HI 5 = MAX. SPEED 6 = MAX. CURRENT 7 = SLIP 8 = RAMP 9 = CTRL. WORD 1 10 = CTRL. WORD 2 11 = SPEED [%] 12 = IPOS PO-DATA
871	Setpoint description PO2	8305	2071			0	0	N/RW	1	See menu no. 870 or index 8304
872	Setpoint description PO3	8306	2072			0	0	N/RW	0	See menu no. 870 or index 8304
873	Actual value description PI1	8307	2073			0	0	N/RW	6	0 = NO FUNCTION 1 = SPEED 2 = OUTP. CURRENT 3 = ACTIVE CURR. 4 = POSITION LO 5 = POSITION HI 6 = STATUS WORD1 7 = STATUS WORD2 8 = SPEED [%] 9 = IPOS-PI DATA 10 = RESERVED 11 = STATUS WORD3
874	Actual value description PI2	8308	2074			0	0	N/RW	1	See menu no. 873 or index 8307
875	Actual value description PI3	8309	2075			0	0	N/RW	2	See menu no. 873 or index 8307
876	PO data enable	8622	21AE			0	0	N/S/RW	1	See menu no. 152 or index 8488
9.. IPOS parameters										
90. IPOS Reference travel										
900	Reference offset [Inc]	8623	21AF			0	0	N/RW	0	-7FFFFFFFh – -0, step 1 0 – 7FFFFFFFh, step 1
901	Reference speed 1 [rpm]	8624	21B0	rps		11	66	N/RW	200000	0 – 5000000, step 200
902	Reference speed 2 [rpm]	8625	21B1	rps		11	66	N/RW	50000	0 – 5000000, step 200
903	Reference travel type	8626	21B2			0	0	N/RW	0	0 – 7, step 1
91. IPOS Travel parameter										
910	Gain X controller	8627	21B3			0	-3	N/RW	500	0 – 32000, step 10
911	Positioning ramp 1 [s]	8628	21B4	s		4	-3	N/RW	1000	10 – 500, step 1 500 – 2000, step 10 2000 – 10000, step 200 10000 – 20000, step 1000
912	Positioning ramp 2 [s]	8696	21F8	s		4	-3	N/RW	1000	10 – 500, step 1 500 – 2000, step 10 2000 – 10000, step 200 10000 – 20000, step 1000
913	Travel speed CW [rpm]	8629	21B5	rps		11	66	N/RW	1500000	0 – 5000000, step 200
914	Travel speed CCW [rpm]	8630	21B6	rps		11	66	N/RW	1500000	0 – 5000000, step 200
915	Speed feedforward [%]	8631	21B7			0	-3	N/RW	100000	-199990 – -0, step 10 0 – 199990, step 10
916	Ramp type	8632	21B8			0	0	N/RW	0	0 = LINEAR 1 = SINE 2 = SQUARED
92. IPOS Monitoring										
920	SW limit switch CW [Inc]	8633	21B9			0	0	N/RW	0	-7FFFFFFFh – -0, step 1 0 – 7FFFFFFFh, step 1
921	SW limit switch CCW [Inc]	8634	21BA			0	0	N/RW	0	-7FFFFFFFh – -0, step 1 0 – 7FFFFFFFh, step 1
922	Position window [Inc]	8635	21BB			0	0	N/RW	50	0 – 32767, step 1
923	Lag error window [Inc]	8636	21BC			0	0	N/RW	5000	0 – 7FFFFFFFh, step 1



