

## **MotionController**

**GEL 8230/8231**

**GEL 8235/8236**

Multifunctional Control with  
PLC Functionality in Accordance  
with IEC 61131-3



Editions to date:

Edition	Remarks
2002-05	First edition valid for firmware version $\geq 3.02$
2002-12	Adaptation to new firmware version 5.03 New: Auto scan and status display of the CAN buses – changes to various default settings and extensions of the system parameters – implementation of the LB2 protocol functions – integration of additional Field bus software modules – analog actual value inputs – inclusion of type GEL 8231
2003-01	Adaptation to new firmware version 5.05 New: Axis positioning with ramps – parameter editor BB 8200
2003-07	Adaptation to new firmware version 5.09 New: Control of stepper motors – changes to various default settings and extensions of the system parameters
2004-05	Adaptation to new firmware version 5.15 – incorporation of GEL 8235/8236 models New: limit switch function (system parameters) – signal states of encoder inputs – parameter editor 'Paraminator III' (replaces BB 8200) – Ethernet field bus module

Issued by:

**Lenord, Bauer & Co. GmbH**  
Dohlenstrasse 32  
46145 Oberhausen • Germany  
Tel: +49 208 9963-0 • Fax: +49 208 676292  
Internet: <http://www.lenord.de> • E-Mail: [info@lenord.de](mailto:info@lenord.de)

## Table of Contents

<b>1</b>	<b>General</b> .....	<b>5</b>
1.1	Safety instructions.....	5
1.2	Proper use.....	5
1.3	Device models.....	6
1.4	Characteristics .....	6
1.5	About this manual .....	8
<b>2</b>	<b>Assembly</b> .....	<b>10</b>
2.1	Scope of delivery.....	10
2.2	Mounting .....	11
2.2.1	GEL 8230/8231 .....	11
2.2.2	GEL 8235/8236.....	11
2.3	Removal .....	12
2.3.1	GEL 8230/8231 .....	12
2.3.2	GEL 8235/8236.....	12
2.4	Fitting a module.....	12
<b>3</b>	<b>Connections</b> .....	<b>14</b>
3.1	Wiring Instructions.....	14
3.2	Connector coding .....	15
3.3	Overall view.....	16
3.4	Voltage supply (V) .....	17
3.5	Serial interface (C1) .....	18
3.6	CAN bus (C2).....	19
3.7	Encoder inputs (E1/2/3).....	20
3.8	Digital inputs (I1/2/3) .....	21
3.9	Analog input (I6).....	22
3.10	PT100 analog inputs (I5, GEL 8231/8236 only) .....	22
3.11	Digital and analog outputs (Q1/2/3).....	23
<b>4</b>	<b>Operation</b> .....	<b>24</b>
4.1	The key pad .....	24
4.2	The display.....	25
4.3	The windows and menus.....	26
4.3.1	Menu structure .....	26
4.3.2	Main window (axes) .....	27
4.3.3	Main window (I/O) .....	28
4.3.4	Main menu .....	29
<b>5</b>	<b>Commissioning</b> .....	<b>35</b>
5.1	MotionController.....	35
5.2	Operating system update .....	37
<b>6</b>	<b>System parameters</b> .....	<b>41</b>
6.1	Explanations.....	41
6.2	Factory defaults.....	42

---

6.2.1	Reset parameter .....	42
6.2.2	Create parameter image .....	42
6.3	Basic settings .....	42
6.3.1	General .....	43
6.3.2	PLC .....	43
6.3.3	Serial communication (RS 232 C, RS 422/485) .....	44
6.4	Analog inputs .....	45
6.4.1	Analog input 1...7 (I61...I57).....	45
6.5	CAN bus.....	45
6.5.1	CAN 1, CAN 2.....	45
6.6	Actual value inputs .....	48
6.6.1	Actual value input 1...6 .....	48
6.7	Nominal value outputs.....	53
6.7.1	Nominal value output 1...6.....	53
6.8	Axis configuration.....	57
6.8.1	Axis 1...6.....	57
6.9	Extension module.....	69
6.9.1	DeviceNet .....	69
6.9.2	InterBus-S .....	69
6.9.3	PROFIBUS-DP.....	69
6.9.4	Ethernet .....	70
6.10	Numerical index .....	70
6.11	Application example .....	72
<b>7</b>	<b>Technical data .....</b>	<b>75</b>
7.1	Dimensional diagram.....	75
7.1.1	GEL 8230/8231 .....	75
7.1.2	GEL 8235/8236.....	76
7.2	Specifications .....	76
	<b>Appendix 1: LB2 protocol.....</b>	<b>79</b>
	<b>Appendix 2: "Paraminator III" parameter editor .....</b>	<b>91</b>
	<b>Index .....</b>	<b>93</b>

## 1 General

### 1.1 Safety instructions

- ▶ In order to make it easier for installation and for any possible servicing, the operating software of the MotionController permits **online change** of system parameters during **running operation**, therefore



**Be absolutely certain, when making use of this feature, that no dangerous situations can occur for persons or for the system itself in the system area, by ensuring that**

- **every parameter to be changed is conscientiously and carefully checked for correctness and applicability and for possible effects (above all for changes of velocities) before transmission,**
- **adequate electrical safeguards, such as limit switches and line breakers, are provided.**

**Please also observe the warning note on manually scanning the CAN buses on page 32.**

- ▶ Before carrying out a firmware update (see section 5.2, page 37), switch off the power circuit of all drives which are controlled by the MotionController.
- ▶ The assembly and commissioning of the MotionController may only be carried out by trained expert personnel, i.e. persons that are familiar with the safety concepts of electrical and automatic control techniques. (On request, product training is offered by LENORD+BAUER.)

### 1.2 Proper use

The MotionController is intended exclusively for the control of drives in the industrial sector. If you should make use of it in typical trade and residential zones you have to take additional measures like filters, HF-screened switching cabinets etc.

First and foremost it is a PLC, with which programs written by yourself or control programs supplied with it can be executed. In addition it can be used as an operator terminal for the control of operating states and for support during commissioning.

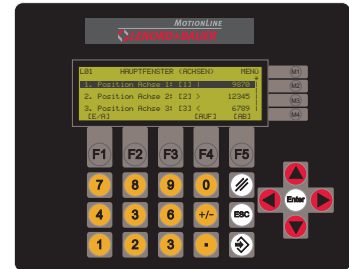
The data and instructions given in this manual must be complied with.

### 1.3 Device models

The **MotionController** device models described here only differ in their hardware equipment:

GEL 8230: Enclosure with **display unit** and **keypad** for installation into a control panel.

GEL 8231: Same as above, however, with:  
 + 2 analog inputs (terminal block I6),  
 + 4 analog inputs for PT100 resistors (terminal block I5),  
 + 8 digital inputs (terminal block I4).



GEL 8235: Enclosure **without control and display elements** for DIN top-hat rail or wall installation and accommodation within a switch cabinet.

GEL 8236: Same as above, however; with additional inputs as provided for GEL 8231.



**i** The **operating** and **observing functions** described in the following only apply to the **GEL 8230/8231** MotionControllers. For the basic configuration of the GEL 8235/8236 models the PC parameter editor must be used (cf. Appendix 2).

### 1.4 Characteristics

The scope of the display can be subdivided into two categories:

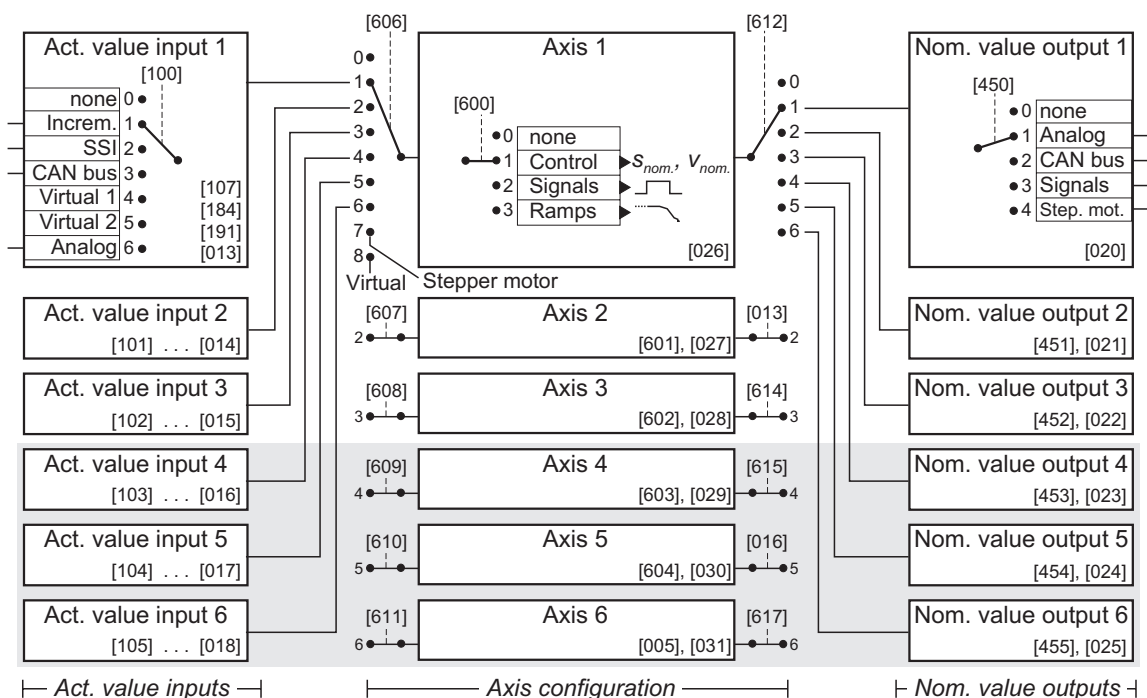
- ▶ Two so called **main windows** provide the following information:
  - Actual positions of up to 6 axes and as well as current data of one axis in a lower level (→ section 4.3.2)
  - Signal conditions on the input and output terminals  
 As part of this feature, designed as an aid for commissioning, is the possibility of operating the drive manually (→ section 4.3.3).
- ▶ Access is made to the following functions via a **main menu** (→ section 4.3.4):
  - Device information
  - System parameters with which the MotionController is user-specifically configured and which provide the necessary data for several function blocks (see below) in a PLC program

Besides the routines for the user interface, the MotionController's operating system also contains routines for the control and feedback control of up to six drives. These routines can be accessed by a PLC program that, by the use of the programming interface CoDeSys, can be developed according to the user's needs. For this purpose an extensive library with function blocks is made available, which are described separately (see section 5.1, page 35 also).

Besides the programming interface CoDeSys and the diverse libraries contained in the delivery of the MotionController, there are also example programs which illustrate the functionality of the MotionController and which can be adapted to the operating requirements. Besides this they are a source of useful help for designing your own PLC programs.

Apart from this, LENORD+BAUER will be offering ready-made PLC programs for various applications (e.g. jog sentence programming) in the foreseeable future. *Please enquire*

The Motion Controller contains 6 actual value inputs, nominal value outputs and axis controls which can be configured independently from each other and freely assigned to each other. The following block diagram illustrates the principle:



**Explanations:**

- Information in square brackets refers to the main system parameters for the corresponding blocks (see the relevant comment on page 9).
- For the actual value inputs/axes/nominal value outputs nos. 2...6 the same contents apply in principle as for no. 1;

differences occur with several factory settings and adjustable characteristics.

- Function elements with grey background are, by default (from the factory), de-activated.

## 1.5 About this manual



This document as well as any additional documentation are only available in electronic form as PDF files on the setup CD (\*.pdf, see also section 2.1) under "manuals" in the respective device folder. Hard copies are available for some fee.

Amendments or extensions which could not yet be included in the manual are described in the "readme.txt" file in the documentation folder.

The manual provides information on the installation and operation of the MotionControllers GEL 823x. It also deals, in brief, with the system parameters, by means of which you can configure the device and the controlled axes, and with the PC parameter editor provided on CD and the local serial LB2 protocol (→ appendices).

Descriptions of a PLC program that has possibly been pre-installed and of the library functions for the development of user programs are not handled here (separate manual under construction; the commentary texts in the declaration part of the individual function blocks provide some information).

### Symbols and designations used:



Paragraphs marked in this way provide important additional information to the subject.



Paragraphs marked in this way contain statements that are important for operation according to instructions.



This symbol points out critical situations or possible damage to the installation or property.

### **User measuring unit**

Position or velocity values for an axis are given in 'user measuring units' or 'user measuring units per second'. The user-specific actual position is derived from the measuring value provided by the actual value input according to the following formula:

$$A = R \cdot c \cdot \frac{m_n}{m_d} \cdot N + H - X$$

A: Actual value in user measuring units (displayed value)

R: Reading from the actual value input (incremental count or absolute value of an encoder)



- c: Factor of the edge evaluation (4 or 2 with incremental encoders [adjustable], else 1)
- $m_{n,d}$ : Multiplier (numerator, denominator) [adjustable]  $\Rightarrow$  resolution
- N: Negation (1/-1) [adjustable]
- H: Home offset [adjustable]
- X: Internal offset (will be adjusted when the actual value changes event-controlled, e.g. when calibrating the axis: calibration value  $\rightarrow$  actual value)

In addition, the display and input format for the position and velocity values may be adjusted by a programmable decimal point.

Example: incremental encoder with 2 counting tracks and 10,000 pulses per revolution (four-edge evaluation  $\Rightarrow$  40,000 counting pulses per revolution); required is a counting range of 360.00° per revolution. This yields the result:

Multiplier =  $36000/40000 = 0.900$ ;

Decimal point = "X.XX" for the required display resolution of 1/100.

All position inputs are then to be made in the user measuring unit '°', if necessary with two decimal digits (velocity in '° /s').

- [123]** A three digits number in square brackets designates a programmable system parameter, described in chapter 6, numerical index on page 70.

## 2 Assembly

### 2.1 Scope of delivery

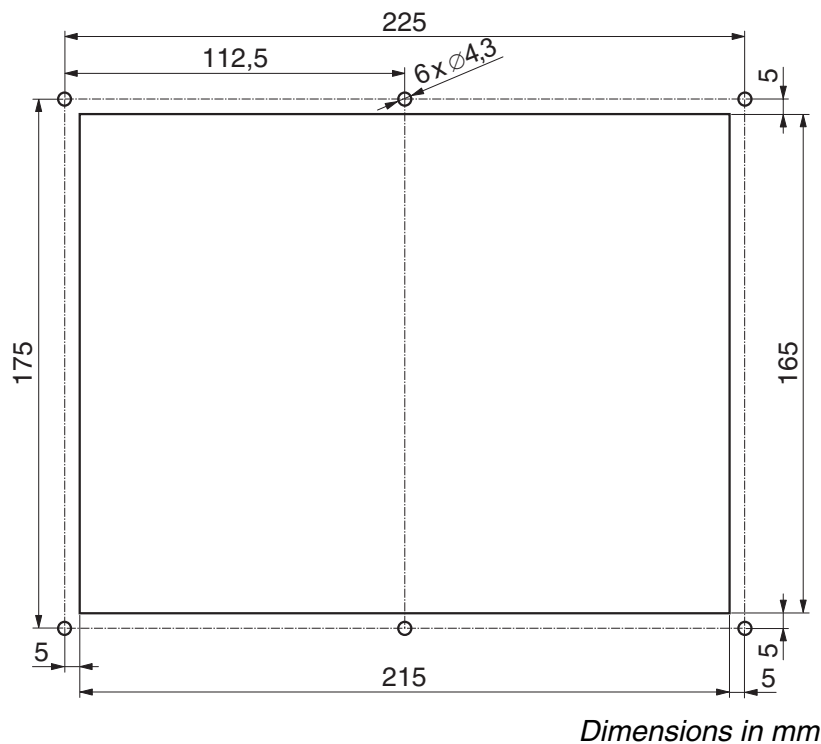
- MotionController GEL 823x
- Mounting kit with 6 nuts, spring washers and plain washers as well as 2 earth clamps (order no. BG 4623)
- Set of mating connectors (order no. GEL 89042 for the MotionController GEL 8230/8235, or GEL 89043 for the MotionController GEL 8231/8236)
- Set of cable clips (order no. BG 4622)
- Installation CD containing the following (may vary):
  - ◆ CoDeSys (PLC programming environment)
    - Set up
    - Libraries
    - Documentation (electronic manuals in various language versions as PDF files)
  - ◆ Updating software LingiMon (tool; see page 37)
  - ◆ MC8230 operating system, latest version (for downloading into the MotionController) and this manual in electronic form as a PDF file as well as configuration files and manuals for the extension modules
  - ◆ Acrobat Reader (tool)
    - Self-extracting installation program; the PDF files supplied can be read with the Acrobat Reader (Freeware from ADOBE SYSTEMS INCORPORATED)

The CD has an auto start program that offers information and various selection alternatives in a dialog window after inserting the CD, providing the auto start function is activated in the PC. If the CD does not start up automatically – but this is required so – an appropriate setting must be made in the system control; Windows Help provides the necessary information for this. The same effect as auto start is obtained by manually calling the program 'start.exe' in the CD root directory.

## 2.2 Mounting

### 2.2.1 GEL 8230/8231

#### Control panel cut-out



- Make the control panel cut-out in the required position in accordance with the dimensional drawing above
- Set the device in the cut-out
- Fasten the device with 6 M4 nuts, spring washers and plain washers
- Make the electrical connections while observing the following instructions

### 2.2.2 GEL 8235/8236

These devices are factory-prepared for DIN top-hat rail installation. You can use two clips already fitted to the base plate to install the device either vertically or horizontally: Hang the MotionController into the top of the DIN top-hat rail and click it in at the bottom.

To fit the device to a mounting plate (wall installation) it comes with corresponding brackets and screws (refer to the dimensional drawing in section 7.1.2).

## 2.3 Removal

### 2.3.1 GEL 8230/8231

- ☑ For a short-term removal of the MotionController loosen the four knurled thumb screws in the cover (they remain in the top and cannot get lost) and withdraw all connectors
  - ! After refitting the cover press on the knurled thumb screws and then tighten them up
- ☑ Remove the 6 nuts on the M4 threaded bolts of the Motion-Controller and remove the device

### 2.3.2 GEL 8235/8236

In case of DIN top-hat rail installation, pull off the device against the mechanical resistance at the bottom and remove it.

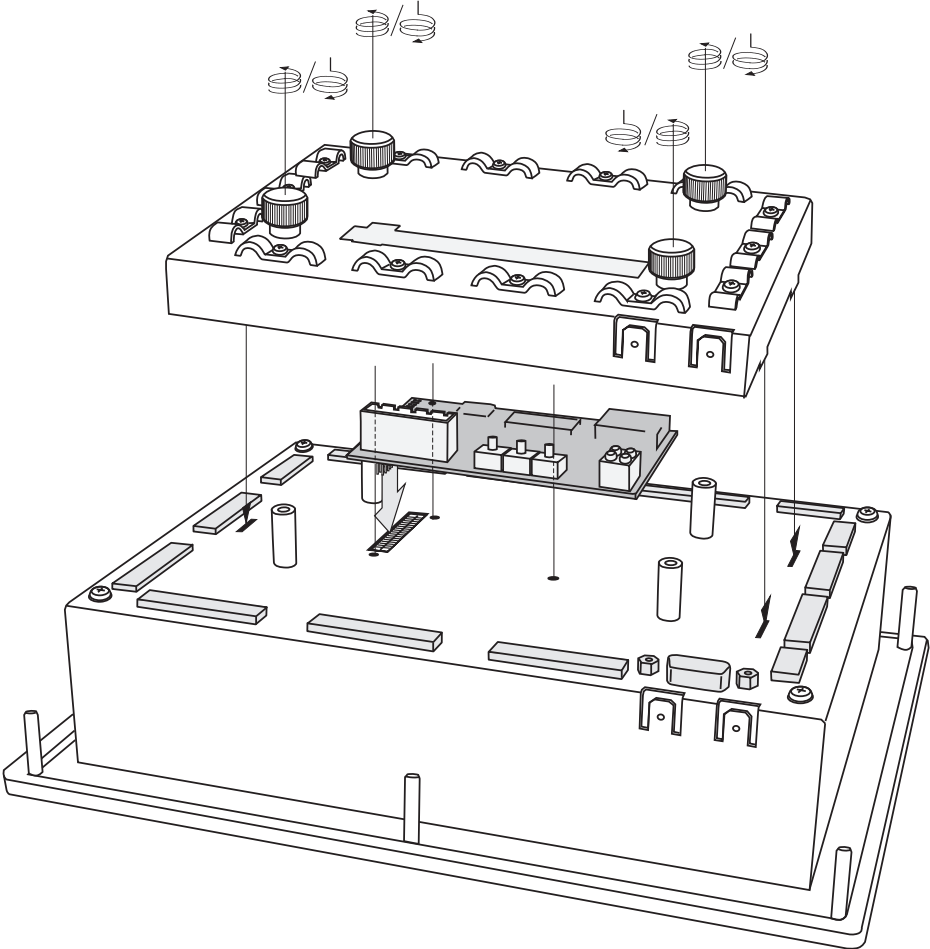
## 2.4 Fitting a module

The Motion Controller can be equipped with a functional module (Field bus) later. Fitting is done in accordance with the following illustration (example for CANopen module).

- ! Please make sure that all pins of the module are sitting correctly in the corresponding housing socket strip.

The connections and other information about the module can be found in a separate document (also in electronic form on the CD supplied with it under *GEL8230\Expansion modules*).

The extension modules are addressed via a PLC program by means of CoDeSys function blocks made available. The basic configuration is made via various system parameters (→ section 6.9).



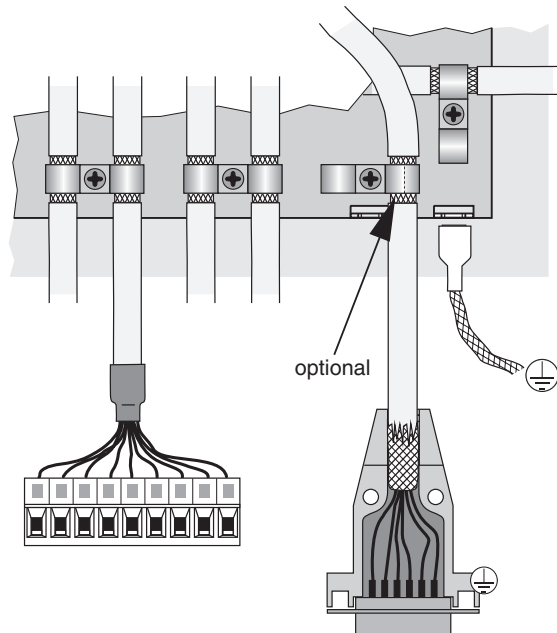
Fitting a module

## 3 Connections

### 3.1 Wiring Instructions

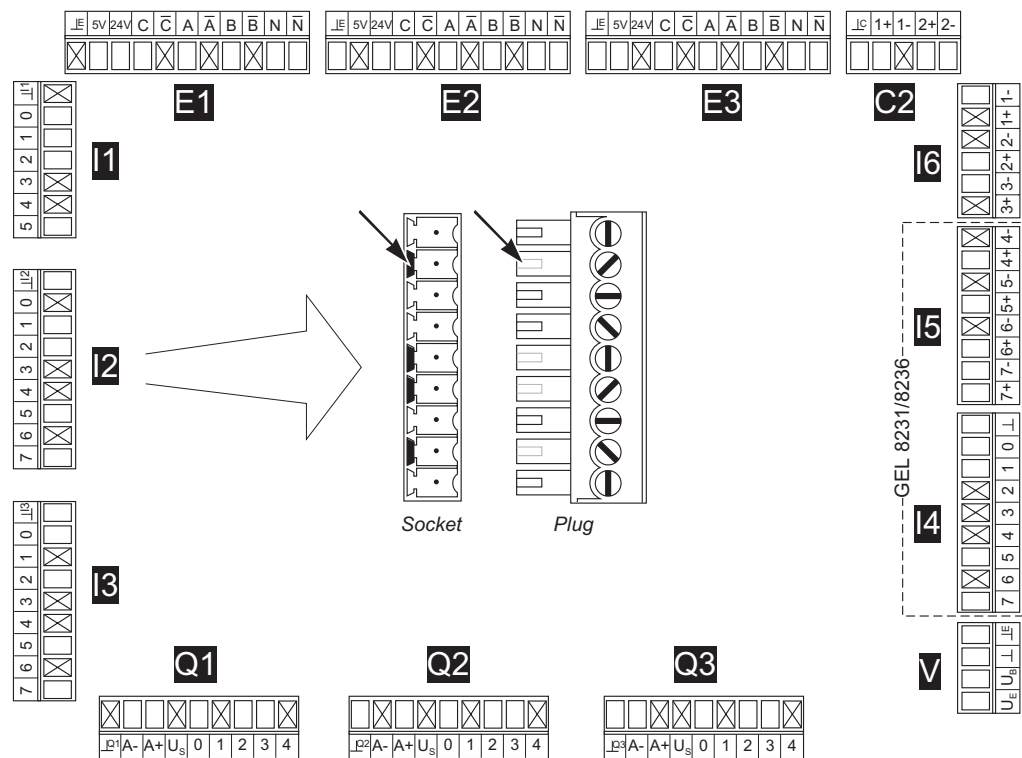
To improve the electromagnetic environment (EMC) please observe the following installation advice:

- ▶ Earth the device correctly: short connections via cable lug contacts (6.3 mm) housing/cover → cubicle (wall) as well as housing/cover themselves (use a low inductance earthing strap or flat band conductor)
- ▶ Earth connector cable at the device: uncover the cable screen at a suitable point and clamp it under a cable clip to the cover (see illustration)
- ▶ Use metallic sub D mating connectors and clamp the cable screen properly between the connector halves
- ▶ Tighten the fastening screws of the Sub D mating connectors firmly in order to guarantee a secure earth contact (it is recommended that this cable is earthed as well, as shown in the diagram)
- ▶ For good electrical contact and mechanical support of the cable in the terminals, as well as for a safe isolation of the bunched conductors used, provide isolated wire end ferrules in accordance with DIN 46228 part 4 that are captively joined to the conductors with the aid of special crimp pliers. If two or more thinner cables are to be connected to one terminal, the use of twin wire end ferrules is advantageous. The maximum connecting cross-section is 1.5 mm<sup>2</sup> (incl. ferrule).  
Use a screwdriver with a blade of 0.4 × 2.5 mm for tightening the terminal screws (M2).
- ▶ Keep all unscreened conductors as short as possible.
- ▶ Lay out the screens with large surfaces at both cable ends
- ▶ The power supply used must correspond to the type of installation class 0 or class 1 in accordance with item B.3 of EN61000-4-5 from 1995.

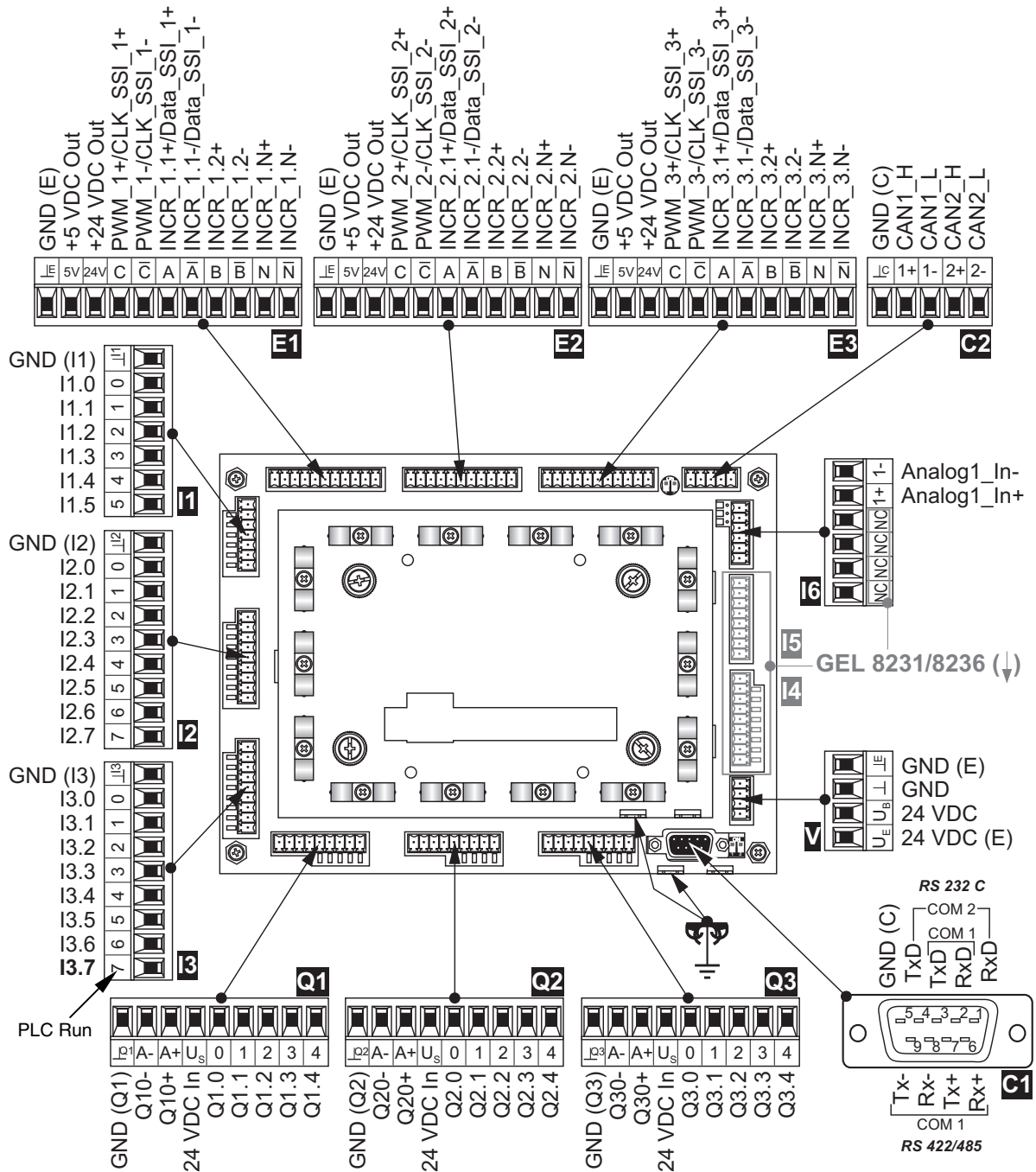


- ▶ Provide spark suppression for inductive loads (relays, contactors) on the digital outputs (free-wheel diodes or RC-networks in parallel and in direct proximity to the coils)
- ▶ Lay out signal and control cables separately from power cables.
- ▶ If there are potential differences between the machine and electronic earthing connections, or if they may occur, suitable measures have to be provided to ensure that no circulating currents can flow in the cable screen (e.g., lay potential equalization cables with large cross-section or use cable with double screening; in doing so, tie the screen time to one side only in each case).

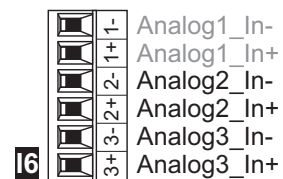
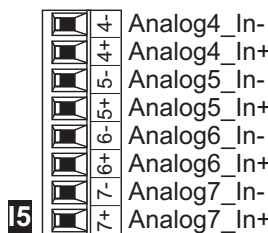
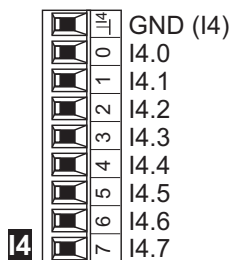
### 3.2 Connector coding



3.3 Overall view



**GEL 8231/8236:**

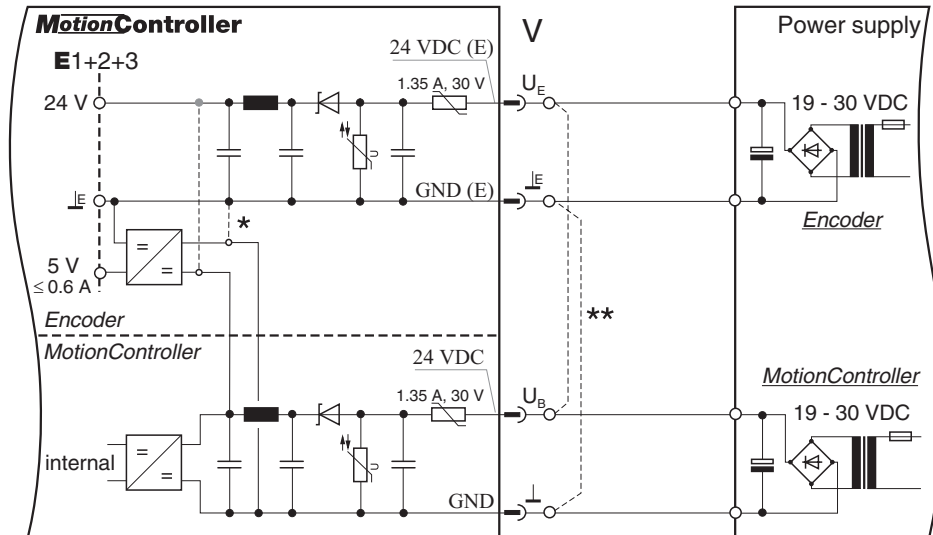




### 3.4 Voltage supply (V)



Both the voltage for the internal circuits of the MotionController ( $U_B$ ) as well for the supply of the encoders connected ( $U_E$ ) are supplied via terminal block V



- \* An older version exists where the **5 V** encoder supply on terminals E1/2/3 is not generated from the 24 V MotionController supply but from the 24 V encoder supply (sketched in dashed lines). In this case 24 V must be supplied to the terminals  $U_E/\perp E$  in order to get the desired 5 V power supply (see also the following note).
- \*\* If only one single voltage source is to be used for supplying the MotionController and the encoder(s) connect the terminals  $U_E/\perp$  and  $U_B/\perp$  as sketched in dashed lines.

### 3.5 Serial interface (C1)

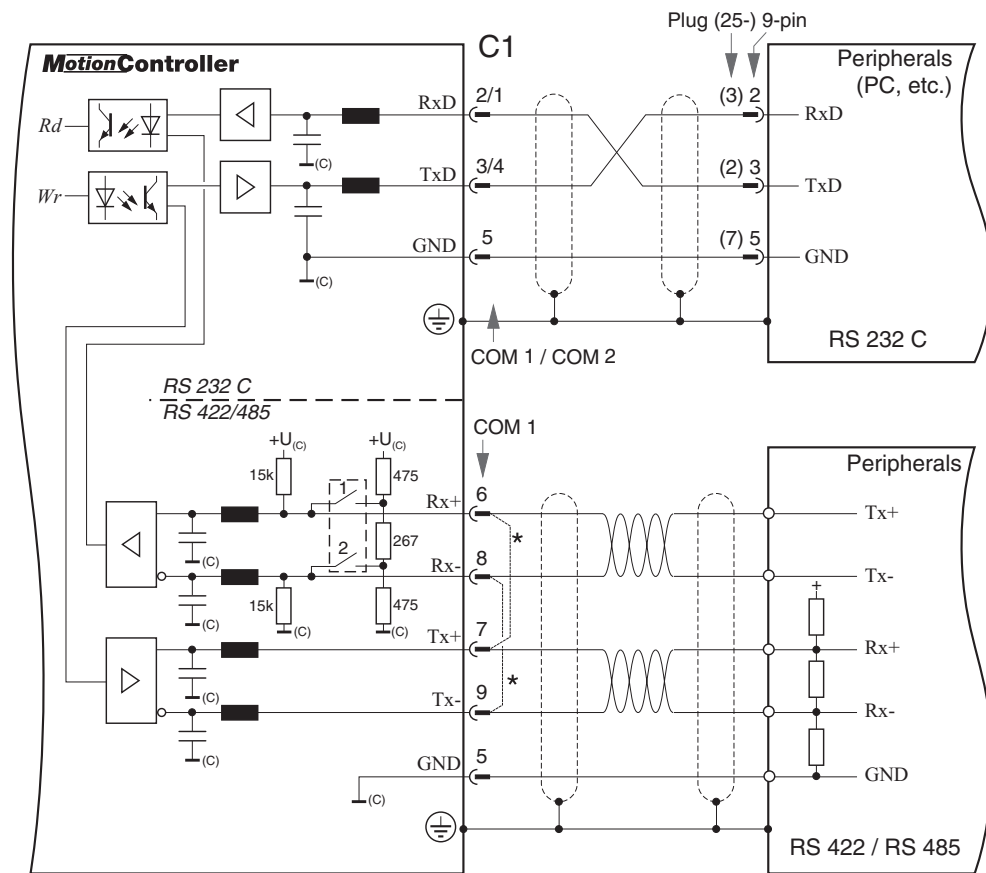


Two independent serial interfaces are available on connector C1: COM 1 and COM 2

COM 2 is designed as a pure RS 232 C interface.

COM 1 can be used either as RS 232 C or RS 422/485.

With the RS 422/485 the internal terminating resistor must be switched in at the first and the last device on the bus with the DIP switches next to the connector: both switches in the ON position. The resulting terminating resistance is approx. 125 Ω.



\* Connect pins for a RS 485

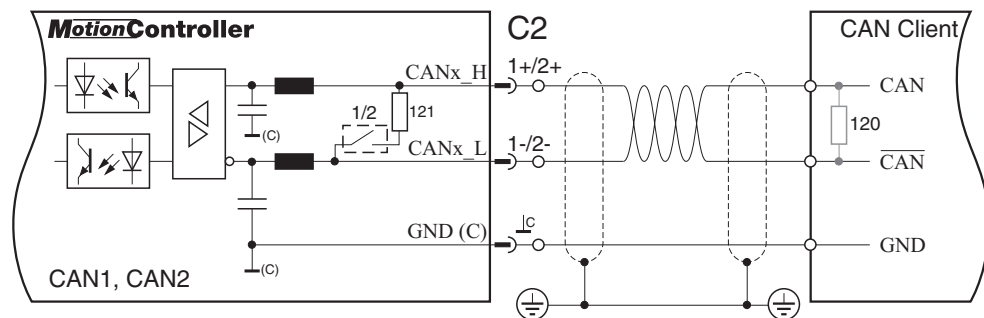
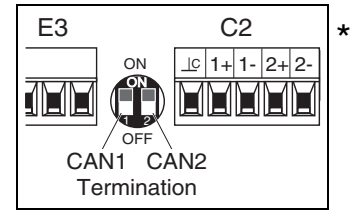
The serial interface has the same ground potential as the CAN buses (connector C2).

### 3.6 CAN bus (C2)



Two CAN bus interfaces are available on connector C2: CAN 1 and CAN 2.

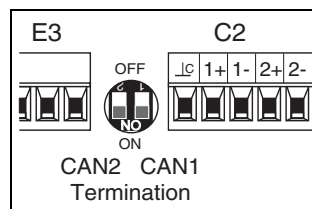
With each of the first and last device on the bus the terminating resistor for the associated CAN bus must be switched in by the DIP switch next to the connector: switch in position ON. The terminating resistance is approx. 120 Ω.



The CAN buses have the same ground potential as the other serial interfaces (connector C1).

CAN 2 is used solely for the control of servo amplifiers of type LD 2000.

\* For some (older) devices there is a second variant (DIP switch "upside down"):



### 3.7 Encoder inputs (E1/2/3)

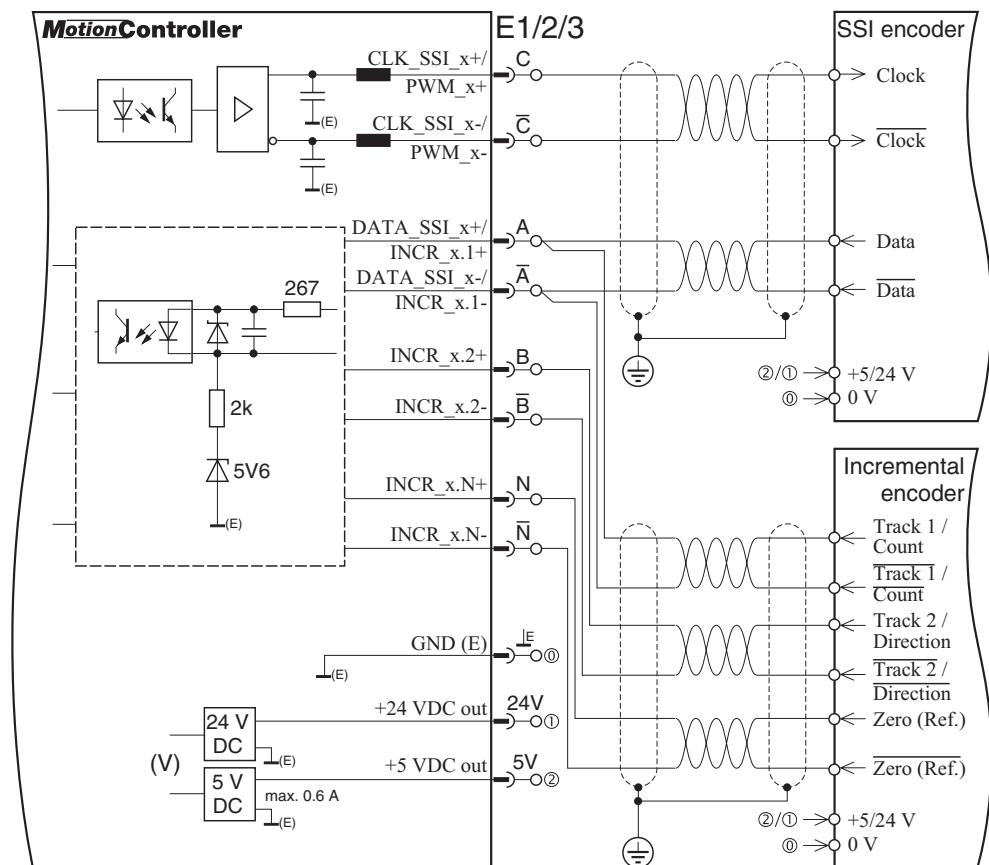


SSI or incremental encoders can be connected to the terminal blocks E1...3.

Here, 5 V and 24 V are available as supply voltage for the encoders (see also terminal block V). The maximum load is

- for 24 V: 300 mA per encoder
- for 5 V: 200 mA per encoder, 0.6 A in total

All encoder inputs use the same ground potential.



**i** • Inverse inputs (INCR<sub>x-</sub>):

If the incremental encoders used have no inverse signal output, the inverse inputs must be connected to ground with the 5V version. For the 24V version they **must** be let open.

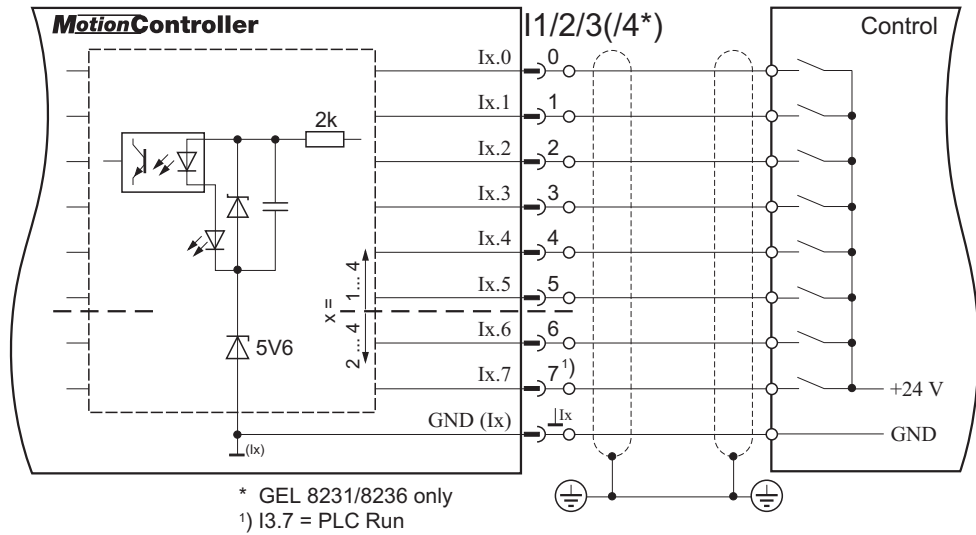
• Outputs for pulse width modulation (PWM<sub>x</sub>, terminal C):

These terminals can be addressed from a PLC program by means of CoDeSys function blocks made available, e.g. for control of stepper motors (configuration also via the operating system: see section 6.7.1.4 on page 57).

### 3.8 Digital inputs (I1/2/3)



The signal state of each of the 6 (I1) or 8 (I2/3/4) digital inputs is indicated by a green LED below the respective terminal (ON  $\triangleq$  High).



**i** The assignment, i.e. the functional reservation, of the digital inputs is made in the respective user/PLC program. Exception: **I3.7** (PLC Run) = Start of the PLC program.

### 3.9 Analog input (I6)



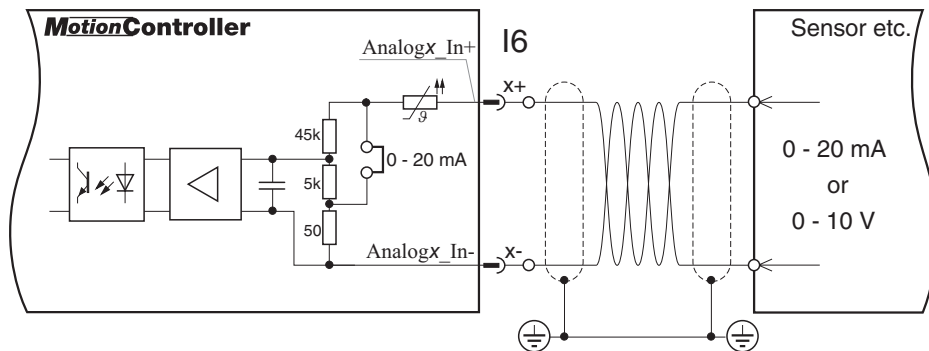
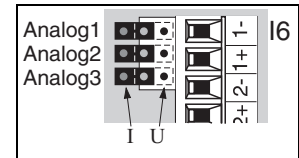
The analog input can be set for the measurement of currents **0 ... 20 mA** or voltages **0 ... 10 V**

For this a jumper is provided next to the terminal block for every one of the max. 3 inputs.

Default configuration: **current** measurement (I).

For voltage measurement (V) change the corresponding jumper. For this, the device cover must be dismantled: remove the 4 corner screws.

The measuring ranges are converted to digital form with a range of 0...1023.