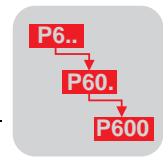
Par. no.	Parameter	Index		Unit/index			Access	Default	Meaning / value range
		Dec	Hex	Abbr.	Sz.	Cv.			
93. IPOS Special functions									
930	Override	8637	21BD		0	0	N/RW	0	See menu no. 152 or index 8488
94. IPOS Encoder									
941	Source actual position	8729	2219		0	0	N/RW	0	0 = MOTOR ENC. (X15) 1 = EXTERN.ENC (X14) 2 = ABSOL.ENC. (DIP)
942	Encoder factor numerator	8774	2246		0	0	N/RW	1	1 – 32767, step 1
943	Encoder factor denominator	8775	2247		0	0	N/RW	1	1 – 32767, step 1
944	Encoder scaling ext. encoder	8787	2253		0	0	N/R/RW	0	0 = x 1 1 = x 2 2 = x 4 3 = x 8 4 = x 16 5 = x 32 6 = x 64
945	Encoder type (X14)	8891	2288		0	0	N/RW	0	0 = TTL 1 = SIN/COS 2 = HTL 3 = HIPERFACE
95. DIP									
950	Encoder type	8777	2249		0	0	N/R/RW	0	0 = NO ENCODER 1 = VISOLUX EDM 2 = T&R CE65,CE100 MSSI 3 = RESERVED 4 = RESERVED 5 = RESERVED 6 = STEGMANN AG100 MSSI 7 = SICK DME-3000-111 8 = STAHL WCS2-LS311
951	Counting direction	8776	2248		0	0	N/R/RW	0	0 = NORMAL 1 = INVERTED
952	Cycle frequency [%]	8778	224A	%	24	-3	N/R/RW	100000	1000 – 200000, step 100
953	Position offset	8779	224B		0	0	N/RW	0	-7FFFFFFFh – -0, step 1 0 – 7FFFFFFFh, step 1
954	Zero offset	8781	224D		0	0	N/RW	0	-7FFFFFFFh – -0, step 1 0 – 7FFFFFFFh, step 1
955	Encoder scaling	8784	2250		0	0	N/R/RW	0	0 = x 1 1 = x 2 2 = x 4 3 = x 8 4 = x 16 5 = x 32 6 = x 64
96. IPOS Modulo Function									
960	Modulo function	8835	2283		0	0	N/RW	0	0 = OFF 1 = SHORT 2 = CW 3 = CCW
961	Modulo numerator	8836	2284		0	0	N/RW	1	1 – 2147483647, step 1
962	Modulo denominator	8837	2285		0	0	N/RW	1	1 – 2147483647, step 1
963	Modulo encoder resolution	8838	2286		0	0	N/RW	1	1 – 65535, step 1



7.3 Quantity and conversion index

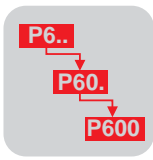
Quantity and conversion index from the PNO sensor/actuator profile

Physical quantity	Quantity index	Unit	Abbreviation	Conversion index
	0	Dimensionless		0
Length	1	Meter	m	0
		Millimeter	mm	-3
		Kilometer	km	3
		Micrometer	µm	-6
Area	2	Square meter	m ²	0
		Square millimeter	mm ²	-6
		Square kilometer	km ²	6
Volume	3	Cubic meter	m ³	0
		Liter	l	-3
Time	4	Second	s	0
		Minute	min	70
		Hour	h	74
		Day	d	77
		Millisecond	ms	-3
		Microsecond	µs	-6
Force	5	Newton	N	0
		Kilonewton	kN	3
		Meganewton	MN	6
Pressure	6	Pascal	Pa	0
		Kilopascal	kPa	3
		Millibar	mbar	2
		Bar	bar	5
Mass	7	Kilogram	kg	0
		Gram	g	-3
		Milligram	mg	-6
		Tonne	t	3
Energy, work	8	Joule	J	0
		Kilojoule	kJ	3
		Megajoule	MJ	6
		Watt hour	Wh	74
		Kilowatt hour	kWh	75
		Megawatt hour	MWh	76
Effective power	9	Watt	W	0
		Kilowatt	kW	3
		Megawatt	MW	6
		Milliwatt	mW	-3
Apparent power	10	Volt amp	VA	0
		Kilovolt amp	kVA	3
		Megavolt amp	MVA	6
		Millivolt amp	mVA	-3
Rotational velocity	11	Revolution/second	rps	0
		Revolution/minute	rpm	67
		Revolution/hour	rph	72
Angle	12	Radian	rad	0
		Second	"	79
		Minute	'	78
		Degree	°	80
Speed	13	Meter/second	m/s	0
		Millimeter/second	mm/s	-3
		Millimeter/minute	mm/min	66
		Meter/minute	m/min	67
		Kilometer/minute	km/min	68
		Millimeter/hour	mm/h	71
		Meter/hour	m/h	72
		Kilometer/hour	km/h	73