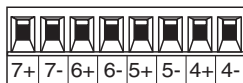


Tip

For stabilizing reasons of the differential input we recommend to connect the encoder's external supply ground to the analog ground of the MotionController. Connecting options:

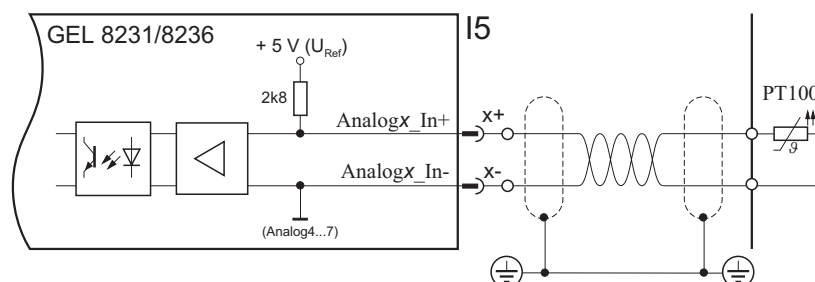
- ▶ Terminal block I5, terminals 4-/5-/6-/7- or
- ▶ Terminal block Q1/2/3, terminal A-.

### 3.10 PT100 analog inputs (I5, GEL 8231/8236 only)



These special analog inputs are provided for use with PT100 resistors temperature measurement in the range of -40 °C ... +351 °C.

The measuring range is converted to digital form with a range of 0...1564 ( $T [^{\circ}\text{C}] = \text{digital value}/4 - 40$ ).



### 3.11 Digital and analog outputs (Q1/2/3)



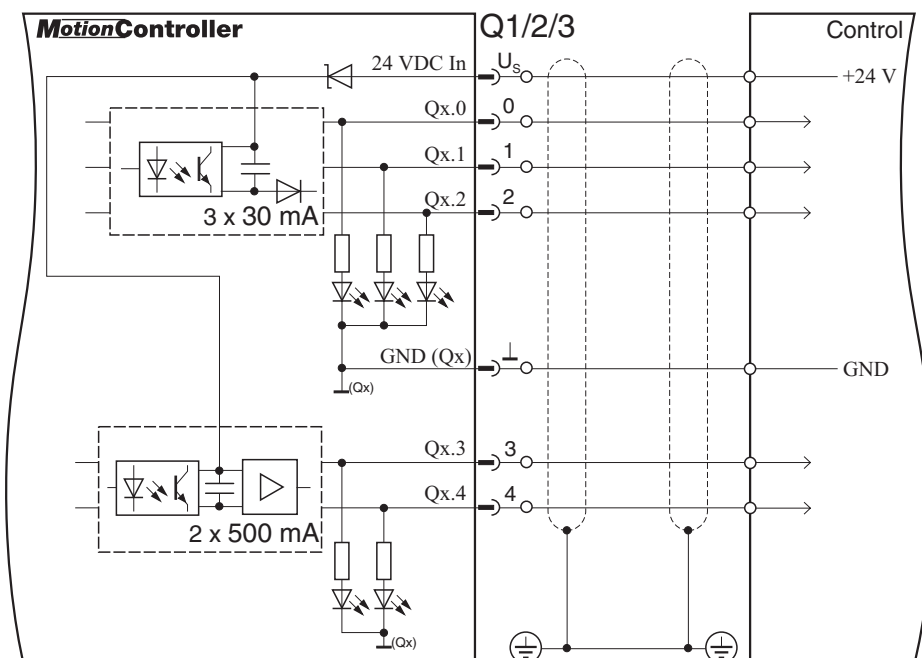
Each of the 3 output terminal blocks Q contain:

- 3 digital outputs 30 mA (terminals 0, 1, 2)
- 2 digital outputs 500 mA (terminals 3, 4)
- 1 analog output for axis control (terminals A+, A-)

The digital outputs on a terminal block each have their own ground potential. The minus connections of the analog outputs are connected to each other.

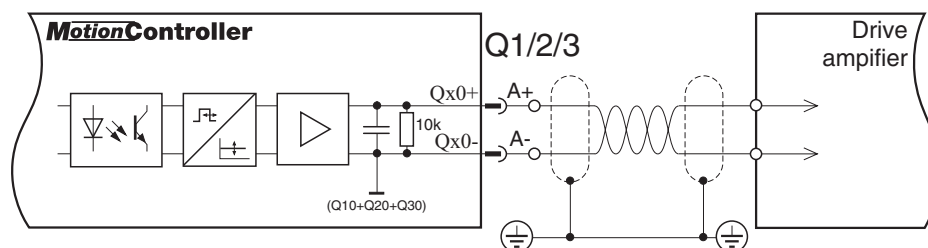
The signal state of the digital outputs is indicated by a red LED below the respective terminal (ON  $\triangleq$  High).

Digital:



**i** The assignment, i.e. functional reservation, of the digital outputs is made according to the application in a user/PLC program (exception: see parameter [528], page 56).

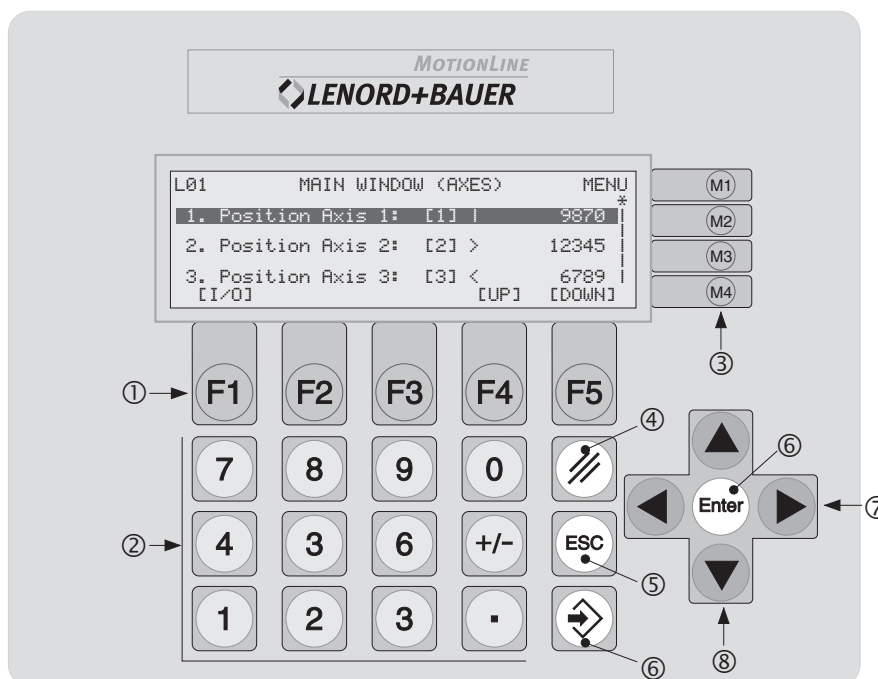
Analog:



## 4 Operation

(GEL 8230/8231)

### 4.1 The key pad



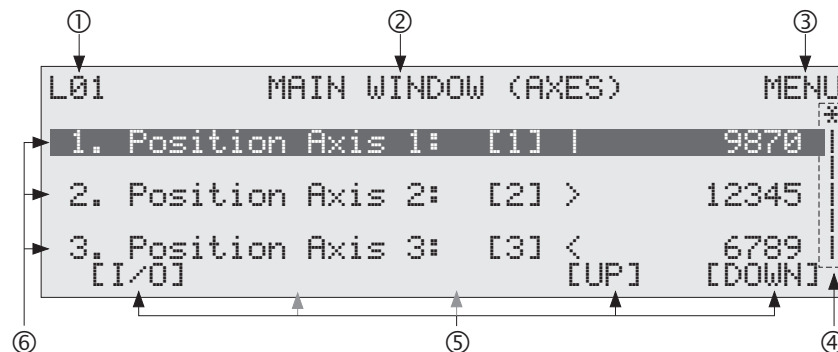
- ① Function keys (assignment dependent on the current window)
- ② Numerical keys (value input)
- ③ Menu keys (assignment dependent on the current window, line orientated)
- ④ Delete value input
- ⑤ Cancel input/function; return to next higher menu level
- ⑥ Confirm input, select/call marked entry (doubly available)
- ⑦ Select keys (select characteristic of a system parameter)
- ⑧ Scroll keys (move window within the displayable list by one line upwards/downwards)

(The labelling of the key blocks ① and ③, as well as the company logo, can be adapted to the application by using customized slide-in strips, see section 7.1.1 on page 75.)



## 4.2 The display

Example:



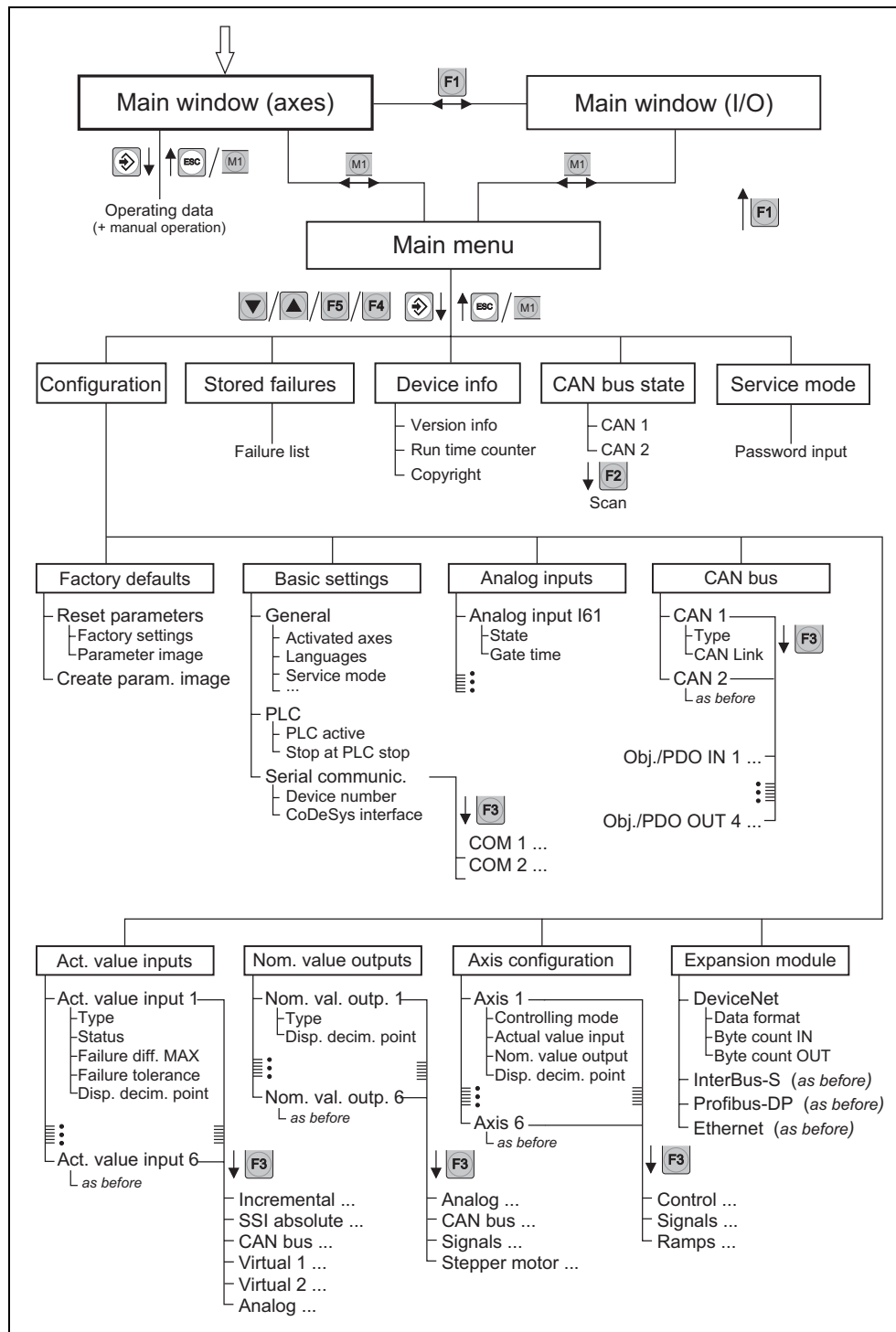
- ① Number of the current menu level; in some information windows alternatively number of the window just displayed with the number of total windows (e.g. 01/03 in the first of three possible display windows)
- ② Name of the menu/function window
- ③ Function of the menu key  $\boxed{M1}$ ; with the other menu keys the list entries shown on the left can be activated in some windows ( $\boxed{M2}$  → 1<sup>st</sup> list entry etc.)
- ④ Scroll bars: information (qualitative) about the position of the current window (\*) within the displayable list (l)
- ⑤ Function of the  $\boxed{F1}$ ... $\boxed{F5}$  keys  
(here 'I/O' →  $\boxed{F1}$ : display of the input/output states;  
'UP' →  $\boxed{F4}$ : scroll 1 window upwards; 'DOWN' →  $\boxed{F5}$ : scroll 1 window downwards)
- ⑥ List entries  
An entry with a dark background can be selected with the enter key (see previous section, point ⑥) i.e. it can be activated; the menu keys  $\boxed{M2}$ ... $\boxed{M4}$  are also used for this (in the example above this is the  $\boxed{M2}$  key)

Further explanations about the various windows you will find in the following descriptions of the menus.

## 4.3 The windows and menus

### 4.3.1 Menu structure

For the operating and observing of the MotionControllers GEL 8230/8231 various hierarchically classified display windows and setting menus are available. The following diagram gives an overall view of the menu structure.



**i** The numbers 01...07 indicate the respective menu depths that correspond to the representation of the menu level in the display (e.g. 03  $\hat{=}$  L03).

After switching on the device and after a possible CAN bus scan (see section 4.3.4.4, page 31) 'MAIN WINDOW (AXES)' appears as the standard display showing the actual value states of the axes; see the next section.

In parallel to this there is the 'MAIN WINDOW (I/O)' that gives information about the signal states of the digital and analog I/O and the encoder inputs of the MotionController; see section 4.3.3.

### 4.3.2 Main window (axes)

```

L01          MAIN WINDOW (AXES)          MENUE *
1. Position Axis 1: [1] | 9870
2. Position Axis 2: [2] > 12345
3. Position Axis 3: [3] < 6789
   [I/O]          [UP] [DOWN]
  
```

F1      F2      F3      F4      F5





M1  
M2  
M3  
M4

Example:

```

2. Position Axis 2: [2] > 12345 |
                ↑   ↑   ↑
                ①  ②  ③
  
```

- ① Number of the actual value input assigned to axis 2 (see parameter [606], page 58), here no. 2 (type defined in parameter [100], see page 48); [-] = not assigned, [U] = virtual function, [S] = stepper motor
- ② Moving state of the axis: | = standstill, > = forward movement, < = backward movement
- ③ Value of the actual position (encoder → 4/2 way edge evaluation with incremental encoders → multiplier → decimal point ⇒ user measuring unit, see page 8)

- Possibilities
1. Browse with / and /
  2. Display operating data about the marked axis with [Enter] or [M3] (as shown above in the window), here, as example, for axis 2:

```

          OPERATING STATE AXIS 2          BACK
State:   START
Encoder/Dir.: [2] >
Pos (n/a): 15000 12345
Velocity (n/a): 500 499
DeltaS: -3
Output (v/U): 3.456 U
[<<] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ]
  
```

F1      F2      F3      F4      F5

M1  
M2  
M3  
M4

Explanations of the display:

- Pos = Position, n/a = nominal/actual, Velocity = velocity in increments per second)
  - DeltaS = tracking distance (= difference between the desired position value and the actual position), see control principle on page 59
  - Output (v/U) = automatic control speed and voltage at the analog output of axis 2
  - The drive can be operated manually with the function keys if the corresponding system parameters have been correctly configured (see section 6.8.1.1, page 59):
    - Ⓜ1: Fast jog in backward direction
    - Ⓜ2: Slow jog in backward direction
    - Ⓜ4: Fast jog in forward direction
    - Ⓜ5: Slow jog in forward direction
  - Ⓜ1: Return to the MAIN WINDOW (AXES)
3. Call information about the inputs and outputs of the Motion-Controller: Change to the WINDOW (I/O) by means of Ⓜ1; see next section
  4. Call device information or configure system parameters: Change to the MAIN MENU with Ⓜ1; see section 4.3.4

### 4.3.3 Main window (I/O)

Activating Function key Ⓜ1 in the MAIN WINDOW (AXES)

L01	MAIN WINDOW (I/O)	MENU	Ⓜ1
1. Input I1.5..I1.0:	100101	[25]	*
2. Input I2.7..I2.0:	00010010	[12]	
3. Input I3.7..I3.0:	00000000	[0]	Ⓜ3
4. Input I4.7..I4.0:	10100000	[A0]	
[AXES]	[UP]	[DOWN]	Ⓜ4
	Ⓜ1		
	Ⓜ2		
	Ⓜ3		
	Ⓜ4		
	Ⓜ5		

Example:

2. Input I2.7..I2.0:	00010010	[12]	
	①	②	③

- ① Digital inputs 0 ... 7 on terminal block I2
- ② Logic states of the 8 inputs in bit form, LSB (right)  $\hat{=}$  state of I2.0; 1 = High  $\hat{=}$  approx. 24 V (definition of the logic states see Technical data, section 7.2)
- ③ Logic states in hexadecimal format

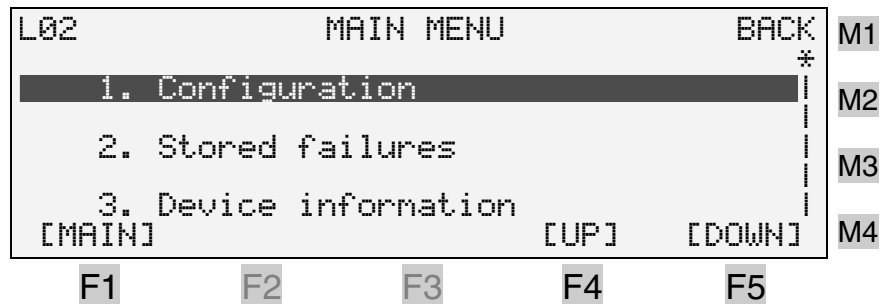
- Possibilities
1. Browse with / and /
  2. Display actual values of axes: Change to MAIN WINDOW (AXES) with ; see section 4.3.2
  3. Call device information or configure system parameters: Change to MAIN MENU with ; see following section

### 4.3.4 Main menu

**Entries**

- ▶ Configuration (4.3.4.1) .....29
- ▶ Stored failures (4.3.4.2) .....30
- ▶ Device information (4.3.4.3) .....30
- ▶ CAN bus state (4.3.4.4).....31
- ▶ Service mode ON/OFF (4.3.4.5) .....34

Activating Menu key in the MAIN WINDOW (AXES) or MAIN WINDOW (I/O)



- Possibilities
1. Browse with / and /
  2. Activate marked menu point with and process the next sub-point (see following sections); one of the menu keys ... can also be used for activating
  3. Change to the MAIN WINDOW (AXES) or MAIN WINDOW (I/O) with , or ; see section 4.3.2/4.3.3

#### 4.3.4.1 Configuration

For the change to the configuration mode a password must first be entered:

**9 2 2 8**

The password request can be deactivated:

- a) chronologically unlimited by programming the system parameter "Service mode" in the Main menu/Configuration/-Basic settings/General to ACTIVE (see page 43)

- b) for the duration of service (i.e. as long as the device remains switched on) by activating the service mode in the main menu (see page 34) with the password

**8 1 1 7**

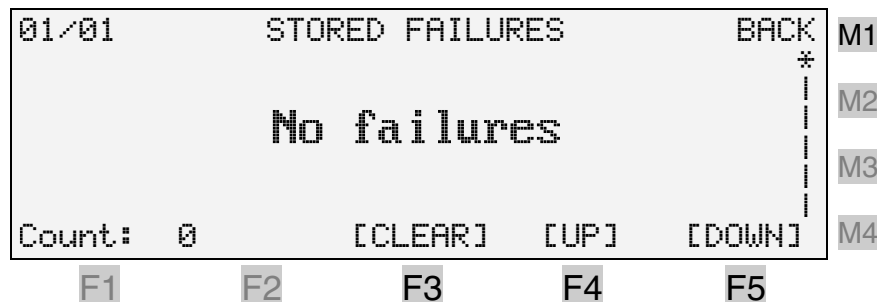
Description of the system parameters → **Chapter 6** (from p. 41)

PC parameter editor → Appendix 2 (from page 91)

#### 4.3.4.2 Stored failures

Here a list of up to 20 failures that have occurred can be called up.

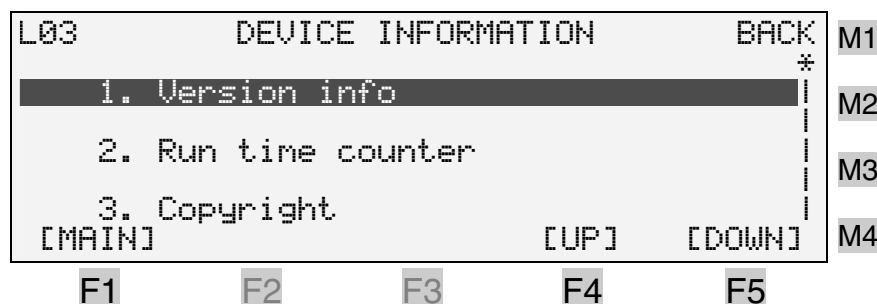
If no failure has occurred (or after deleting the list), the following message appears:



The following failures are registered:

- DeltaS > DeltaS max. → maximum tracking distance exceeded (see parameter [696], page 61)
- DeltaS < DeltaS min. → minimum tracking distance is fallen short of (see parameter [702], page 61)
- General data transmission failure (stop bit, parity, overwriting or checksum error)
- Error in the LB2 protocol (82h repetition missing, receive count error), see Appendix 1 (from page 79)

#### 4.3.4.3 Device information

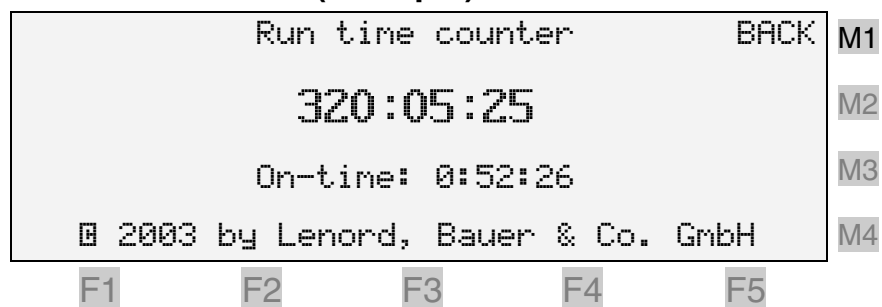


## ► Version info (example):

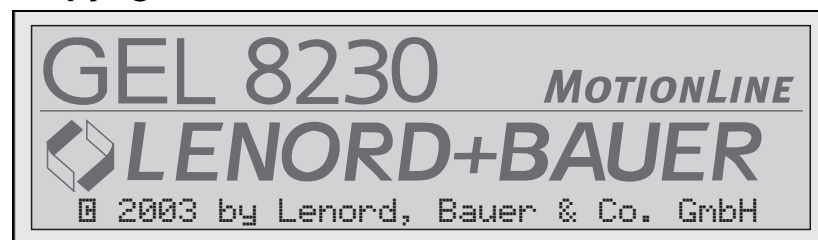


STD = standard, SSW = special software; CoDeSys: library versions, BAS = MC8230\_Basic22.lib, HMI = MC8230\_HMI\_Basic22.lib

## ► Run time counter (example):

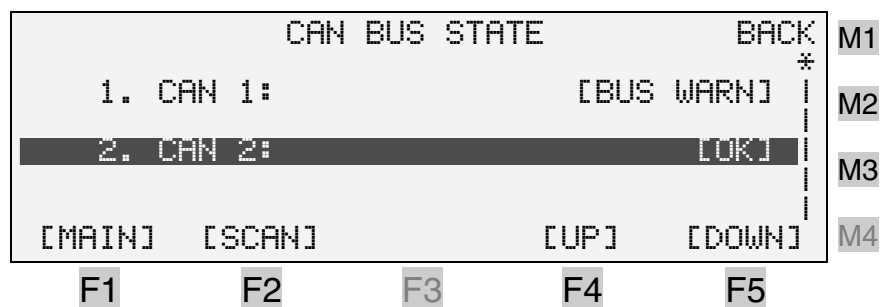


## ► Copyright:



This display also appears directly after switching on the MotionController; this can be prevented with system parameter [32] (see page 43).

## 4.3.4.4 CAN bus state



The CAN buses can be selectively read in again with **F2**. This procedure takes place automatically after switch-on in case a

CAN bus was activated by the system parameters (node address set to a value 1...16 and object type  $\neq$  'Inactive').

If the object type with an Object/PDO IN is set to 'Axis' (see page 46, [326])

- the type of the accompanying actual value input must be set to 'CAN bus' (see page 48, [100]) and
- its Object/PDO-IN characteristic must refer to the Object/PDO IN of the CAN bus (see page 51, [254]),

in order that these objects are taken into consideration during scanning.

The same applies to the output objects Object/PDO OUT in connection with the nominal value outputs (see pages 53, [450], and 55, [486]).

Objects of type 'Other' are not registered during scanning.

The application example in section 6.11 (page 72) explains the contexts.

After changing a CAN system parameter the scan procedure must be started manually, **but be careful:**



When manually starting the scan function please notice that

- PLC processing and
- the feedback control

are **stopped** during the scan procedure (although the Node Guarding telegram will still be generated for the first 16 node addresses and the Sync telegram sent every 10 ms if the MotionController is configured as master).

Therefore, the scan function was provided with a password to prevent unintentional calling ('9228', as for the configuration menu, with the same possibilities for circumventing this, see section 4.3.4.5, page 34).




**Only call the scan function when all axes are**

- **in the stopped operating state (without position control)**
- and**
- **at standstill.**

BUS WARN signalises that a CAN bus cannot be addressed because it is either not activated or e.g. the slave is not switched on. For the latter case a manual scan procedure must be carried out after switching on the slave.

OK shows that a physical connection has been set up.

 on a marked CAN bus entry provides information about the nodes connected in the address range from 1 to 16:



CAN2	In	Out	Profile	Name	OP	dI	dO	BACK
Nd01	1	1	0192h	SD03	00h	dI:T	dO:T	*
Nd02	2	2	-	-	-	dI:F	dO:F	
Nd03	3	3	-	-	-	dI:F	dO:F	
Nd04	4	4	-	-	-	dI:F	dO:F	
[MAIN]					[UP]	[DOWN]		

F1
F2
F3
F4
F5

Explanations:

CAN2 = marked CAN bus

In, Out = object/PDO number assigned to the node Nd

Profile = CANopen profile:

0191h (= DSP401) = CAN remote I/O terminal

0192h (= DSP402) = servo amplifier

Name = manufacturer or device name

(e.g. SD03 = Servo Drive LD 2000, 3 A)

OP = Operating mode (OPMODE) with the LD 2000

(00h = "Digital (rotational) speed ")

dI, dO = new input/output data; available = T rue,

none = F alse

If dI = F and dO = T are displayed with the master, the slave addressed may have been lost for a short time (e.g. because the supply voltage was too low). In spite of this, OK is signaled for the CAN bus state. In this case a scan procedure must be initiated.

Detailed information about the marked nodes (values of the 8 input and output bytes in the Object/PDO) can be had with :

CAN2 ND01: STATE	BACK							
	b0	b1	b2	b3	b4	b5	b6	b7
Inputs:	92	94	03	00	00	00	00	14
Outputs:	00	00	00	00	00	00	00	00
[MAIN]								

F1
F2
F3
F4
F5

The output bytes depicted are stored values, not current values read from the node. They are written into a buffer memory designated for this by the control or from a CoDeSys function block.

#### 4.3.4.5 Service mode ON/OFF



For the duration of service, i.e. as long as the device remains switched on, the request for a password to the configuration menu can be suppressed by means of the following password:

**8 1 1 7**

Hence, for continuously changing in and out of the configuration menu, i.e. while commissioning, the required password does not need to be entered again and again.

The other possibility of permanently deactivating the password request by means of a system parameter (under Basic settings/General, see page 43), has the danger that it may be forgotten to cancel this setting afterwards. If this happens there is the possibility of unlimited access to all system parameters – even for unauthorized persons.

## 5 Commissioning

### 5.1 MotionController

! Before switching on the MotionController for the first time it must be ensured that the connection I3.7 (PLC Run) is at low potential in order to prevent a stored PLC program from being started immediately.

After mounting and making all the necessary electrical connections, the MotionController can be switched on by applying the supply voltage to the terminal block V (UB/⊥).

After a short initialization period (perhaps with CAN-Bus scan, see section 4.3.4.4) the MotionControllers GEL 8230/8231 show the following display:

```

L01          MAIN WINDOW (AXES)          MENU
-----
1. Position Axis 1: [1] | 0
2. Position Axis 2: [2] | 0
3. Position Axis 3: [3] | 0
  [I/O]                [UP]  [DOWN]
  
```

Individual adaptations for the MotionController and the axes to be controlled can now be carried out. The system parameters listed in chapter 6 serve this purpose.

Configuration as delivered (factory defaults see also the principle in the diagram on page 7):

- Activated axes: 3;            an actual value input and a nominal value output always assigned with the same number
- Actual value inputs: 3;    1...3: incremental
- Nom. value outputs: 3;    1...3: analog
- CAN-Objects/PDOs (Master)
  - CAN 1: all inactive
  - CAN 2: IN/OUT 1...4: addr. 1...4, type: axes

After the setting of all axis-dependent control parameters the drives can be operated manually via the keyboard of the GEL 8230/8231 (see section 4.3.2, page 27).

If the axes control is done by a PLC program, commissioning of the axes, i.e. the setting of the system parameters, is usually carried out by the program.

If a PLC program is already stored in the control unit, this can be started with a high level input to terminal I3.7 (PLC Run).

For drawing up user PLC programs the programming environment CoDeSys must first be installed: run Setup from the corresponding directory on the CD or simply click on the corresponding menu point in the CD start program.

! The CoDeSys version 2.2 supplied requires the operating system firmware  $\geq 5.00$  described here.

The previous version (2.1) can still be used, however internal libraries (see below), that are generated with this version, must be compiled and saved again before they can be used under CoDeSys 2.2.

The default directory for CoDeSys on the hard disk (e.g. C:) is: *C:\Program files\CoDeSys V2.2*

In order that the device-specific function blocks can be used by a program the relevant libraries must be linked; explanation of this can be found in the (electronic) CoDeSys manual.

Via the menu point "CoDeSys Target Update" of the start program on the CD supplied the following device-specific files are copied into the sub-directory *Targets* (per default):

- Libraries MC8230\_\*.lib (under *Lib\_GEL8230*), see also further below
- Device file GEL8230.cfg as well as other files (\*.ico) for the generation of a process image of the inputs and outputs (under *IO\_GEL8230*)
- Example projects with which the use of certain function blocks is to be demonstrated (under *Examples\_GEL8230*); the project SPC8230\_xxxxx.pro is also found here as an example of a sentence-programmable positioning controller (xxxxx = version number)
- Information about the target system that is needed by CoDeSys2.2 for the generation of a new project.

The MotionController device libraries in detail:

- MC8230\_Basic22.lib: external<sup>1</sup> library with the basic functions and function blocks of the MotionController operating and run-time system.
- MC8230\_HMI\_Basic22.lib: as before, however, exclusively for the programming of the HMI (Human Machine Interface) for operating and observing via the keyboard and display of the GEL 8230/8231
- MC8230\_HMI\_Techno22.lib: This internal library provides solutions for diverse applications in encapsulated blocks, in

<sup>1</sup> 'External' denotes a library whose code was not created in CoDeSys, but rather in another development environment (e.g. in C++).

which functions and function blocks of the basic libraries are referred back to; with this, for example, user display menus can be developed

– MC8230\_LD100\_Basic22.lib

The function blocks summarized in this external library are identical to those for the cam plate module (MotionCard) LD 100 from LENORD+BAUER. If this library is linked, MotionCard programs or program parts can also be used in combination with the MotionController.

The description of the individual function blocks is found in a separate reference manual (*under construction at present; the function blocks are, however, annotated in detail in the library so that they can be applied for specific applications*).

## 5.2 Operating system update



Before starting an update of the operating system, de-energize all power circuits of the drives being controlled by the MotionController.



Because of the fact that several function blocks concerning the memory object structure have been modified operating systems from version 5.04 are no longer downward compatible with CoDeSys libraries MC8230\_Basic22.lib less than 5.01, i.e.

operating system GEL 823x  $\geq$  V5.04  $\Rightarrow$   
library MC8230\_Basic22  $\geq$  V5.01

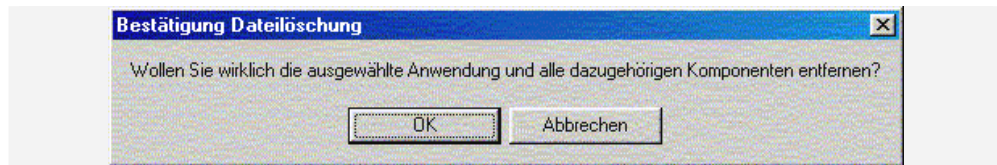
This means that older PLC programs must be recompiled and reloaded after an operating system update to a version  $\geq$  V5.04 if they are to be used any longer (the libraries must be updated first, e.g. using the corresponding start menu item on the application CD).

If the MotionController needs to be provided with a new operating system software (available e.g. on Internet: [www.lenord.de](http://www.lenord.de)), use the utility 'LingiMon' supplied (on CD) for this purpose.

- For the **installation** of LingiMon run the program LingiMonSetup.exe on the CD or simply click on the corresponding menu point in the CD start program.  
The utility will now be installed on the hard disc in a Windows-specific procedure (the directories can still be adapted to user needs).



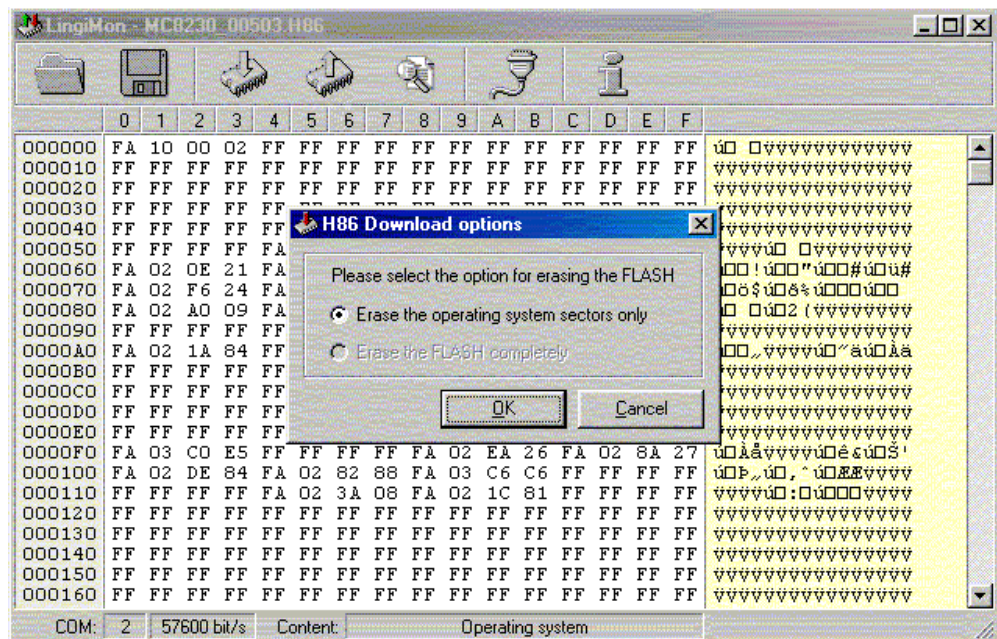
If LingiMon has already been installed, the "old" version will first be deinstalled by the installation routine:



For a new installation following that LingiMon Setup must be executed once again.

- ☑ After the installation, start the program either via the Windows Start menu or with a double click with the left hand mouse key on a binary file with the extension '.h86' or '.b86' in the Windows Explorer.
  - ☑ By means of a null modem cable make a connection between the PC (RS232 C: COM 1/2) and the MotionController (connector C1).
  - ☑ Select the COM port used in the program (set via the connector icon).
- i** Per default the transmission rate is set to 57,600 baud. If problems occur during data transfer, reduce the transmission rate in steps down to the minimum value of 9,600 baud (to be set via the connector icon).

### Starting LingiMon with selected binary file (.h86):



- i** The status bar provides information about the selected COM port (COM 1), the transmission rate (57600 baud) and the type of the file loaded (e.g. operating system).

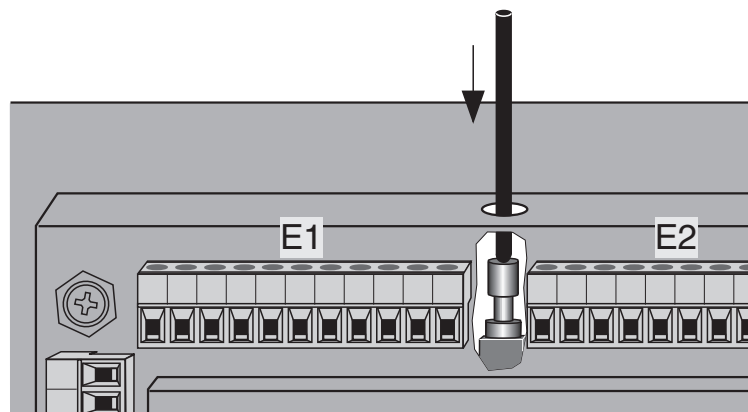
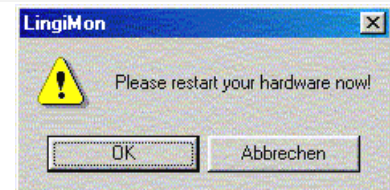
Before transmission of the data in the flash memory of the device this is first of all deleted. In general it is sufficient to delete only the operating system area – as given as a default by the program. In this way, an existing PLC program and the

system parameter settings are preserved. The other option can be enabled with the 'Download options' (click on the connector icon).

- ! During the transmission of data **do not** change to another application – the transmission window must remain active; the procedure, then, can be followed on the progression bar shown. Otherwise Windows-specific errors can occur during the transmission procedure that cause the MotionController to permanently "get lost". In such a case repeat the transmission.

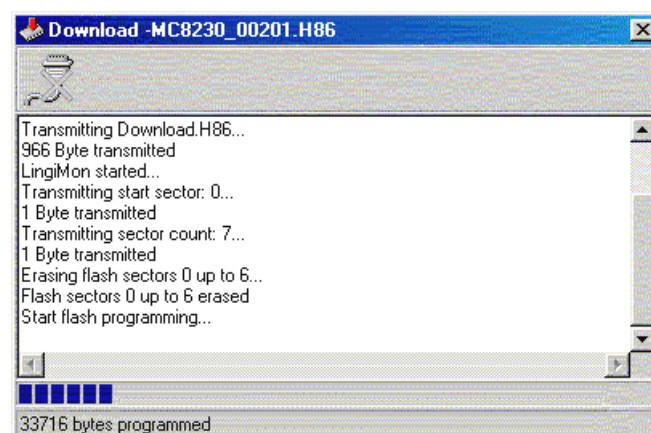
After confirming with OK the program requests the resetting of the device.

- Switch the MotionController off
- Hold the miniature key in the housing pressed down with a non-metal stick, as shown in the following illustration, (e.g. a match) and switch on the device again.



- Release the key  
The MotionController is now prepared for the download (only one or more lines can be recognised in the display).

After confirmation with OK the transmission procedure begins (approx. 250 Kbytes):



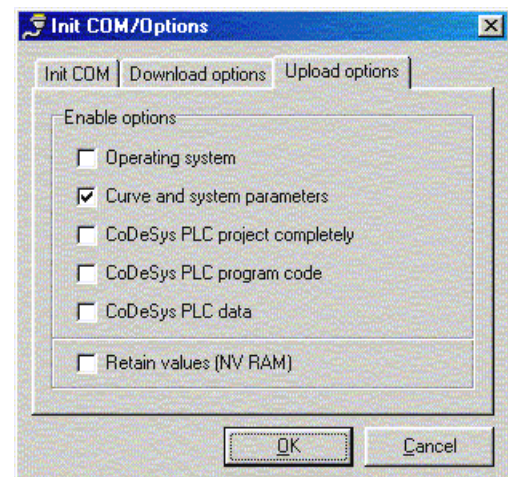
At the end of the (successful) transmission the MotionController switches to normal operation.

If this does not happen, repeat the transmission procedure, with a reduced transmission rate perhaps.

The contents of the complete flash memory or only a part of it can also be 'uploaded', i.e. read out and saved in a file with the extension '.b86' (upload). This can then be written back again at any time – as explained above (backup function). The name of the file is automatically created from the selected option, but can be altered as required.

The memory range to be uploaded is defined via the 'Upload options' (click on the connector symbol).

A comparison between the data displayed with the data in the MotionController is done by means of the magnifying glass symbol. For this a corresponding upload procedure is started.






## 6 System parameters

### 6.1 Explanations





The following is a list with short descriptions of the settable (system) parameters in the GEL 8230/8231 **configuration** menu. The headings depict the menu structure in the device.

 The basic configuration of the built-in models GEL 8235/8236 is done via the PC parameter editor "Paraminator III" supplied on CD, which also offers a comfortable possibility of programming parameters on the PC with the other models. On the other hand, parameters may also be modified in a PLC program.




<b>Entries</b>	▶ Factory defaults (6.2) .....	42
	▶ Basic settings (6.3) .....	42
	▶ Analog inputs (6.4).....	45
	▶ CAN bus (6.5) .....	45
	▶ Actual value inputs (6.6) .....	48
	▶ Nominal value outputs) (6.7) .....	53
	▶ Axis configuration (6.8).....	57
	▶ Extension module (6.9).....	69

**Programming** When changing a system parameter, a distinction must be made between two types:


a) Value parameter (e.g. positions)

After selecting the parameter, i.e. marking with the cursor bar by pressing a numerical key, the entry value of the input can be started with directly. Alternatively, the parameter can be activated by first pressing the [Enter] key and then changed. Activation can also be made directly, that is, without previous selection, with the aid of a one of the menu keys ,  or . The input field has the form .

b) Characteristic parameters (e.g. "Languages")

After selecting the parameter, i.e. marking with the cursor bar by pressing a selection key  or , the required characteristic (variant) can be set directly. In parallel to the text, the possible settings from 0 onwards are numbered consecutively. The alternative activation of a parameter is made as described in a). The input field has the form .

In the lower line of the display the numbers and the min/max values for the respective parameters are shown.

After completing the input/selection with  – i.e. for reprogramming – the parameter values are taken over by the power

failure-safe flash memory when returning to the main menu (level 02), the lowest line of the display gives information about this.

**!** Do not switch off the device during this procedure otherwise **all parameter settings will be lost!**

**i** For representation of the system parameters below the following applies:

- The factory defaults are printed in **bold**.
- On the left, beside the parameter designation, the internal system numbers are shown in square brackets, as they are also displayed during the programming of a parameter in the lowest display line. A system parameter can be addressed directly by this number when creating a user PLC program. If there are given several numbers these are valid for parameters with the same designation in the category shown in the head line (e.g. Axis 1...6); cf. the index on page 70.

## 6.2 Factory defaults

On delivery, the settable system parameters in the main menu are set to certain standard values (defaults), see also section 5.1, page 35. The MotionController can be reset to these factory defaults or to another personal configuration at any time.

### 6.2.1 Reset parameter

A selection can be made here as to whether the system parameters are set back to the

- ▶ factory defaults  
or to a
- ▶ parameter image created before.

### 6.2.2 Create parameter image

A copy of all system parameters is filed in the memory of the MotionController which is safe against power failures.

## 6.3 Basic settings

Here the application-specific configuration of the MotionController is defined.

### 6.3.1 General

[000]	Act. axes/act. in.	None   1 AX/AI   ...   <b>3 AX/AI</b>   ...   6 AX/AI
-------	--------------------	---

Here the number of the activated axes ('AX') and actual value inputs ('act. in.', 'AI') is determined, to which control/read time is allocated by the operating system.

Axes and actual value inputs not needed should be deactivated in order to reduce the control sampling time for the other axes and to allocate more resources for a PLC program. The value 3 (default) reduces the number of permitted actual value inputs to 3 and deactivates the last three axes (no. 4, 5, and 6).

This parameter is closely related to the actual value input parameter [107] ('State', see page 48), with which the individual actual value inputs themselves are activated. If more actual value inputs have been assigned the state 'Active' as were defined with [000], an error message "Invalid number of actual value inputs" is shown after a change. What to do: select a higher variant at [000] or reduce the number of (still) activated actual value inputs via [107]...[112].

**!** Before reducing the number of the axes/actual value inputs in [000] all actual value inputs not used must be deactivated ⇒ **number of actual value inputs ≥ number of activated ones**

[001]	Languages	<b>Deutsch</b>   English
-------	-----------	--------------------------

[002]	Service mode	<b>Inactive</b>   Active
-------	--------------	--------------------------

With this the password request before entering the configuration menu is permanently switched off (see also section 4.3.4.1, page 29).

[032]	Company logo	Off   <b>On</b>
-------	--------------	-----------------

Display of the copyright window after switching on the Motion-Controller (see page 31) can be suppressed: 'Off'.

### 6.3.2 PLC

[003]	PLC active	<b>Yes</b>   No
-------	------------	-----------------

With 'Yes' a high level on terminal I3.7 causes execution of the loaded PLC program.

[004]	Stop at PLC stop	No   <b>Control</b>   Regulate
-------	------------------	--------------------------------

The drive may be stopped together with the PLC program. With 'Control' there is maximum deceleration and braking without feedback control; with 'Regulate' it is done the way the axis is configured (from parameter [600], see page 57).

### 6.3.3 Serial communication (RS 232 C, RS 422/485)

[005]	Device number	<b>0</b> ...31
-------	---------------	----------------




If there are several MotionControllers on the RS-4xx bus, do not use device number 0, because a device with this number will always reply, also if e.g. device 5 is addressed.

[006]	CoDeSys interface	<b>COM 1</b>   COM 2
-------	-------------------	----------------------

Here you can define which of the two ports on connector C1 is be used for communication with the development environment CoDeSys. If, additionally, the LB2 protocol is operated this will use the other port (both protocols are of binary type and cannot be operated using the same port); see also the next parameter.

[019]	ASCII interface	<b>COM 1</b>   COM 2
-------	-----------------	----------------------

Here you can define which of the two ports on connector C1 is to be used for communication with the PC parameter editor (or another program with an ASCII protocol). If you chose a port different from that in [006] the parameter editor and CoDeSys can exchange data with the MotionController simultaneously (each using an individual PC COM port).