Physical quantity	Quantity index	Unit	Abbreviation	Conversion index
Volume rate of flow	14	Cubic meter/second	m ³ /s	0
		Cubic meter/minute	m ³ /min	67
		Cubic meter/hour	m ³ /h	72
		Liter/second	l/s	-3
		Liter/minute	l/min	66
		Liter/hour	l/h	71
Mass rate of flow	15	Kilogram/second	kg/s	0
		Gram/second	g/s	-3
		Tonne/second	t/s	3
		Gram/minute	g/min	66
		Kilogram/minute	kg/min	67
		Tonne/minute	t/min	68
		Gram/hour	g/h	71
		Kilogram/hour	kg/h	72
		Tonne/hour	t/h	73
		Torque	16	Newton meter
Kilonewton meter	kNm			3
Meganewton meter	MNm			6
Temperature	17	Kelvin	K	0
		Degree Celsius	°C	100
		Degree Fahrenheit	°F	101
Temperature difference	18	Kelvin	K	0
Entropy	19	Joule/(Kelvin × kg)	J/(K × kg)	0
		kJ/(K × kg)	kJ/(K × kg)	3
		MJ/(K × kg)	MJ/(K × kg)	6
Enthalpy	20	Joule/kilogram	J/kg	0
		Kilojoule/kilogram	kJ/kg	3
		Megajoule/kilogram	MJ/kg	6
Electrical voltage	21	Volt	V	0
		Kilovolt	kV	3
		Millivolt	mV	-3
		Microvolt	μV	-6
Electrical current	22	Amp	A	0
		Milliamp	mA	-3
		Kiloamp	kA	3
		Microamp	μA	-6
Electrical resistance	23	Ohm	Ω	0
		Milliohm	mΩ	-3
		Kiloohm	kΩ	3
		Megaohm	MΩ	6
Ratio	24	Percentage	%	0
Relative humidity	25	Percentage	%	0
Absolute humidity	26	Gram/kilogram	g/kg	-3
Relative change	27	Percentage	%	0
Frequency	28	Hertz	Hz	0
		Kilohertz	kHz	3
		Megahertz	MHz	6
		Gigahertz	GHz	9

Conversion index	A (conversion factor)	1/A (reciprocal conversion factor)	B (offset)
0	1.E+0	1.E+0	0
1	10 = 1.E+1	1.E-1	0
2	100 = 1.E+2	1.E-2	0
3	1000 = 1.E+3	1.E-3	0
etc.			
-1	0.1 = 1.E-1	1.E+1	0
-2	0.01 = 1.E-2	1.E+2	0
-3	0.001 = 1.E-3	1.E+3	0



Conversion index	A (conversion factor)	1/A (reciprocal conversion factor)	B (offset)
etc.			
66	$1.E-3/60 = 1.667 E-5$	6.000 E+4	0
67	$1/60 = 1.667 E-2$	6.000 E+1	0
68	$1.E+3/60 = 1.667 E+1$	6.000 E-2	0
69			0
70	60	1.667 E-2	0
71	$1.E-3/3600 = 2.778 E-7$	3.6 E+6	0
72	$1/3600 = 2.778 E-4$	3.6 E+3	0
73	$1.E+3/3600 = 2.778 E-1$	3.6	0
74	3600	$1/3600 = 2.778 E-4$	0
75	$3600 \times 1.E+3 = 3.600 E+6$	2.778 E-7	0
76	$3600 \times 1.E+6 = 3.600 E+9$	2.778 E-10	0
77	86 400	$1/86\ 400 = 1.157 E-5$	0
78	$p / 10\ 800 = 2.909 E-4$	3.438 E+3	0
79	$p / 648\ 000 = 4.848 E-6$	2.063 E+5	0
80	$p / 180 = 1.745 E-2$	5.730 E+1	0
81	$p / 200 = 1.571 E-2$	6.366 E+1	0
100	1	1	273.15 K
101	$5/9 = 0.5556$	1.8	255.37 K

Example

The conversion figures should be used as follows:

$$(\text{Physical value in multiples or fractions of the unit}) = (\text{Transferred value} \times \text{Unit}) \times A + B$$

Example:

Transferred via the bus:

Numerical value	Quantity index	Conversion index
1500	4	-3

The recipient assigns the following values to these figures:

4 → Measurement quantity "Time"

-3 → Unit of measurement "Milliseconds"

→ $1500 \text{ ms} = 1500 \text{ s} \times A + B = 1500 \text{ s} \times 0.001 + 0 \text{ s} = 1.5 \text{ s}$

Conversion indices greater than +64 generally have a special meaning that must be determined in the table above. These units include day, hour, minute rather than SI-compatible units, such as Fahrenheit, etc.



8 Index

A

Acyclical data exchange 17
Address byte 20

B

Block check character, creating 25
Broadcast 22
Broadcast address 22

C

CAN bus identifier 37
CAN identifiers, assignment example 52
Character delay time 27
Character frame 26
Conversion index 32
Cyclical data exchange 17

D

Data contents 29
Data exchange, slave 37

G

Group addressing 21
Group parameter message 39
Group process data message 40

I

Incorrect performance of service 30
Index addressing 30
Index addressing, SBus 43
Individual addressing 20
Installation, RS-232 interface 16
Installation, RS-485 interface 14
Installation, system bus 12
INTEL format 60

M

Management of the parameter message, SBus 43
Master data exchange 47
Master/slave operation via SBus 50
Message processing 28
Message structure
 Request message 18
 Response message 18
Message traffic 17
MOTOROLA format 57
MOVILINK® parameter channel, structure 29
MOVILINK® parameters 67
MOVILINK®, general description 9
Multicast 21

N

Notes, important 4

O

Overview of serial interfaces 5

P

Parameter channel, management 29
Parameter list 68
 - 0.. *Display values* 68
 - 1.. *Setpoints/ramp generators* 72
 - 2.. *Controller parameters* 75
 - 3.. *Motor parameters* 76
 - 4.. *Reference signals* 77
 - 5.. *Monitoring functions* 77
 - 6.. *Terminal assignment* 78
 - 7.. *Control functions* 80
 - 9.. *IPOS parameters* 82
Parameter messages 40
Parameter services, description 31
Parameter settings 41
Parameter, reading 32
Parameter, writing 33
Parameterization with cyclical PDU types 34
PDU type
 Acyclical 24
 Cyclical 23
 Structure 23
PDU types 36
Process data messages 39

Q

Quantity index 31

R

Request message, structure 18
Response delay time 27
Response message, structure 18
Return codes for parameterization 46
RS-485 timeout 27

S

Safety notes 4
Safety notes on bus systems 4
Sample application, control via 3 process data words 35
SBus parameter setting, sample program 48
SBus project planning example 62
SBus startup problems 64
Setting parameters via the CAN bus 42
Start character 19
Start pause 19
Structure of the parameter message, SBus 42
Synchronization message 38