 Change to the configuration level for the respective serial port (return with  or 

#### 6.3.3.1 COM 1, COM 2

[007], [008]	Baud rate	9600 BD   <b>19200 BD</b>   28800 BD   38400 BD   57600 BD
--------------	-----------	--

[009], [010]	Parity check	None   Odd   <b>Even</b>
--------------	--------------	--------------------------


[011], [012]	Stop bits	<b>1 Bit</b>   2 Bits
--------------	-----------	-----------------------

## 6.4 Analog inputs

The setting options are the same for all analog inputs. The following parameter assignment is valid:

Analog input	Terminal block	Terminal	Parameter
I61	I6	1+/1-	[035], [042]
I62 *	I6	2+/2-	[036], [043]
I63 *	I6	3+/3-	[037], [044]
I54 *	I5	4+/4-	[038], [045]
I55 *	I5	5+/5-	[039], [046]
I56 *	I5	6+/6-	[040], [047]
I57 *	I5	7+/7-	[041], [048]


\* GEL 8231/8236 only (I5x are inputs for PT100 resistors)

 Copying of parameter values of an analog input into another one.

### 6.4.1 Analog input 1...7 (I61...I57)

[035]...[041]	State	<b>Inactive</b>   Active
---------------	-------	--------------------------

The analog inputs can be activated individually.


 Each analog input activated will reduce the cyclic process time of the PLC program by approx. 40 µs.

[042] ... [048]	Gate time	0.01 ... <b>1.00</b> s
-----------------	-----------	------------------------

The measured values at the analog input are determined over the gate time period set (measuring cycle ≈ 10 ms).

## 6.5 CAN bus

The operating mode of the MotionController for the two CAN connections (terminal block C2) is defined here.

 To use the CAN bus at least 2 axes must be activated (minimum control sampling time necessary = 2 ms).

### 6.5.1 CAN 1, CAN 2

[300], [301]	Type	<b>Master</b>   Slave
--------------	------	-----------------------

Definition of the main characteristics of the MotionController on the CAN bus; an additional assignment for CAN 1 can be made with the object parameter [334]...[341], see page 47.




CAN 2 ([301]) is firmly set to 'Master', alteration not possible.

[302], [303]	CAN Link	No   Yes
--------------	----------	----------

CAN Link makes it possible to set up a user CAN network with a user protocol that, therefore, is not bound by the CANopen conventions. From the hardware point of view, both CAN interfaces can be used for this, set via this parameter. A change is first effective after switching on the MotionController again – an information window draws attention to this.

The network can be used only from a PLC program by use of the corresponding CAN Link function blocks from the basic library MC8230\_Basic22.lib.

14 input/output channels are available. Only **one device per channel may transmit**, whereas the number of receivers per channel is variable. One device may transmit on several channels.

 Change to the configuration level for the respective CAN bus (return with  or .

#### 6.5.1.1 Object/PDO IN 1...OUT 4

The following Objects/PDOs) dealt with have an identical structure. For the parameter numbers shown the following applies: 1<sup>st</sup> line for CAN 1 and 2<sup>nd</sup> line for CAN 2, each sequentially for IN 1, IN 2, IN 3, IN 4, OUT 1, OUT 2, OUT 3, OUT 4.

[310]...[317] [342]...[349]	Node address	0, 1 ... 127
--------------------------------	--------------	--------------

The default setting is identical to the number of the object.

The value 0 sets the parameter its the default value.

For the object type 'Axis' (see below) the node address must not exceed the maximum of 16.

[326]...[333] [358]...[365]	Object type	<b>Inactive</b>   Digital   Analogue   Axis   Other
		–   –   –   <b>Axis</b>   –

For the input and output objects 3 and 4 of CAN 1 the variant 'Axis' cannot be selected (= 'Inactive').

The variants 'Digital' or 'Analogue' are to be set when using the CAN remote I/O module from LENORD+BAUER. 'Axis' is valid in connection with the servo amplifier LD 2000 and 'Other' for all other CAN devices (are not registered by the CAN-Bus scan; see also section 4.3.4.4, page 31).

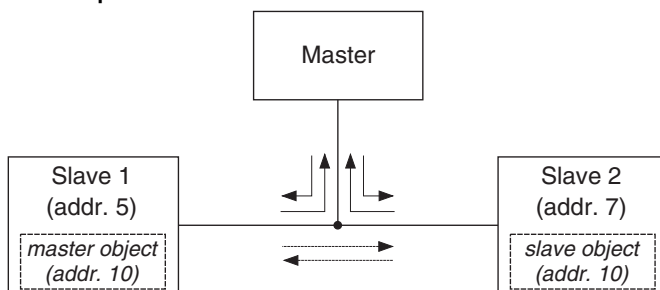
[334]...[341] [366]...[373]	Master bypass	No   Yes
--------------------------------	---------------	----------

Setting only possible for **CAN 1**

With the CANopen protocol it is not intended that slaves directly communicate with each other on the CAN bus, i.e., by bypassing the Master as "mediator". However, it can be done with the MotionController.

For this purpose the master/slave characteristic for the CAN-Object/PDOs concerned are swapped over by means of this parameter. In the standard setting ('No') all objects have the characteristic defined with the parameter [300]/[301]. By programming 'Yes' for the input and output object with the same address this will be assigned the inverse characteristic.

Example:



Programming:

It will be assumed that digital inputs/-outputs are used.

For the master, the first two input and output CAN objects are designated for 'normal' communi-

cation with the slaves (addresses 5 and 7). For the slaves each first input and output object is responsible for this, whereas the fourth input and output object for communication of the slaves among themselves is used (address 10); for slave 1 the master/slave assignment was inverted at the same time.

Output objects not used remain 'Inactive' so that they do not load the bus unnecessarily.

In the following representation **changed** settings (as against the defaults) are shown in **bold**.

Master device

Type:	[300] = <b>Master</b>
Object/PDO IN 1:	[310] = <b>5</b> , [326] = <b>Digital</b> , [334] = No
Object/PDO OUT 1:	[314] = <b>5</b> , [330] = <b>Digital</b> , [338] = No
Object/PDO IN 2:	[311] = <b>7</b> , [327] = <b>Digital</b> , [335] = No
Object/PDO OUT 2:	[315] = <b>7</b> , [331] = <b>Digital</b> , [339] = No
Object/PDO OUT 3/4:	[332]/[333] = inactive

First slave device

Type:	[300] = Slave
Object/PDO IN 1:	[310] = <b>5</b> , [326] = <b>Digital</b> , [334] = No
Object/PDO OUT 1:	[314] = <b>5</b> , [330] = <b>Digital</b> , [338] = No
Object/PDO IN 4:	[313] = <b>10</b> , [329] = <b>Digital</b> , [337] = <b>Yes</b>
Object/PDO OUT 4:	[317] = <b>10</b> , [333] = <b>Digital</b> , [341] = <b>Yes</b>
Object/PDO OUT 2/3:	[331]/[332] = inactive

☐ Second slave device

Type:	[300] = Slave
Object/PDO IN 1:	[310] = <b>7</b> , [326] = <b>Digital</b> , [334] = No
Object/PDO OUT 1:	[314] = <b>7</b> , [330] = <b>Digital</b> , [338] = No
Object/PDO IN 4:	[313] = <b>10</b> , [329] = <b>Digital</b> , [337] = No
Object/PDO OUT 4:	[317] = <b>10</b> , [333] = <b>Digital</b> , [341] = No
Object/PDO OUT 2/3:	[331]/[332] = inactive



The standard master/slave communication contains a Sync telegram and a Node Guarding telegram for the first **16** addresses. For the slave/slave communication these telegrams are dropped.

## 6.6 Actual value inputs

For the actual value inputs (max. 6, 3 of which on the terminal blocks E1...E3) certain functions can be defined, independent of the axes assignment.

The setting options are the same for almost all the inputs; differences are marked accordingly.



Copy all parameter values of an actual value input into another one.

### 6.6.1 Actual value input 1...6

[100]...[102]	Type	None   <b>Increm.</b>   SSI   CAN-Bus   Virt.1   Virt.2   Analogue
[103]...[105]		<b>None</b>   None   None   CAN-Bus   Virt.1   Virt.2   Analogue

If the 'CAN-Bus' variant is set, to be effective the CAN bus must be scanned again after leaving the programming mode (see section 4.3.4.4, page 31), an information window draws attention to this.

Configuration of the selected type is described further below.



During reprogramming, a calculated reference value offset stored safe against power failures is reset to zero (see also parameter [247]...[252], page 51).

[107]...[109]	State	Inactive   <b>Active</b>
[110]...[112]		<b>Inactive</b>   Active

Deactivating an actual value input that is not to be used




In the delivered state 3 actual value inputs and 3 axes are activated. If the number of axes is to be reduced further (parameter [000], see page 43), at least the same number of actual value inputs must be deactivated **beforehand**, namely all those that are not assigned to an active axis. Otherwise an error



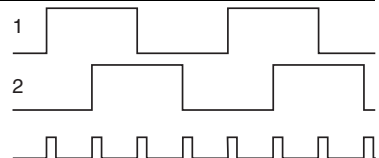
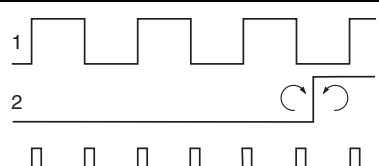
message like "Invalid number of actual value inputs" is shown after a change. The following condition must be fulfilled:

**Activated actual value inputs  $\leq$  number in [000]**


[184]...[189]	Failure diff. MAX	0 ... 99999999
	Maximum permissible actual value error within two consecutive control cycles; on exceeding the value the failure suppression described in the following is triggered; 0 deactivates this function; value in user measuring units.	
[191]...[196]	Failure tolerance	0 ... 3
	Failure suppression: number of consecutive control cycles where an actual value error occurring ("Failure diff. MAX" exceeded) is compensated For 1...3 the faulty actual value is ignored and, instead, a suitable value based on the actual velocity is interpolated.	
[013]...[018]	Display decimal pnt.	X.   X.X   X.XX   X.XXX   X.XXXX
	This setting only has an effect on values entered/displayed within this menu which are related to an actual value. The parameter with the same designation in the axes configuration is responsible for the representation in the MAIN WINDOW (AXES), (see page 59).	

 Change to the configuration level for the various input types (return with  or .

### 6.6.1.1 Incremental

[128]...[130]	Signal evaluation	Ink_V   Ink_S
	Ink_V: four edge evaluation of the encoder signals by the use of two signals phase shifted by 90° for recognition of the direction.	
	Ink_S: two edge evaluation of the encoder signal (counting pulses) by the use of the second track as direction signal	

The following possibilities of setting also apply for the other input types (identical parameter numbers).

[198]...[203]	Negation	<b>No</b>   Yes
	Reversal of counting direction; cf. "User measuring unit" on page 8	
[205]...[210]	Multiplier numerator	0 ... <b>10000</b> ... 16777215
	(see the following parameter)	
[212]...[217]	Multipl. denominator	0 ... <b>10000</b> ... 16777215
	<p>Example: Incremental encoder with 2,500 pulses per revolution, four edge evaluation (<math>\Rightarrow</math> 10,000 counting pulses); required counting range per revolution: 60.00 <math>\Rightarrow</math> multiplier = <math>[205]/[212] = 6,000/10,000 = 0.600</math>;</p> <p>for display of the required resolution of 1/100 in the MAIN WINDOW (AXES) the decimal point of axes parameters must be set to "X.XX" (see page 59).</p> <p>With appropriate choice of the counter and denominator values, multipliers can be realized with any accuracy required (<math>1/1 \equiv 10/10 \equiv 100/100</math> etc.).</p>	
	 Tip	It is advantageous to use values < 65536 for the numerator and denominator because of a decreased calculating time.
[219]...[224]	Home offset adjustm.	<b>0</b> ... $\pm 99999999$
	<p>For the value programd here the encoder count value is increased (+) or reduced (-), before it is displayed as actual position and processed further (cf. "User measuring unit" on page 8).</p> <p>Value in user measuring units</p>	
[226]...[231]	Calibration	<b>Inactive</b>   Forward   Backward   For/Back
	Definition of the jog direction for setting the reference value during the jog procedure	
[233]...[238]	Calibration edge	<b>Inactive</b>   H->L   L->H   H->LL->H
	Evaluation of the negative, positive or both signal edges	
[240]...[245]	Calibration input	None   <b>E1.N</b>   E2.N   E3.N   I1.0   ...   I1.5   CAN bus
	The default setting is dependent on the number of the actual value input: no. 1...3 $\rightarrow$ E1.N...E3.N, no. 4...6 $\rightarrow$ CAN bus.	

[247]...[252]	Calibration value	0 ... ±99999999
---------------	-------------------	-----------------

During referencing the actual value is set to this value. The resulting internal offset (see User measuring unit on page 8) is stored safe against power failures and can only be reset to zero by reselecting the encoder type (parameter [100]...[105], no 'Virtual' variant).

Value in user measuring units

### 6.6.1.2 SSI absolute

[135]...[137]	SSI code	Gray   Binary
---------------	----------	---------------

[142]...[144]	SSI- Type	Single-turn   Multi-turn
---------------	-----------	--------------------------

[149]...[151]	Single-turn bits	0 ... 12, 13
---------------	------------------	--------------

Additional setting possibilities identical to Incremental, from parameter [198]...[203] (see page 50).

### 6.6.1.3 CAN bus


[254]...[259]	Object/PDO IN	Inactive, CAN1 I1   CAN1 I2   <b>CAN2 I1</b>   ...   CAN2 I4
---------------	---------------	--

The default setting is dependent on the number of the actual value input: no. 1...4 → CAN 2, Object/PDO IN 1...4 ('CAN2 I1...4'); no. 5, 6 → CAN 1, Object/PDO IN 1, 2 ('CAN1 I1/2'). The input objects 3 and 4 of CAN 1 cannot be used as actual value inputs.

For a change to be effective the CAN bus must be scanned again after leaving the programming mode (see section 4.3.4.4, page 31), an information window draws attention to this.

[261]...[266]	CAN bus source	<b>LD 2000</b>
---------------	----------------	----------------

Here only a servo amplifier of type LD 2000 can serve as actual value source on the CAN bus at present. .

 For the servo amplifier the parameter "OPMODE" must be set to "Digital (rotational) speed".

Additional setting possibilities identical to Incremental from parameter [198] (see page 50).

### 6.6.1.4 Virtual1

Here e.g. an incremental encoder can be simulated by presetting a counter frequency.

[170]...[175]	Counter frequency	0 ... ±99999999
---------------	-------------------	-----------------

Value in increments per second

Further setting possibilities identical with Incremental from parameter [219] (see page 50).

### 6.6.1.5 Virtual2

With this feature the values of another actual value input or the data of an axis can be defined for the actual value input. In this way, e.g., a master-slave relationship between two axes may be established.

[156]...[161]	Type of source	Act. value   Frequency   Position   Velocity   Stepper motor
---------------	----------------	--


'Actual value' and 'Frequency' are the output quantities of an actual value input.

'Position', 'Velocity' and 'Stepper motor' (clock) are the output quantities of an axis (predetermined desired values of the control system, see page 59).

[163]...[168]	Number of the source	1 ... 6
---------------	----------------------	---------

Number of the actual value input or the axis

The default setting is dependent on the number of the actual value input: no. 2 → 2, no. 3 → 3 etc.

-  If the stepper motor clock shall be used as source you must map this actual value input at the axis the number of which is assigned here (parameter [606]...[611]). Otherwise counting errors may occur when changing the driving direction.

Further setting possibilities identical with Incremental from parameter [198] (see page 50).

### 6.6.1.6 Analog

The analog inputs 1...7 can also be used as actual value inputs.

[268]...[273]	Analogue input	Inactive   I/U I61   I/U I62   I/U I63   Temp I54   Temp I55   Temp I56   Temp I57
---------------	----------------	--

The default setting is dependent on the number of the actual value input: no. 2 → I/U I62, ... no. 6 → Temp I56 (variants from

"I/U I62" only in combination with the GEL 8231/8236, see also section 6.4, page 45).

[275]...[280]	Type of analog value	<b>Average</b>   Reading
---------------	----------------------	--------------------------


For generating the average value the measuring value (reading) is averaged consecutively with a sampling time of approx. 10 ms (= gate time).

Further setting possibilities are identical with Incremental, from parameter [198] (see page 50).

## 6.7 Nominal value outputs

Certain functions can be defined for the nominal value outputs, independent of the assignment of the axes. There are 6 outputs available, of which up to 3 can be used for analog or signal control purposes on terminal blocks Q1...Q3 and up to 3 for stepper motor control on terminal blocks E1...E3.

The setting options are the same for all outputs. Different default values are marked accordingly.

-  Copy all parameter values of a nominal value output into another one.




### 6.7.1 Nominal value output 1...6

[450]...[452]	Type	None   <b>Analogue</b>   CAN-Bus   Signals   Stepper m.
[453]...[455]		<b>None</b>   None   CAN-Bus   None   None

If the 'CAN-Bus' variant is set, to be effective the CAN bus must be scanned again after leaving the programming mode (see section 4.3.4.4, page 31), an information window draws attention to this.

[020]...[025]	Display decimal pnt.	<b>X.</b>   X.X   X.XX   X.XXX   X.XXXX
---------------	----------------------	---

This setting applies to position values to be entered/displayed within the nominal value outputs menu. The parameter with the same designation in the axes configuration is responsible for the representation in the MAIN WINDOW (AXES), see page 59.

-  Change to the configuration level for the various output types (return with  or .

### 6.7.1.1 Analog

[474]...[476]	Voltage range	+/- 10 V   + 10 V
---------------	---------------	-------------------

For the unipolar variant the direction information may be determined using the next parameter.


[546]...[548]	Direction on Qx	None   forw x.2   backw x.2   f/b .2/1
---------------	-----------------	--

The output assignment of direction signals is free to be made via the controlling PLC program. However, a disadvantage of this method is that the output is delayed by the duration of one program cycle.

In order to avoid such time critical processes the signals may be output in real time via the operating system (variants  $\neq$  'None'):

- Forward or backward direction on terminal Qx.2 or
  - Forward direction on Qx.2 and backward direction on Qx.1
- (x = number of the nominal value output; see also page 23)

The parameter should be set to 'None' if the signal output is to be controlled exclusively by the PLC program.

 If two or more signals are assigned the same terminal then these signals are OR-linked.

[468]...[470]	Direction	+ = Forw.   - = Forw.
---------------	-----------	-----------------------

Positive or negative voltage for forward jog (actual value counts upwards)

[480]...[482]	U max.	0.000 ... 10.000 V
---------------	--------	--------------------

Highest voltage for the drive amplifier for achieving the highest velocity of the drive (in both directions)

[504]...[506]	U min. +	0.000 ... 10.000 V
---------------	----------	--------------------

Lowest positive voltage for the drive amplifier with which it can still just operate the drive in the forward direction

[510]...[512]	U min. -	0.000 ... 10.000 V
---------------	----------	--------------------

Lowest negative voltage for the drive amplifier with which it can still just operate the drive in the backward direction

[516]...[518]	Dead range +	0 ... 99999999
---------------	--------------	----------------

As long as the actual position is in this range below the nominal position after reaching this position, the nominal value output remains switched off (output 0.000 V); value in user measuring units. The same also applies for every other (resting) position

with activated feedback position control (see 'Feedback control' under axes configuration).

[522]...[524]	Dead range -	0 ... 99999999
---------------	--------------	----------------

As long as the actual position is in this range above the nominal position after reaching this position, the assigned nominal value output remains switched off (output 0.000 V); value in user measuring units. The same also applies for every other (resting) position with activated feedback position control (see 'Feedback control' under axes configuration).

### 6.7.1.2 CAN bus

[486]...[491]	Object/PDO OUT	Inactive, CAN1 O1   CAN1 O2   <b>CAN2 O1</b>   ...   CAN2 O4
---------------	----------------	---


The default setting is dependent on the number of the nominal value output: no. 1...4 → CAN 2, Object/PDO OUT 1...4 ('CAN2 O1...4'); no. 5, 6 → CAN 1, Object/PDO OUT 1, 2 ('CAN1 O1/2').

The output objects 3 und 4 of CAN 1 cannot be used as nominal value outputs.

For a change to be effective the CAN bus must be scanned again after leaving the programming mode (see section 4.3.4.4, page 31), an information window draws attention to this.

[492]...[497]	CAN bus dest.	<b>LD 2000</b>
---------------	---------------	----------------

Here only a servo amplifier of type LD 2000 can serve as actual value destination on the CAN bus at present.

 For the servo amplifier the parameter "OPMODE" must be set to "Digital (rotational) speed".

[468]...[470]	Direction	<b>+</b> = Forw.   - = Forw.
---------------	-----------	------------------------------

Positive or negative values for forward jog (actual value counts upwards)

[498]...[503]	Inc. / rounding	0 ... <b>65536</b> ... 99999999
---------------	-----------------	---------------------------------

With this value the number of steps per motor revolution is determined if, instead of the resolver actual value, that of a separate encoder is to be used (determined with parameter [100]...[105], see page 48).

Example: Incremental encoder with 250 rated pulses per revolution and directly connected to the motor shaft ⇒ 1000 (always

4-fold edge evaluation and actual value input multiplier ignored!);  
with a single-turn SSI encoder with 12 bits used  $\Rightarrow$  4096

Because there is no feedback of the control being enabled via CAN bus the time for opening the brake (see parameter [744], page 62) should be set to approx. 30 ms if you intend to make use of the software enable of the controller. If several axes are to be started at the same time this value is to be multiplied by the number of axes.

### 6.7.1.3 Signals

For the output of binary signals for the fast/slow jog control only the first three nominal value outputs are available. Further information is given in the axes configuration: section 6.8.1.2 "Signals", page 65.

[528]...[530]	Signal output to Qx	<b>None</b>   Slow x.3   S/F .3/4
---------------	---------------------	-----------------------------------


The output assignment of the binary slow/fast signals is free to be made via the controlling PLC program. However, a disadvantage of this method is that the output is delayed by the duration of one program cycle.

In order to avoid such time critical processes the signals may be output in real time via the operating system (variants  $\neq$  'None'):

- Slow jog on terminal Qx.3 or
- Slow jog on Qx.3 and fast jog on Qx.4

(x = number of the nominal value output; see also page 23)

The parameter should be set to 'None' if the signal output is to be controlled exclusively by the PLC program.

 If two or more signals are assigned the same terminal then these signals are OR-linked.

[546]...[548]	Direction on Qx	<b>None</b>   forw x.2   backw x.2   f/b .2/1
---------------	-----------------	---

The output assignment of direction signals is free to be made via the controlling PLC program. However, a disadvantage of this method is that the output is delayed by the duration of one program cycle.


In order to avoid such time critical processes the signals may be output in real time via the operating system (variants  $\neq$  'None'):

- Forward or backward direction on terminal Qx.2 or
- Forward direction on Qx.2 and backward direction on Qx.1

(x = number of the nominal value output; see also page 23)

The parameter should be set to 'None' if the signal output is to be controlled exclusively by the PLC program.



 If two or more signals are assigned the same terminal then these signals are OR-linked.

[534]...[536]	Reverse direction	<b>No</b>   Yes
---------------	-------------------	-----------------

The direction signals on the terminals can be swapped over

#### 6.7.1.4 Stepper motor

The clock control signal for the stepper motor is output on terminals C and /C of one of the encoder inputs E1 to E3 and the necessary direction signal on a specific digital output of terminal block Q1...3.