System bus, general description 10

T

Technical data 8

USS21A serial interface 7

Transfer formats, INTEL format 60

Transfer formats, MOTOROLA format 57

Transmission process 26

Transmission rate 27

Transmission reliability 25

U

Universal addressing 21

USS21A 7

USS21A serial interface, technical data 7

V

Variable message, acyclical sending 56

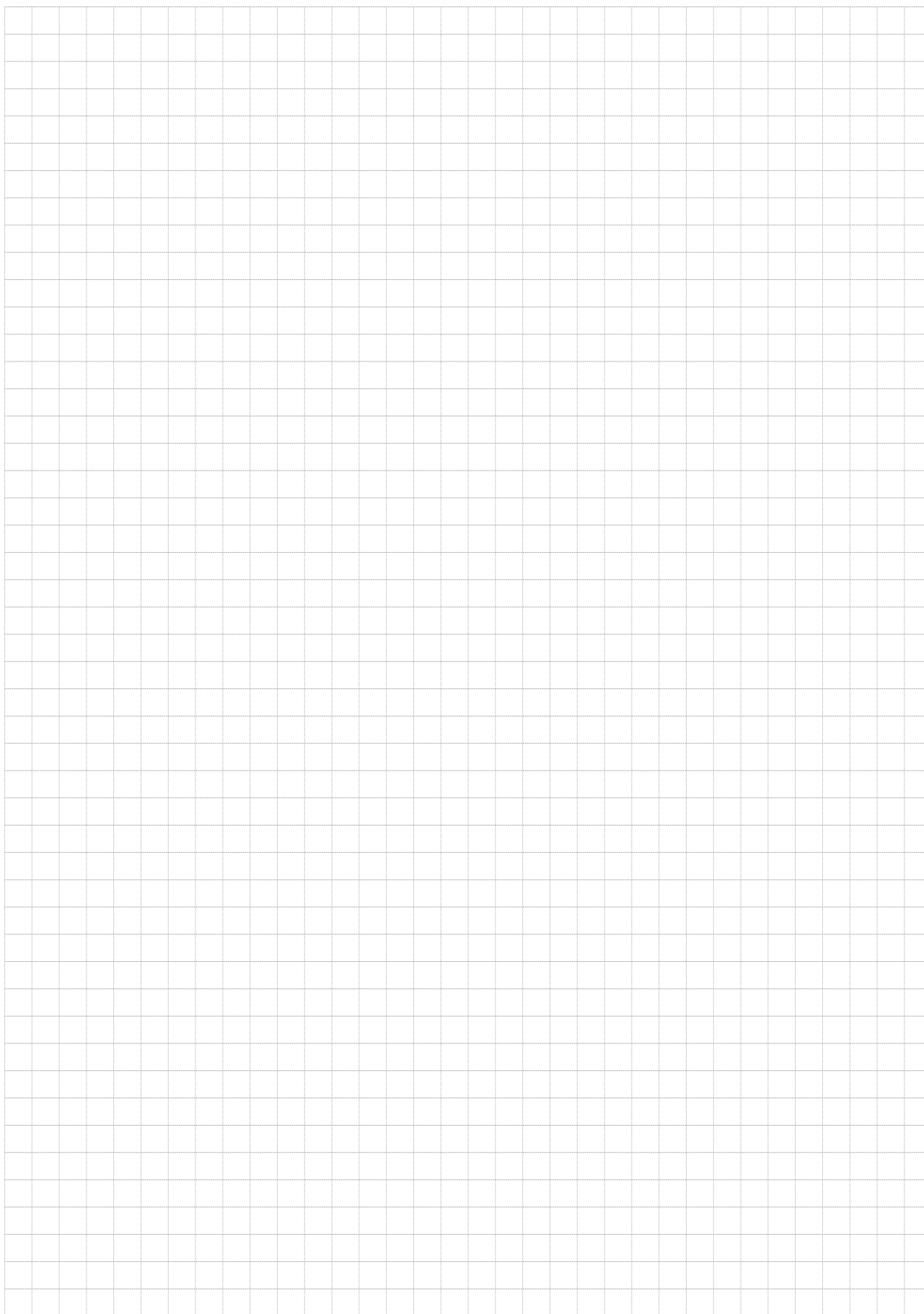
Variable message, cyclical sending 54

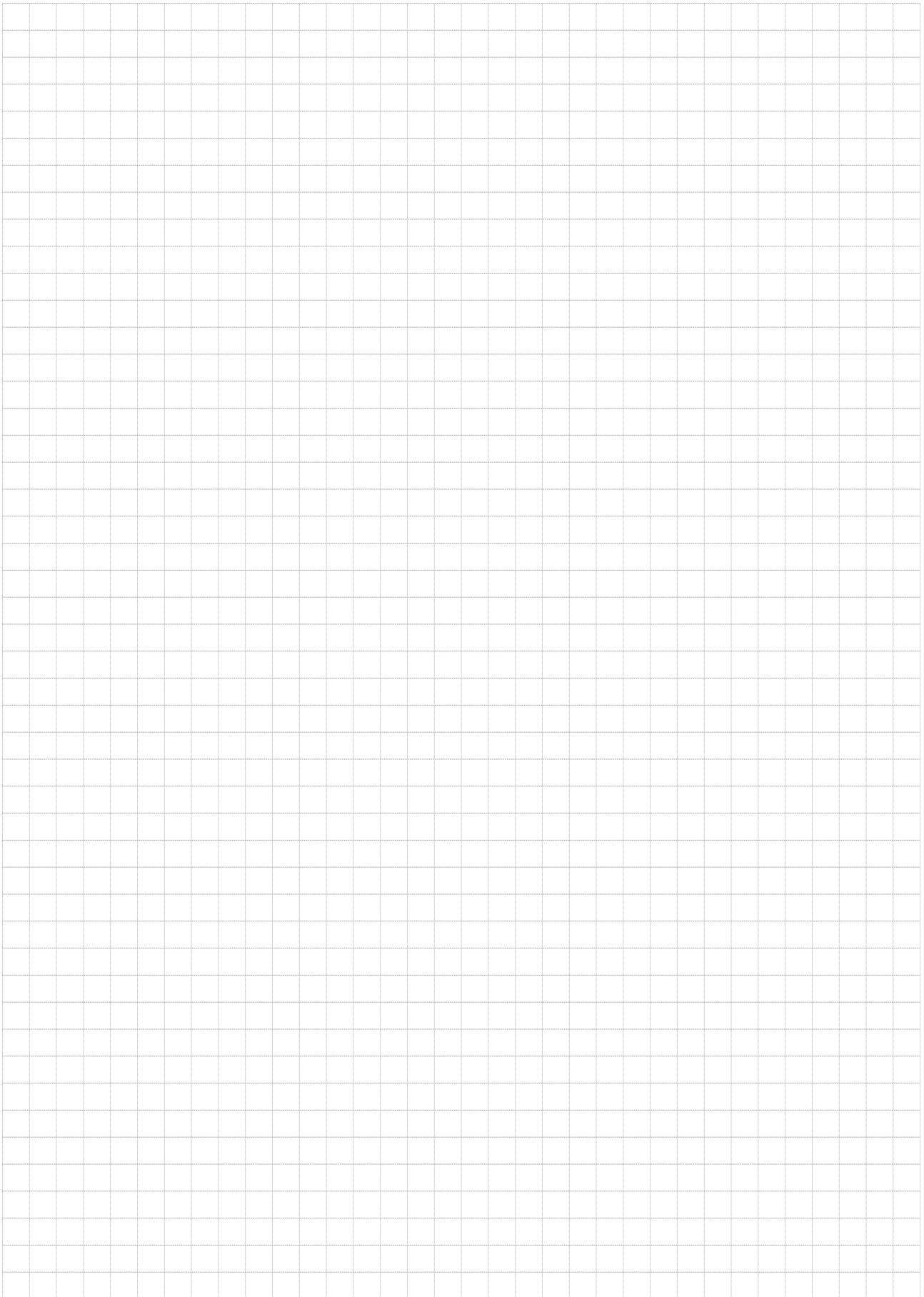
Variable message, receiving 55

Variable messages 51

W

Warning instructions 4





SEW-EURODRIVE GmbH & Co · P.O. Box 3023 · D-76642 Bruchsal/Germany · Phone +49-7251-75-0
Fax +49-7251-75-1970 · <http://www.sew-eurodrive.com> · sew@sew-eurodrive.com

SEW
EURODRIVE