[540]...[542]	max. stepping frequ.	<b>0</b> ... 100000 Hz
---------------	----------------------	------------------------

Number of steps per second which are to be output for achieving the maximum speed ([642]...[647], see page 60)

Further parameters as for "Signals" (see previous section, [546]...[548] and [534]...[536])

### 6.8 Axis configuration

The parameters described in the following are significant, above all, for the positioning of a drive. They are accessed via certain function blocks in a PLC program. But a few parameters are also needed for manual operation of the drive via the MAIN WINDOW (AXES) as a commissioning aid.

Certain functions can be defined for the maximum of 6 axes, independent of one another.

The setting options are the same for all axes. Different default values are marked accordingly.

 Copy all parameter values of an axis into another one.

#### 6.8.1 Axis 1...6

[600]...[602]	Controlling mode	None   <b>Control.</b>   Signals   Ramps
[603]...[605]		<b>None</b>   Control.   Signals   Ramps

The default setting applies only to the first three axes.


The variant 'None' decreases the computing time by the amount needed for the controlling procedure. Thus, the processor has more time available for the processing of a PLC program. For example, this is practical if only one actual value input is to be

used without having to move the drive as well. Every active axis increases the control sampling time by 1 ms.

With 'Regulate' the internal feedback control system, consisting of speed pre-control and a desired/actual value comparison, is activated. But the latter has an effect only if the control factor KSP (parameter [648]...[653], see page 60) is set to a value  $\neq 0$ . This is important for when position control is to be active in the nominal or stop position (parameters [732]...[735] and [738]...[743], see page 62).

With 'Signals' a two-stage drive control is made via the signals direction, slow jog and fast jog (see parameter [528], page 57). This is generally called fast/slow jog control .

With "Ramps" the braking process is influenced by the remaining distance to the target position. Above all, hydraulic drives and stepper motors may be adjusted more effectively by using this method. This feature has been designed for controlling drives via CAN bus, analog voltage or stepper motor by means of a PLC program.

The specific parameters are set in the next sub-menu (key ) , see the following sections.

[606]...[611]

Actual value input	None   <b>Input 1</b>   ...   Input 6   Stepper motor   Virtual
--------------------	---

Assignment of one of 6 actual value inputs, a stepper motor application or a virtual function

If no input is assigned, the axis is without function but it is not inactive, i.e. it claims a part of the program cycle time. To prevent this the axis must be switched off (see parameter [000], page 43).

The 'Virtual' option creates a simulated axis which will follow the path and velocity profile given by the controller, without tracking error (actual position  $\equiv$  position value calculated by the controller). Thus, the positioning behaviour of a future real axis can very easily be tested by a PLC program in the laboratory. At the commissioning stage of the axis in the system a real actual value input must simply be assigned to it.

Using the 'Stepper motor' option the output clock pulses are counted and the number is supplied to the axis as actual position value.

The default setting is dependent on the number of the axis:  
no. 2  $\rightarrow$  Input 2, no. 3  $\rightarrow$  Input 3 etc.

[612]...[617]	Nominal value output	None   <b>Output 1</b>   ...   Output 6
---------------	----------------------	---

Assignment of one of the 6 nominal value outputs

**i** Even if no output is assigned, the nominal values for position and velocity generated by the control of this axis can nevertheless be made accessible for another axis by defining their accompanying actual value input as 'Virtual2' (see actual value inputs, page 52).

The default setting is dependent on the number of the axis:  
no. 2 → output 2, no. 3 → output 3 etc.

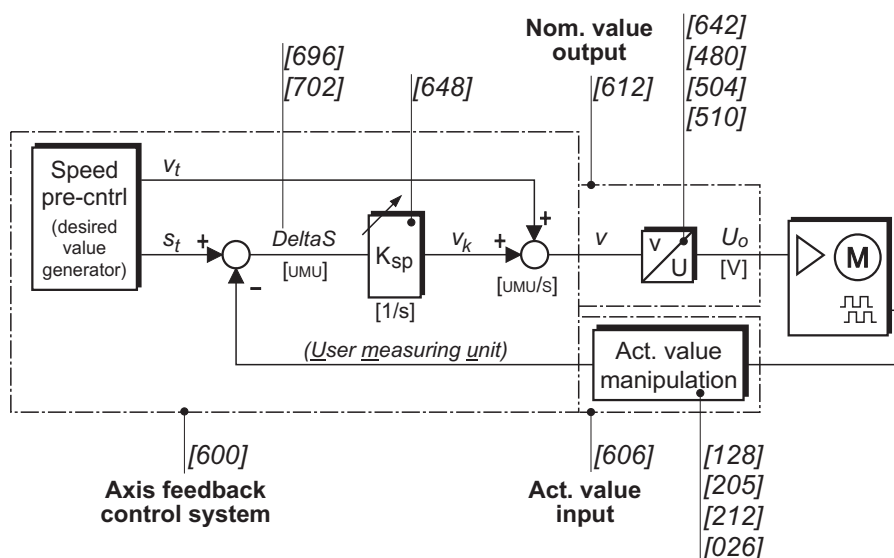
[026]...[031]	Display decimal pnt.	<b>X.</b>   X.X   X.XX   X.XXX   X.XXXX
---------------	----------------------	---

This setting applies to position values to be entered/displayed within the axes menu and in the MAIN WINDOW (AXES); further decimal point settings can be made in the menus "Actual value inputs" (page 49) and "Nominal value outputs" (page 53).

**F3** Change to the configuration level for the control modes (return with **M1** or **ESC**).

### 6.8.1.1 Feedback control ('Controlled')

The axis feedback control of the operating system works according to the following principle (example for an incremental actual value input and an analog nominal value output):




**Note:** Only a few selected parameter numbers are given in the diagram, each related to the first item of a category. The (correct) behaviour of the main block "Speed pre-control", that generates the desired values for position and velocity per control cycle, is influenced by almost all of the following parameters.

[642]...[647]	V max.	<b>0</b> ... 3276800 (2730666)
<p>Highest possible velocity of the drive</p> <p>The value in brackets applies for the case where all 6 axes are activated (see also the footnote in the application example on page 74).</p> <p>The value determined here is closely related to the maximum voltage with analogue outputs ([480]...[482], see page 54) and the maximum output frequency with stepper motor control ([540]...[542], see page 57).</p> <p>Value in user measuring units per second</p>		
[684]...[689]	V Work	<b>0</b> ... 3276800 (2730666; see above)
<p>Working velocity for the positioning of the drive</p> <p>This information is important when using certain movement function blocks in a PLC program where no velocity can be preset.</p> <p>Value in user measuring units per second, &lt; "V max."</p>		
[648]...[653]	KSP	<b>0.0</b> ... 999.9 s <sup>-1</sup>
<p>Feedback control factor responsible for the dynamics of the drive control (determine suitable values by test runs)</p> <p>0.0 = no feedback control (speed pre-control only); because of this an activated position control in the stop or target position is ineffective (see further below in this section).</p>		
[654]...[659]	t max. acceleration	<b>0.000</b> ... 9.999 s
<p>Duration of the highest possible acceleration of the drive from standstill up to "V max." (see above), absolute limit</p> <p>With 0.000 no positioning procedure will be carried out.</p>		
[660]...[665]	t max. deceleration	<b>0.000</b> ... 9.999 s
<p>Duration of the highest possible deceleration (braking) of the drive from "V max." (see above) to standstill – absolute limit</p> <p>With 0.000 the value of 't max. acceleration' is used.</p>		
[666]...[671]	t acceleration	<b>0.000</b> ... 9.999 s
<p>Duration of the required working acceleration of the drive; value <math>\geq</math> 't max. acceleration' (see above)</p> <p>With 0.000 the value of 't max. acceleration' is used.</p>		

[678]...[683]	t deceleration	0.000 ... 9.999 s
	Duration of the required working deceleration of the drive; value $\geq$ 't max. deceleration' (see above) With 0.000 the value of 't max. deceleration' is used.	
[690]...[695]	T jerking	0.000 ... 9.999 s
	Time for achieving the maximum acceleration of the drive The jerk determines the positioning characteristic of the drive; a longer jerk time ( $\Rightarrow$ smaller jerk) results in a more moderate start up and braking process.	
[696]...[701]	Max. contour. error +	0 ... 99999999
	As soon as the actual value <u>falls short</u> of the nominal value by this amount ( $\Delta S \geq [696]$ ) the drive is stopped. With 0 tracking error monitoring is disabled. Value in user measuring units	
[702]...[707]	Max. contour. error -	0 ... 99999999
	As soon as the actual value <u>advances</u> the nominal value by this amount ( $\Delta S \leq -[702]$ ) the drive is stopped. With 0 tracking error monitoring is disabled. Value in user measuring units	
[708]...[713]	Tolerance +	0 ... 99999999
	As long as the actual position remains in this range <u>below</u> the nominal position, after it has once been reached, the state 'actual=nominal' is preserved (may be polled in a PLC program); see also parameter [750] further below. Value in user measuring units	
[714]...[719]	Tolerance -	0 ... 99999999
	As long as the actual position remains in this range <u>above</u> the nominal position, after it has once been reached, the state 'actual=nominal' is preserved (may be polled in a PLC program); see also parameter [750] further below. Value in user measuring units	
[720]...[725]	Tool correction	-99999999 ... 0 ... 99999999
	Correction value for compensating of cutting waste or tool wear and tear	

[732]...[737]	Control in stop	<b>No</b>   Yes
	Activation of feedback position control for the stopped state of the drive; immediately effective, however only for KSP > 0 (parameter [648]...[653], page 60)	
[738]...[743]	Control in nom. pos.	No   <b>Yes</b>
	Feedback position control for the 'actual=nominal' state of the drive is activated per default; only effective for KSP > 0 (parameter [648]...[653], page 60)	
[744]...[749]	t brake open	<b>0.00</b> ... 9.99 s
	Time delay for the positioning control to become effective after the start signal (control start delay)	
[750]...[755]	t brake close	<b>0.00</b> ... 9.99 s
	Time delay for disabling the positioning control after entering the 'actual=nominal' tolerance window (control stop delay)	
[756]...[761]	V jog slow forward	<b>0</b> ... 3276800 (2730666)
	Manual operation of the drive at low speed in the forward direction The value in brackets applies for the case where all 6 axes are activated (see also the footnote in the application example on page 74). Value in user measuring units per second	
[762]...[767]	V jog fast forward	<b>0</b> ... 3276800 (2730666; see above)
	Manual operation of the drive at high speed in the forward direction Value in user measuring units per second	
[768]...[773]	V jog slow backward	<b>0</b> ... 3276800 (2730666; see above)
	Manual operation of the drive at low speed in the backward direction Value in user measuring units per second	
[774]...[779]	V jog fast backward	<b>0</b> ... 3276800 (2730666; see above)
	Manual operation of the drive at high speed in the backward direction Value in user measuring units per second	

[780]...[785]	Jogging forward	[>][>>]   [<<][<]
	The jog polarity assignment of the manual operation keys can be swapped over.	
[786]...[791]	Control jog	No   Yes
	Manual operation of the drive can be feedback-controlled or simply velocity controlled (fixed voltage curve)	
[792]...[797]	Auto calibration V1	0 ... 3276800 (2730666; see above)
	Velocity of the drive for the automatic calibration <u>up to</u> the turning point (initiator) Value in user measuring units per second	
[798]...[803]	Auto calibration V2	0 ... 3276800 (2730666; see above)
	Velocity of the drive for the automatic calibration <u>after</u> the turning point (initiator) in the direction of the reference point. Value in user measuring units per second	
[804]...[809]	Auto calibration	None   Forward   Backward
	Forward: calibration value is set in the forward direction with jog, i.e. the calibration starts in the backward direction (towards the turning point initiator).	
[810]...[815]	Control calibration	No   Yes
	The automatic calibration can be feedback-controlled or simply velocity controlled (fixed voltage curve).	
[924]...[929]	SW limit switch +	Inaktiv   Aktiv
	Activation of the upper software limit switch.	
	 Software and Hardware limit switches: Limit switch monitoring is processed independently by the operating system. But you may change all system parameter settings <b>temporarily</b> in a PLC program (→ MC8230_Basic22 library, under Axis control/Status). When overriding (+) or dropping below (-) a limit switch position (value or signal) the operating system stops the corresponding axis; PLC drive functions will then return the error code 4 (SW+) / 8 (SW-) or 40h (HW+) / 80h (HW-) when calling. In this case only those motion commands will be executed which move the drive back into the admissible range.	

Software limit switch monitoring does **not** affect the CoDeSys function block "Auto\_Calibration".

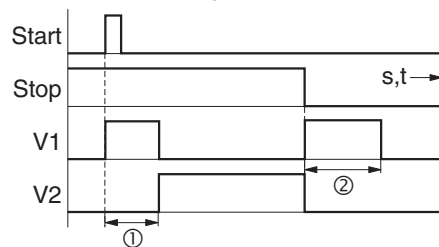
[930]...[935]	SW limit position +	-99999999 ... <b>0</b> ... 99999999
	Value of the upper software limit switch position.	
[936]...[941]	SW limit switch -	<b>Inactive</b>   active
	Activation of the lower software limit switch.	
[942]...[947]	SW limit position -	-99999999 ... <b>0</b> ... 99999999
	Value of the lower software limit switch position.	
[948]...[953]	HW limit switch +	<b>Inactive</b>   Low active   High active
	Activation of the upper hardware limit switch by determination of the active signal edge.	
[954]...[959]	HW limit switch in +	<b>IX 1.0</b>   IX 1.2   ...   IX 4.7
	Determination of the digital input for the upper hardware limit switch: I1.0...I4.7 (I4.0...I4.7 with GEL 8231/8236 only).	
[960]...[965]	HW limit switch -	<b>Inaktiv</b>   Low-aktiv   High-aktiv
	Activation of the lower hardware limit switch by determination of the active signal edge.	
[966]...[971]	HW limit switch in -	<b>IX 1.0</b>   IX 1.2   ...   IX 4.7
	Determination of the digital input for the lower hardware limit switch: I1.0...I4.7 (I4.0...I4.7 with GEL 8231/8236 only).	



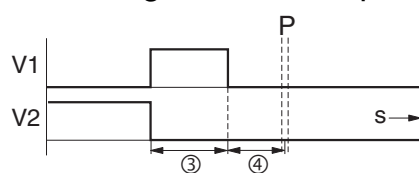
### 6.8.1.2 Signals

The fast/slow jog control results in the following signal sequence (all information related to axis 1, controlled by 3 signals):

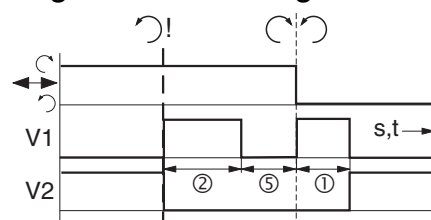
#### a) Start and stop



#### b) Reaching the nominal position



#### c) Jog direction change



#### Legend:

V1 = slow jog

V2 = fast jog

P = nominal position

↔ = direction

① = 'ds(dt) acceleration':  
[816] or [852]

② = 'dt shut down: [846]

③ = 'ds shut down +/-':  
[822]/[834]

④ = 'ds run out +/-':  
[828]/[840]

⑤ = 'dt run out': [870]

A jog direction change is made, e.g. with automatic calibration, at the turning point; but this is only done in slow jog (V1).

[816]...[821]	ds acceleration	0 ... 99999999
---------------	-----------------	----------------

Distance that the drive needs to accelerate to the first velocity stage (slow jog)

This information has priority over an alternative time preset (see next parameter).

Value in user measuring units

[852]...[857]	dt acceleration	0.000 ... 9.999 s
---------------	-----------------	-------------------

Time that the drive needs to accelerate to the first velocity stage (slow jog)

This information is ignored if a value > 0 is programmed in the previous parameter.

[822]...[827]	ds shut down +	0 ... 99999999
---------------	----------------	----------------

Distance that the drive needs with forward jog to brake from fast to slow jog.

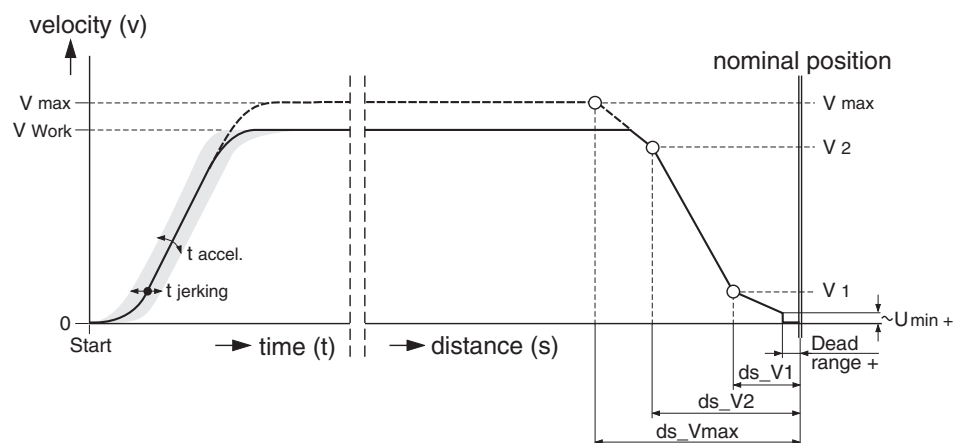
Value in user measuring units

[828]...[833]	ds run out +	0 ... 99999999
	Distance that the drive needs with forward jog to get from slow jog to standstill. Value in user measuring units	
[834]...[839]	ds shut down -	0 ... 99999999
	As [822]...[827], but for backward jog	
[840]...[845]	ds run out -	0 ... 99999999
	As [828]...[833], but for backward jog	
[846]...[851]	dt shut down	0.000 ... 9.999 s
	Change of direction: time that the drive needs to reduce its speed from fast to slow jog	
[870]...[875]	dt run out	0.000 ... 9.999 s
	Change of direction: time that the drive needs to reduce its speed from the slow jog so that a signal can be given for the reverse direction	
[708]...[713]	Tolerance +	0 ... 99999999
	As long as the actual position remains in this range <u>below</u> the nominal position, after it has once been reached, the state 'actual=nominal' is preserved (may be polled in a PLC program). Value in user measuring units	
[714]...[719]	Tolerance -	0 ... 99999999
	As long as the actual position remains in this range <u>above</u> the nominal position, after it has once been reached, the state 'actual=nominal' is preserved (may be polled in a PLC program). Value in user measuring units	
[744]...[749]	t brake open	0.00 ... 9.99 s
	Time delay for the output of the slow jog signal after the start signal (control start delay)	
[780]...[785]	Jogging forward	[>][>>]   [<<][<]
	The jog polarity assignment of the manual operation keys can be swapped over.	

[864]...[869]	Fast pos.	<b>No</b>   Yes
<p>The fast jog signal may be activated for the positioning of the drive.</p> <p>This information is important when using certain movement function blocks in a PLC program where no velocity can be preset.</p>		
[858]...[863]	Fast jog	<b>No</b>   Yes
<p>The fast jog signal may be activated for the manual operation of the drive.</p> <p>This information is important when using certain movement function blocks in a PLC program where no velocity can be preset.</p>		
[804]...[809]	Auto calibration	<b>None</b>   Forward   Backward
<p>Forward: calibration value is set in the forward direction with jog, i.e. the calibration starts in the backward direction (towards the turning point initiator). For the auto calibration no fast jog signal is output.</p>		

### 6.8.1.3 Ramps

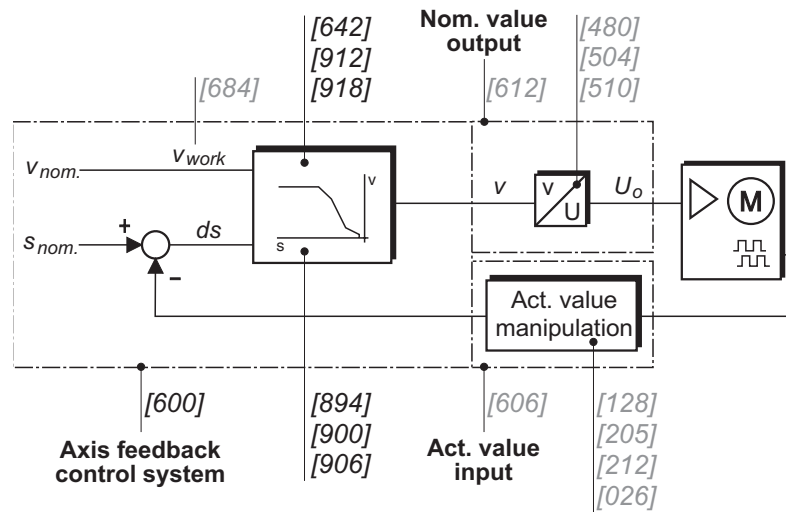
The ramp control functions as follows:



While the accelerating process runs time-controlled – as it is with feedback control via speed pre-control (with *t acceleration* and *t jerking*) – the braking process is carried out with a three-stage speed reduction depending on the residual distance from the target position.

The (theoretical) shape of the curve is related to *V max*. If you operate at a smaller speed the ramp entry point is reached later fulfilling the condition: ramp velocity  $v \leq V \text{ working}$  ([684]).

The complete positioning procedure is carried out without tracking error monitoring ( $KSP = 0$ ) according to the following principle (shown for axis 1 with analog nominal value output, valid after leaving the acceleration stage):



The subgroup "Ramps" contains all the parameters of the Feedback control subgroup (see section 6.8.1.1), and additionally:

[918]...[923]	V2 – Ramp	0 ... 3276800 (2730666)
	Velocity of the medium ramp segment; in user measuring units per second	
	The value in brackets applies for the case where all 6 axes are activated (see also the footnote in the application example on page 74).	
[912]...[917]	V1 – Ramp	0 ... 3276800 (2730666; see above)
	Velocity of the last ramp segment in user measuring units per second	
[894]...[899]	ds Vmax – Ramp	0 ... 99999999
	Distance of the ramp point 'V max.' from the target position in user measuring units	
[906]...[911]	ds V2 – Ramp	0 ... 99999999
	Distance of the ramp point 'V2' from the target position in user measuring units	
[900]...[905]	ds V1 – Ramp	0 ... 99999999
	Distance of the ramp point 'V1' from the target position in user measuring units	

## 6.9 Extension module

The following parameters listed make up the basic configuration for the Field bus modules only. Operation is carried out via a PLC program by means of appropriate function blocks that are made available in the basic library.

### 6.9.1 DeviceNet

[056]	Data format	<b>Intel</b>   Motorola
-------	-------------	-------------------------

Intel: LSB at the lower address

Motorola: LSB at the higher address

[060]	Byte count IN	<b>0</b> ... 512
-------	---------------	------------------

The size of the input data is settable from 1 to 512 bytes (0 = inactive).

[061]	Byte count OUT	<b>0</b> ... 512
-------	----------------	------------------

The size of the output data is settable from 1 to 512 bytes (0 = inactive).

### 6.9.2 InterBus-S

[056]	Data format	<b>Intel</b>   Motorola
-------	-------------	-------------------------

Intel: LSB at the lower address

Motorola: LSB at the higher address

[070]	Byte count IN/OUT	<b>0</b> ... 20
-------	-------------------	-----------------

The size of the input and output data is settable from 1 to 20 bytes (0 = inactive).

### 6.9.3 PROFIBUS-DP

[056]	Data format	<b>Intel</b>   Motorola
-------	-------------	-------------------------

Intel: LSB at the lower address

Motorola: LSB at the higher address

[080]	Byte count IN	<b>1 Byte</b>   2 Byte   3 Byte   8 Byte   16 Byte   32 Byte   64 Byte   128 Byte
-------	---------------	--

For the sizes of the input data various values are selectable between 1 and 128 bytes.

[081]	Byte count OUT	<b>1 Byte</b>   2 Byte   3 Byte   8 Byte   16 Byte   32 Byte   64 Byte   128 Byte
-------	----------------	--

For the sizes of the output data various values are selectable between 1 and 128 bytes.

#### 6.9.4 Ethernet

Identical with DeviceNet (→ section 6.9.1).

### 6.10 Numerical index

[000]	Activated axes (general)	43
[001]	Languages (general)	43
[002]	Service mode (general)	43
[003]	PLC active (PLC)	43
[004]	Stop at PLC stop (PLC)	44
[005]	Device number (ser. comm.)	44
[006]	CoDeSys interface (ser. comm.)	44
[007], [008]	Baud rate (COM 1/2)	44
[009], [010]	Parity check (COM 1/2)	44
[011], [012]	Stop bits (COM 1/2)	44
[013]...[018]	Decimal point (actual value input 1...6)	49
[019]	ASCII interface (ser. comm.)	44
[020]...[025]	Decimal point (nominal value output 1...6)	53
[026]...[031]	Decimal point (axis 1...6)	59
[032]	Company logo (general)	43
[035]...[041]	Status (analog inputs 1...7)	45
[042]...[048]	Gate time (analog inputs 1...7)	45
[056]	Data format (Field bus)	69
[060]	Byte count IN (DeviceNet)	69
[061]	Byte count OUT (DeviceNet)	69
[070]	Byte count IN/OUT (InterBus-S)	69
[080]	Byte count IN (PROFIBUS-DP)	69
[081]	Byte count OUT (PROFIBUS-DP)	70
[100]...[105]	Type (actual value input 1...6)	48
[107]...[112]	State (actual value input 1...6)	48
[128]...[130]	Signal evaluation (actual value input 1...3)	49
[135]...[137]	SSI code (actual value input 1...3)	51
[142]...[144]	SSI type (actual value input 1...3)	51
[149]...[151]	Single-turn bits (actual value input 1...3)	51
[156]...[161]	Type of source (actual value input 1...6)	52
[163]...[168]	Number of the source (actual value input 1...6)	52
[170]...[175]	Counter frequency (actual value input 1...6)	52
[184]...[189]	Failure diff. MAX (actual value input 1...6)	49
[191]...[196]	Failure tolerance (actual value input 1...6)	49
[198]...[203]	Negation (actual value input 1...6)	50
[205]...[210]	Multiplier numerator (actual value input 1...6)	50
[212]...[217]	Multipl. denominator (actual value input 1...6)	50

[219]...[224]	Home offset adjustm. (actual value input 1...6)	50
[226]...[231]	Calibration (actual value input 1...6)	50
[233]...[238]	Calibration edge (actual value input 1...6)	50
[240]...[245]	Calibration input (actual value input 1...6)	50
[247]...[252]	Calibration value (actual value input 1...6)	51
[254]...[259]	Object/PDO IN (actual value input 1...6)	51
[261]...[266]	CAN bus source (actual value input 1...6)	51
[268]...[273]	Analog input (actual value input 1...6)	52
[275]...[280]	Type of the analogue value (actual value input 1...6)	53
[300], [301]	Type (CAN 1/2)	45
[302], [303]	CAN Link (CAN 1/2)	46
[310]...[317]	Node address (Object/PDO IN/OUT 1...4, CAN 1)	46
[326]...[333]	Object type (Object/PDO IN/OUT 1...4, CAN 1)	46
[334]...[341]	Master bypass (Object/PDO IN/OUT 1...4, CAN 1)	47
[342]...[349]	Node address (Object/PDO IN/OUT 1...4, CAN 2)	46
[358]...[365]	Object type (Object/PDO IN/OUT 1...4, CAN 2)	46
[366]...[373]	Master bypass (Object/PDO IN/OUT 1...4, CAN 2)	47
[450]...[455]	Type (nominal value output 1...6)	53
[468]...[470]	Direction (nominal value output 1...3)	54, 55
[474]...[476]	Voltage range (nominal value output 1...3)	54
[480]...[482]	Maximum voltage (nominal value output 1...3)	54
[486]...[491]	Object/PDO OUT (nominal value output 1...6)	55
[492]...[497]	CAN bus dest. (nominal value output 1...6)	55
[498]...[503]	Increm./revolution (nominal value output 1...6)	55
[504]...[506]	Minimum positive voltage + (nominal value output 1...3)	54
[510]...[512]	Minimum negative voltage - (nominal value output 1...3)	54
[516]...[518]	Dead range + (nominal value output 1...3)	54
[522]...[524]	Dead range - (nominal value output 1...3)	55
[528]...[530]	Signals to Qx.2/3/4 (nominal value output 1...3)	56
[534]...[536]	Reverse direction (nominal value output 1...3)	57
[540]...[542]	max. stepping frequency (nominal value output 1...3)	57
[546]...[548]	Direction on Qx (nominal value output 1...3)	54, 56, 57
[600]...[605]	Controlling mode (axis 1...6)	57
[606]...[611]	Actual value (axis 1...6)	58
[612]...[617]	Nominal value output (axis 1...6)	59
[642]...[647]	V max. (axis 1...6)	60
[648]...[653]	KSP (axis 1...6)	60
[654]...[659]	t max. acceleration (axis 1...6)	60
[660]...[665]	t max. deceleration (axis 1...6)	60
[666]...[671]	t acceleration (axis 1...6)	60
[678]...[683]	t deceleration (axis 1...6)	61
[684]...[689]	V working (axis 1...6)	60
[690]...[695]	t jerking (axis 1...6)	61
[696]...[701]	Max. contour. error + (axis 1...6)	61
[702]...[707]	Max. contour. err. - (axis 1...6)	61
[708]...[713]	Tolerance + (axis 1...6)	61, 66
[714]...[719]	Tolerance - (axis 1...6)	61, 66
[720]...[725]	Tool correction (axis 1...6)	61
[732]...[737]	Control in stop (axis 1...6)	62
[738]...[743]	Control in nom. pos. (axis 1...6)	62
[744]...[749]	t brake open (axis 1...6)	62, 66
[750]...[755]	t brake close (axis 1...6)	62
[756]...[761]	V jog slow forward (axis 1...6)	62
[762]...[767]	V jog fast forward (axis 1...6)	62
[768]...[773]	V jog slow backward (axis 1...6)	62
[774]...[779]	V jog fast backward (axis 1...6)	62

[780]...[785]	Jogging forward (axis 1...6)	63, 66
[786]...[791]	Control jog (axis 1...6)	63
[792]...[797]	Auto calibration V1 (axis 1...6)	63
[798]...[803]	Auto calibration V2 (axis 1...6)	63
[804]...[809]	Auto calibration (axis 1...6)	63, 67
[810]...[815]	Control calibration (axis 1...6)	63
[816]...[821]	ds acceleration (axis 1...6)	65
[822]...[827]	ds shut down + (axis 1...6)	65
[828]...[833]	ds run out + (axis 1...6)	66
[834]...[839]	ds shut down - (axis 1...6)	66
[840]...[845]	ds run out - (axis 1...6)	66
[846]...[851]	dt shut down (axis 1...6)	66
[852]...[857]	dt acceleration (axis 1...6)	65
[858]...[863]	Fast jog (axis 1...6)	67
[870]...[875]	dt run out (axis 1...6)	66
[894]...[899]	ds Vmax - Ramp (Axis 1...6)	68
[900]...[905]	ds V1 - Ramp (Axis 1...6)	68
[906]...[911]	ds V2 - Ramp (Axis 1...6)	68
[912]...[917]	V1 - Ramp (Axis 1...6)	68
[918]...[923]	V2 - Ramp (Axis 1...6)	68
[924]...[929]	SW limit switch + (axes 1...6)	63
[930]...[935]	SW limit position + (axes 1...6)	64
[936]...[941]	SW limit switch - (axes 1...6)	64
[942]...[947]	SW limit position - (axes 1...6)	64
[948]...[953]	HW limit switch + (axes 1...6)	64
[954]...[959]	HW limit switch in + (axes 1...6)	64
[960]...[965]	HW limit switch - (axes 1...6)	64
[966]...[971]	HW limit switch in - (axes 1...6)	64

## 6.11 Application example

Subject Control of a single feedback-controlled axis via CAN bus

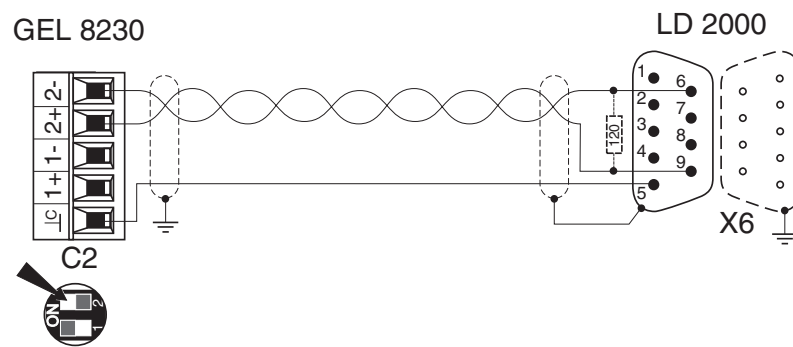
**!** For the CAN bus communication a control sampling time (cycle time) of at least 2 ms is necessary. ⇒ At least 2 axes need to be activated, also when only one axis is actually used.

Definition

- Master = MotionController GEL 8230
- Slave = servo amplifier LD 2000
- CAN bus = CAN 2
- Axis = no. 1, with default assignment of actual value input no. 1 and nominal value output no. 1



## CAN connection



The MotionController should be the first device on the bus. Therefore, the internal terminating resistor for CAN 2 must be switched in: DIP switch 2 to ON.

If the servo amplifier is the last device on the bus, an external resistor of approx. 120  $\Omega$  must be soldered in the connector (see diagram).

## Configuration

The settings listed in the following are only a selection in relation to the CAN connection of the two devices looked at.

The complete parameterization of the servo amplifier and the MotionController e.g. related to the axes control parameters is not dealt with here. Information about this is given in the commissioning manual of the LD 2000, and the description of the system parameters is given in this chapter.

System parameters that maintain their default settings for this example are not mentioned further.

**!** After setting (changing) the CAN parameters in the MotionController, a scan procedure must be carried out so that the CAN controller can set up a connection to the servo amplifier that is prepared to function.

**To do this please pay attention to the instructions in section 4.3.4.4 (→ page 32).**

LD 2000 Setting via Drive.exe:

- a) Under basic adjustments:
  - Address = 1
  - Baud rate = 500 kBaud
  - (AutoEnable = off; only if the drive in the uncontrolled state is not to go into feedback position control)
- a) OPMODE = 0: Digital (rotational) speed

GEL 8230 Settings in the MotionController: menu *Configuration*

- a) Deactivate actual value inputs and axes that are not used (thus reducing the cycle time):
  - (1) [108]...[112] = Inactive (*Actual value inputs/... 2...6/State*)
  - (2) [000] = 2 (*Basic settings/General/Act. axes...*)
- b) Switch off feedback control for the axis 2 not used (because of this more computing time is available to a loaded PLC program):  
[601] = None (*Axes configuration/Axis 2/Controlling mode*)
- c) Actual value input 1  
[100] = CAN bus (*Actual value inputs/Actual value input 1/Type*)  
(Info: IN1 is already preset as Object/PDO of CAN 2, to which, by default, address 1 is assigned)
- d) Nominal value output 1  
[450] = CAN bus (*Nominal value outputs/Nominal value output 1/Type*)  
(Info: OUT 1 is already preset as Object/PDO of CAN 2, to which, by default, address 1 is assigned)

After these preparations the drive can already be operated manually via the MotionController if, at least, the following control parameters have been defined (*Configuration/Axes configuration/Axis 1/[NEXT]/ [NEXT]*):

- 'V max' [642]  
Because the nominal value output works with a resolution of 65,536 increments per revolution, for a maximum velocity of e.g.  $3000 \text{ min}^{-1}$  ( $50 \text{ s}^{-1}$ ) the value  $50 * 65,536 = 3,276,800^*)$  would need to be programmed.
- 't max. acceleration' [654] (e.g. 0.5 s)
- Velocity values for manual operation: [756], [762], [768], [774]

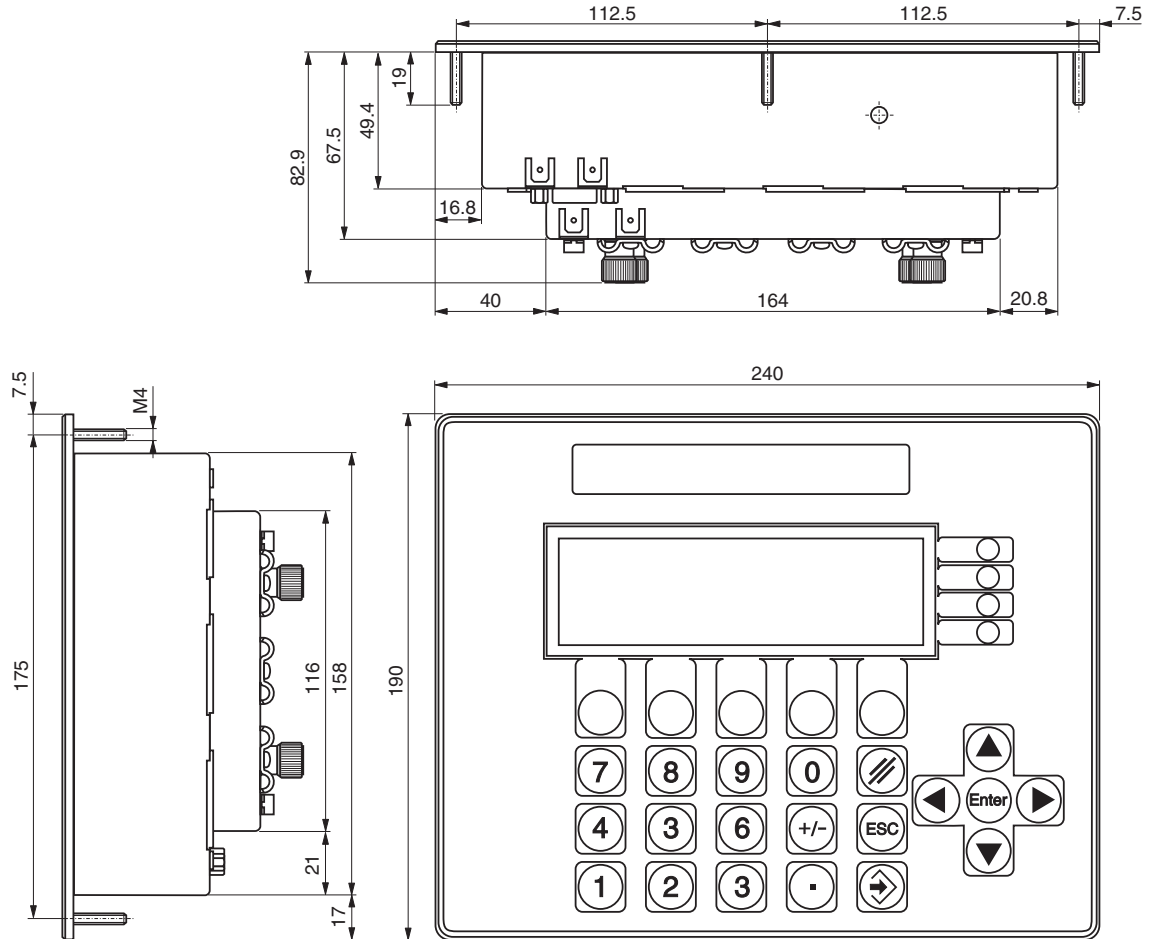
---

<sup>\*)</sup> If all 6 axes are activated the maximum value for 'V max' is reduced to 2.730.666 for internal computation reasons. In order to be able to work at  $3000 \text{ min}^{-1}$  under these circumstances the resolution must be reduced so that the above value is not exceeded (multiplier =  $8333/10000$  or smaller).

## 7 Technical data

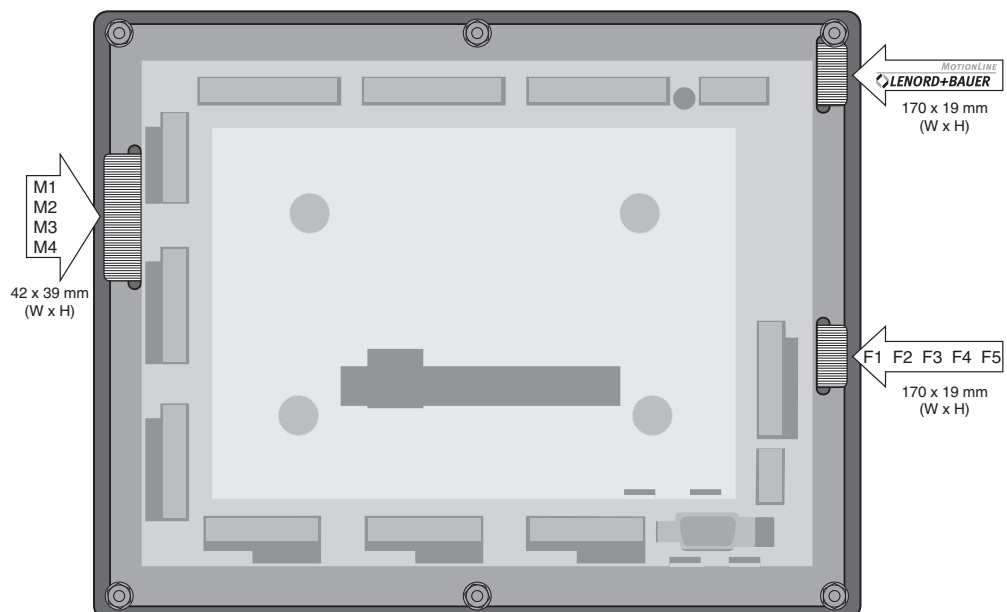
### 7.1 Dimensional diagram

#### 7.1.1 GEL 8230/8231

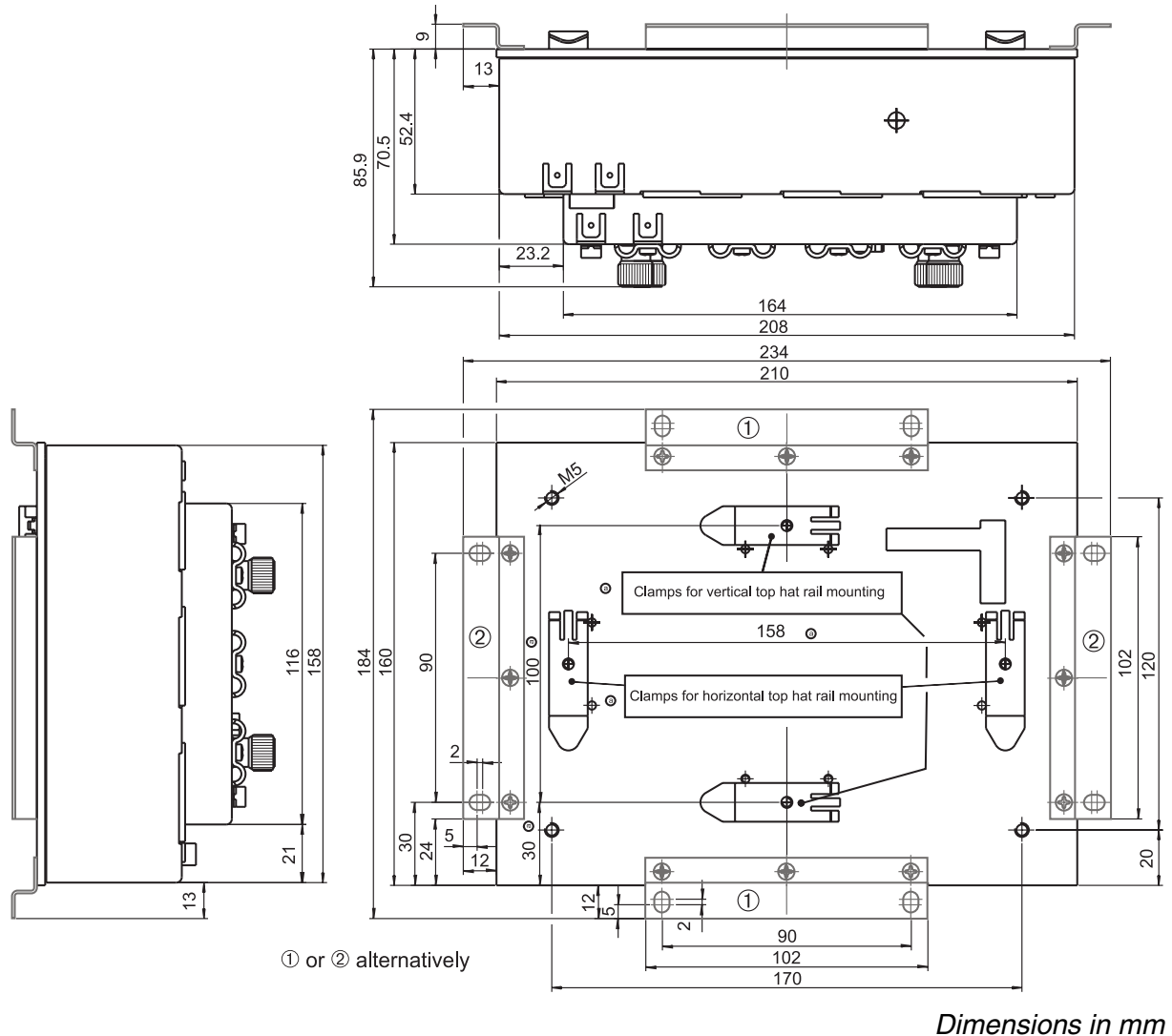


Dimensions in mm

#### Slide-in strips



## 7.1.2 GEL 8235/8236



## 7.2 Specifications

<b>Supply</b>	Voltage	19 ... 30 V DC (Controller und optional encoders)
	Max. load	
	– Controller	1 A
	– encoders	1 A (24 V) / 0.6 A (5 V)
	μprocessor	C167
<b>Control</b>	Memory:	
	RAM	1 Mbytes
	Flash	1 Mbytes
	NV-RAM	8 Kbytes
	Control sampling time	1 ms per activated axis
Internal counting range	$-2^{31} \dots +2^{31}-1$ (-2,147,483,648 ... +2,147,483,647)	

<b>Interfaces</b>	Serial (PC communication/pro- gramming)	2, with adjustable baud rate COM 1: RS 232 C or RS 422/485 COM 2: RS 232 C
	CAN bus	2 (CAN 1, CAN 2)
	– protocol	CANopen, CAN Link, LB2
	– min. contr. sampling time necessary	2 ms ( $\cong$ 2 activated axes)
	– input objects (PDOs)	4 per CAN bus, each 64 bit data width
– output objects (PDOs)	4 per CAN bus, each 64 bit data width	
– SDO objects	max. 64 per CAN bus	
	Field bus	1 extension slot for PROFIBUS-DP, InterBus-S, DeviceNet (others on request)
<b>Inputs</b>	Digital	22 (GEL 8231/8236: 30); 24 V, status indicated by green LEDs;
	– logic level	Low = 0...+5 V, High = +15...30 V
	Analog	Measuring cycle 10 ms
	– current/voltage	1 (GEL 8231/8236: 3) 0...20 mA or 0...10 V (range of values 0...1023)
	– temperature (PT100)	4 (GEL 8231/8236 only); -40...+351 °C (range of values: 0...1564)
	Encoder	3 (incremental/SSI); 5 V or 24 V
– Incremental	2 signal tracks or 1 signal track and direction signal, 1 reference track; max. input frequency 200 kHz	
– SSI	single-turn (13 bits) or multi-turn; clock frequency 125 kHz	
<b>Outputs</b>	Digital	9 × 30 mA, 6 × 500 mA 24 V, status indicated by red LEDs
	Analog	3; $\pm 10$ V, 10 mA; resolution 2 mV
	PWM clock	5 V push-pull, $f_{\max} = 100$ kHz, $f_{\min} \approx 0.3$ Hz (terminal blocks E1, E3) or 5 Hz (E2). $T_{\text{pulse, min}} \approx 5 \mu\text{s}$
<b>PLC</b>	Memory range	
	– program	256 Kbytes
	– data	128 Kbytes

	– data backup	128 Kbytes
	– non-volatile RAM	4 Kbytes
	Programming	in accordance with IEC 61131-3 with development environment CoDeSys
<b>Environment</b>	<b>application classes (KWF) according to DIN 40040</b>	
	Working temperature	0 °C ... +50 °C
	Operating temperature	-20 °C ... +50 °C
	Storage temperature	-20 °C ... +70 °C
	Relative humidity	≤ 95%, non-condensing
<b>EMC</b>	(by observance of the wiring information, see page 14)	
	Emitted interference	DIN-EN 50081-2
	Interference immunity	DIN-EN 61000-6-2
	CE sign	in accordance with EU Guideline EMC 89/336/EEC
<b>Display</b>	Type	LCD, 240 × 64 pixel, LED illumination
GEL 8230/8231	Visible area	133 mm × 39 mm
<b>Enclosure</b>	Material	Sheet steel, galvanized
	Front panel GEL 8230/8231	Aluminum with edge protection
	Mounting panel GEL 8235/8236	Aluminum
	Weight	approx. 1.7 kg
<b>Protective class</b>	Front side GEL 8230/8231	IP 65
	Mounting side GEL 8235/8236	IP 20
	Connection side	IP 20
<b>Maintenance</b>	The device is maintenance-free; clean the front plate with a damp cloth; do not use solvents.	

## Appendix 1 LB2 protocol

The LB2 protocol is an in-house company binary protocol with which most LENORD+BAUER devices can communicate with one another.

The protocol can be used for, e.g.

- implementation of user-friendly operation and monitoring with the aid of in-house PC application programs
- the communication of several MotionControllers with each other or with other LENORD+BAUER controller or display devices.

It can be used via CoDeSys function blocks and can be extended at will with own functions.

### Transmission

The transmission rate of the serial interface (RS 232 C, RS 422/485) can be set, default is 19,200 bits/s. For the telegram either no parity check or even parity check can be set, default is even parity check.

Parity check	Start bit	Data bits	Parity bit	Stop bit
none	1	8	–	1
even	1	8	1 (even)	1

In principle the protocol works **without** hardware handshake.

### General information

- For the LB2 protocol functions described here the maximum response time of the MotionController to a request can be longer than 15 ms.
- All numerical data is, if not otherwise stated, in hexadecimal format and labeled with the 'h' suffix.
- If not explained in more detail
  - **Read** or **Receive** means that the Master (PC, external PLC etc.) is expecting data from the MotionController
  - **Write** or **Send** means the transmission of data from the Master to the MotionController.

Therefore:

write/send:      Master     $\longrightarrow$     MotionController  
 read/receive:    Master     $\longleftarrow$     MotionController

**Telegram structure**

<b>Head</b>	<b>Data</b>	<b>Check byte</b>
2 bytes	n bytes	1 byte

 **Head**1<sup>st</sup> byte = 82h (LSB)2<sup>nd</sup> byte = 96h (MSB) **Data**1<sup>st</sup> byte: number of all following bytes (up to and including the **check byte**)2<sup>nd</sup> byte: function3<sup>rd</sup> up to n<sup>th</sup> byte: data **Check byte**Bit by bit XOR logic operation of all bytes of the **Data** blockThis produces the following structure for the **S**ending and **R**eceiving of data:

	Head	Number	Function	Data				Check byte
<b>S</b>	82h 96h					...		
<b>R</b>	82h 96h					...		

**! Important information for programming:**

- ▶ For **multi-byte** data words the lowest value byte (LSB) is always transmitted first and the highest value byte (MSB) is transmitted at the end.
- ▶ If a **Data** byte or the **Check byte** contains the value **82h**, this byte is sent a second time in order that it is not falsely interpreted as the first **Head-Byte**.  
This additional byte is included with neither the number (first **Data** byte) nor with the **Check byte**. The Master transmission program must take this into account when sending and receiving data.  
If, in what follows, a maximum value is given for the data bytes to be transmitted, the real value can be higher than the possible 82h repeats.
- ▶ The transmission of operating values is always carried out in **Integer number format**. A decimal point in the MotionController display is to be ignored, i.e. the value is transmitted in pure display units.  
Example: 50.83 user measuring units = 5083 display units = 000013DBh (transmission: DB 13 00 00).



- If an error occurs during or after transmission, instead of the 'normal' reply telegram with the function number an error telegram is sent back with the following structure:

	Head	Number	Function	Data	Check byte
<b>R</b>	82h 96h	03h	<b>FFh</b>	xxh	??h

xxh = error code (1 byte) with the following meaning:

a) Transmission error

xxh	Meaning
01h	Parity error Parity set differently at the transmitter and receiver or transmission error.
02h	Stop bit error Stop bit shows wrong polarity (low level).
03h	Overwrite error Characters in the transmission buffer are overwritten before they can be read out. If this error occurs the transmission rate should be reduced.
04h	Check byte error Check byte wrongly calculated or transmission error.
05h	Number error The value in the number byte does not match that of the permitted value for the function.

b) Function error

xxh	Meaning
10h	Wrong function number
20h	Wrong sub-function number (function CCh)
21h	Wrong flash/RAM bank number
22h	Flash delete error: flash component faulty
23h	Flash write error: flash component faulty
24h	Flash write protection
25h	Number of values too large The length handed over or established from the number during reading or writing of a RAM/flash bank is larger than 128.
26h	Uneven number An uneven number of data is to be written into flash. This can only happen per WORD (always 2 bytes).

xxh	Meaning
27h	Range overflow The sum of <i>Offset</i> and <i>Length</i> during reading or writing of a RAM/flash bank is larger than 65536 (FFFFh+1).

**Functions** In the following the LB2 protocol functions implemented in the operating system of the MotionController are described in numerical order.

1) Standard functions of the LB2 protocol:

**00h** Select device

**40h** Request device type

**41h** Request the version number of the operating system

**43h** Request the version number of the customized software

**50h** Read out failures

**52h** Read out number of currently stored failures

**53h** Delete all stored failures

2) Function **CCh** as **extension** to LB2 protocol, especially for the MotionController. It comprises further sub-functions:

**21h** Delete flash bank

**22h** Read flash bank

**23h** Write flash bank

**24h** Read RAM bank

**25h** Write RAM bank

**26h** Write flash bank with offset

**30h** Read parameter, simple

**31h** Read parameter, extended

**32h** Write parameter, simple (RAM)

**33h** Write parameter, extended (RAM)

**34h** Call loader

**35h** Write parameter safe against power failure into flash

**00h** Select device

	Head	Number	Function	Data	Check byte
<b>S</b>	82h 96h	03h	<b>00h</b>	<i>Device number</i>	??h
<b>R</b>	82h 96h	03h	<b>00h</b>	<i>Device number</i>	??h

- ◆ *Device number* (1 byte)  
00h ... 1Fh: device 0 ... 31

**i** The device number is defined with parameter [005]. If there is no device with the given number a timeout is triggered in the Master.  
Exception: a controller with device number **0** will **always** answer (even if another controller is addressed). Therefore, for the operation of several controllers on the interface (bus operation), none of the controllers must be programmed with device number 0.

Example: Calling the MotionController with the number 2

Transmit: | 82h 96h || 03h | **00h** | 02h || 01h |  
 Receive: | 82h 96h || 03h | **00h** | 02h || 01h |

**40h** Request device type

	Head	Number	Function	Data	Check byte
<b>S</b>	82h 96h	02h	<b>40h</b>	–	42h
<b>R</b>	82h 96h	05h	<b>40h</b>	<i>Device</i>   <i>Axes</i>	??h

- ◆ *Device* (2 bytes)  
2026h = (GEL) 8230  
2027h = (GEL) 8231 etc.
- ◆ *Axes* (1 byte)  
06h: max. number of axes

**41h** Request the version number of the operating system

	Head	Number	Function	Data	Check byte
<b>S</b>	82h 96h	02h	<b>41h</b>	–	43h
<b>R</b>	82h 96h	06h	<b>41h</b>	<i>Version</i>   <i>Revision</i>	??h

- ◆ *Version* (2 bytes)  
Version number before the separating point, e.g. **05** for V5.01
- ◆ *Revision* (2 bytes)  
Revision number after the separating point, e.g. **01** for V5.01

**43h** Request the version number of the customized software

	Head	Number	Function	Data	Check byte
<b>S</b>	82h 96h	02h	<b>43h</b>	–	41h
<b>R</b>	82h 96h	06h	<b>43h</b>	<i>Version</i>   <i>Revision</i>	??h

- ◆ *Version* (2 bytes)  
Version number before the separating point
- ◆ *Revision* (2 bytes)  
Revision number after the separating point

No customized software implemented  $\Rightarrow$  data = 00 00 00 00

**50h** Read out failures

	Head	Number	Function	Data	Check byte
<b>S</b>	82h 96h	02h	<b>50h</b>	–	52h
<b>R</b>	82h 96h	??h	<b>50h</b>	<i>Failure</i>	??h

- ◆ *Failure* (max.  $20 \times 2$  bytes)  
Structure:  
1<sup>st</sup> byte = axis that caused the failure (1...6, 0 = system failure)  
2<sup>nd</sup> byte = number of the failure (see also section 4.3.4.2, page 30):  
5 = DeltaS > DeltaS max.  
6 = DeltaS < DeltaS min.  
26 = stop bit error  
27 = parity error

- 34 = overwrite error
- 35 = check byte error
- 39 = 82h repeat missing (LB2)
- 40 = number of errors for the serial communication

If no failure is stored in the MotionController the data field is empty (*Number* = 02h).

## 52h Read out number of currently stored failures

	Head	Number	Function	Data	Check byte
<b>S</b>	82h 96h	02h	<b>52h</b>	–	50h
<b>R</b>	82h 96h	??h	<b>52h</b>	<i>Number</i>	??h

- ◆ *Number* (1 byte)  
00h...14h: number of stored failures (none ... max. 20)

## 53h Delete all stored failures

	Head	Number	Function	Data	Check byte
<b>S</b>	82h 96h	02h	<b>53h</b>	–	51h
<b>R</b>	82h 96h	03h	<b>53h</b>	<i>Number</i>	??h

- ◆ *Number* (1 byte)  
00h...14h: number of stored failures (none ... max. 20)

**CCh** Main function of the LB2 protocol extension for the MotionController with the following *sub-functions*:

### 21h Delete flash bank

	Head	Number	Func.	Sub	Data	Check byte
<b>S</b>	82h 96h	04h	<b>CCh</b>	<b>21h</b>	<i>Number</i>	??h
<b>R</b>	82h 96h	04h	<b>CCh</b>	<b>21h</b>	<i>Number</i>	??h

- ◆ *Number* (1 byte)  
0Fh, 10h: bank 15 or 16

### 22h Read flash bank

	Head	Number	Func.	Sub	Data			Check byte
<b>S</b>	82h 96h	07h	<b>CCh</b>	<b>22h</b>	<i>No.</i>	<i>Offs</i>	<i>Cnt</i>	??h
<b>R</b>	82h 96h	??h	<b>CCh</b>	<b>22h</b>	<i>No.</i>	<i>Data</i>		??h

- ◆ *No.* (1 byte)  
0Fh, 10h: bank 15 or 16
- ◆ *Offs* (2 bytes)  
0000h ... FFFFh: offset within the bank (0...65535)
- ◆ *Cnt* (1 byte)  
00h ... 80h: number of bytes to be read (0...128)
- ◆ *Data* (max.128 Bytes): data read out

### 23h Write flash bank

	Head	Number	Func.	Sub	Data			Check byte
<b>S</b>	82h 96h	??h	<b>CCh</b>	<b>23h</b>	<i>No.</i>	<i>Enabl</i>	<i>Data</i>	??h
<b>R</b>	82h 96h	04h	<b>CCh</b>	<b>23h</b>	<i>No.</i>			??h

- ◆ *No.* (1 byte)  
0Fh, 10h: bank 15 or 16
- ◆ *Enabl* (1 byte): write protection, 00h = yes, 01h = no
- ◆ *Data* (max. 64 × 2 bytes)  
The data to be written is organised per WORD. The writing procedure always starts with offset 0. The offset is automati-

cally raised by the number of written data (in this way writing to a flash storage cell several times is prevented).

## 24h Read RAM bank

	Head	Number	Func.	Sub	Data			Check byte
<b>S</b>	82h 96h	07h	<b>CCh</b>	<b>24h</b>	<i>No.</i>	<i>Offs</i>	<i>Cnt</i>	??h
<b>R</b>	82h 96h	??h	<b>CCh</b>	<b>24h</b>	<i>No.</i>	<i>Data</i>		??h

- ◆ *No.* (1 byte)  
0Ch: Bank 12
- ◆ *Offs* (2 bytes)  
0000h ... FFFFh: offset within the bank (0...65535)
- ◆ *Cnt* (1 byte)  
00h ... 80h: number of bytes to be read (0...128)
- ◆ *Data* (max. 128 Bytes): data read out

## 25h Write RAM bank

	Head	Number	Func.	Sub	Data			Check byte
<b>S</b>	82h 96h	??h	<b>CCh</b>	<b>25h</b>	<i>No.</i>	<i>Offset</i>	<i>Data</i>	??h
<b>R</b>	82h 96h	04h	<b>CCh</b>	<b>25h</b>	<i>No.</i>			??h

- ◆ *No.* (1 byte)  
0Ch: bank 12
- ◆ *Offset* (2 bytes)  
0000h ... FFFFh: offset within the bank (0...65535)
- ◆ *Data* (max. 128 Bytes): data to be written

## 26h Write flash bank with offset

	Head	Number	Func.	Sub	Data			Check byte
<b>S</b>	82h 96h	??h	<b>CCh</b>	<b>26h</b>	<i>No.</i>	<i>Offset</i>	<i>Data</i>	??h
<b>R</b>	82h 96h	04h	<b>CCh</b>	<b>26h</b>	<i>No.</i>			??h

- ◆ *No.* (1 byte)  
0Fh, 10h: bank 15 or 16

- ◆ *Offset* (2 bytes)  
Range: 0h ... FFFFh (0 ... 65535)
- ◆ *Data* (max. 64 × 2 bytes)  
Please note: A flash storage cell must not be written double using this function.

### 30h Read parameter, simple

	Head	Number	Func.	Sub	Data	Check byte
<b>S</b>	82h 96h	05h	<b>CC</b> h	<b>30</b> h	<i>Offs</i>	??h
<b>R</b>	82h 96h	09h	<b>CC</b> h	<b>30</b> h	<i>Offs</i>   <i>Value</i>	??h

- ◆ *Offs* (2 bytes)  
0000h ... 03E7h: offset within the parameter list  
(0... 999)
- ◆ *Value* (4 bytes): parameter content

### 31h Read parameter, extended

	Head	Number	Func.	Sub	Data	Check byte
<b>S</b>	82h 96h	05h	<b>CC</b> h	<b>31</b> h	<i>Offs</i>	??h
<b>R</b>	82h 96h	??h	<b>CC</b> h	<b>31</b> h	<i>Offs</i>   <i>Data</i>	??h

- ◆ *Offs* (2 bytes)  
0000h ... 03E7h: offset within the parameter list (0... 999)
- ◆ *Data* (max. 68 bytes): data structure with following content:
  - *Variant* (9 bytes): set parameter variant (text, zero-terminated)
  - *Minimum* (4 bytes): minimum parameter value
  - *Maximum* (4 bytes): maximum parameter value
  - *Default* (4 bytes): factory set for the parameter
  - *Name* (21 bytes): parameter name (text, zero-terminated)
  - *Group* (max. 26 bytes): parameter group (text, zero-terminated)



**32h** Write parameter, simple (RAM)

	Head	Number	Func.	Sub	Data		Check byte
<b>S</b>	82h 96h	09h	<b>CC</b> h	<b>32</b> h	<i>Offs</i>	<i>Value</i>	??h
<b>R</b>	82h 96h	06h	<b>CC</b> h	<b>32</b> h	<i>Offs</i>	<i>Loader</i>	??h

- ◆ *Offs* (2 bytes)  
0000h ... 03E7h: offset within the parameter list (0... 999)
- ◆ *Value* (4 bytes): parameter content
- ◆ *Loader* (1 byte): return value of the loader (see sub-function 34h)  
00h = OK, 01h...7Fh = warning, 80h...FFh = error

**33h** Write parameter, extended (RAM)

	Head	Number	Func.	Sub	Data			Check byte
<b>S</b>	82h 96h	09h	<b>CC</b> h	<b>33</b> h	<i>Offs</i>	<i>Value</i>		??h
<b>R</b>	82h 96h	??h	<b>CC</b> h	<b>33</b> h	<i>Offs</i>	<i>Loader</i>	<i>Text</i>	??h

- ◆ *Offs*, *Value*, *Loader* as before
- ◆ *Text* (max. 41 bytes): loader message in plain text

**34h** Call loader

This function must always be called directly after writing a parameter or parameter block. By doing this the values transmitted are checked for their validity, necessary calculations are carried out and the parameter values loaded into the working storage (procedure as for exiting the programming mode at the device).

	Head	Number	Func.	Sub	Data	Check byte
<b>S</b>	82h 96h	03h	<b>CC</b> h	<b>34</b> h	–	FBh
<b>R</b>	82h 96h	07h	<b>CC</b> h	<b>34</b> h	<i>Offset</i>	??h

- ◆ *Offset* (4 bytes): parameter address (offset within the parameter list) that caused an error; if no warning and no error is present FFFFFFFFh (-1) is returned.

**35h** Write parameter safe against power failure into flash


	Head	Number	Func.	Sub	Data	Check byte
<b>S</b>	82h 96h	03h	<b>CC</b> h	<b>35</b> h	–	FAh
<b>R</b>	82h 96h	03h	<b>CC</b> h	<b>35</b> h	–	FAh

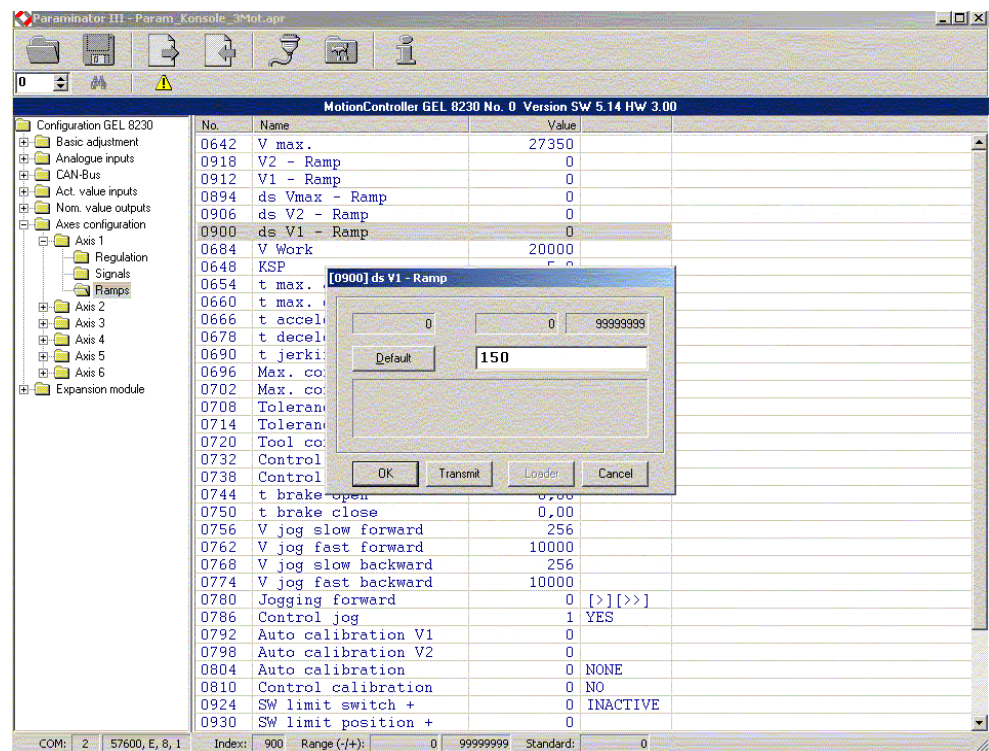
## Appendix 2 "Paraminator III" parameter editor

The parameter editor offers a comfortable possibility of programming the configuration parameters in the GEL 823x MotionController via PC keyboard and screen. Parameters can be

- read,
- modified,
- re-written,
- stored on the hard disk of the PC as ASCII file (file name: \*.apr),
- loaded from the hard disk of the PC.

The program runs under WINDOWS 9x, NT and 2000. Installation and connection are carried out as described for the upgrade utility "LingiMon" (see section 5.2).

The application language can be switched via the  icon.




### Setting up a connection

The current communication data are shown in the status line and may be adjusted via the  icon in the symbol bar:




- ! Pay attention to the transmission rate: it must be the same for both participants (default: 19200 Baud).

For establishing the connection, either

- search a MotionController via the  icon and select the desired one or
- enter (or chose) the desired device number in the  field; with the first start of a transmission a permanent connection to the MotionController selected is established.


If connection set-up is successful the designation and version data of the device are displayed underneath the symbol bars (see the screenshot).

**Functioning** After starting the program all parameters are set to their defaults.

Via the  icon the parameters can be read from the MotionController. When confirming with the "OK" button the values are loaded into the editor.



For editing a parameter select the matching list entry and confirm it with the [ENTER] key or double-click with the left mouse button on it (see the screenshot).

A modified parameter can be transmitted immediately from the input mask ("Transmit") and activated after that ("Loader").

 When transmitting parameters to the MotionController it must not currently be in the parameter programming mode. The application would respond with the error message "Invalid operating mode!".

If invalid parameters have been transmitted they will be displayed in a list where they can be edited immediately (after a double-click). In the case of transmitting individual parameters an adequate error message will appear in the input mask.

In order to make the changed parameters permanent they must be saved in the flash memory of the MotionController. The application reminds you of this by a message which is displayed either

- after transmission of all parameters at a time (via the  icon) or – at the latest –
- when exiting the application (via the close icon ) if individual parameters have been transmitted from the input mask.

You can force this process with the  icon.

If parameters have been edited you will be requested for saving the parameters in a file when closing the application.

## Index

- .b86 38, 40
- .cfg 36
- .h86 38
- .ico 36
- .lib 36
- .pdf 8
- .pro 36
- [xxx] 9
- Acceleration 60, 65
  - maximum 60
  - working 60
- Acrobat Reader 10
- Actual position 27
- Actual value error 49
- Actual value input 27, 58, 59
  - CAN bus 51
  - Object/PDO IN 51
  - source 51
- Actual value inputs 43, 48
- Actual value source
  - CAN bus 51
  - virtual 52
- Actual=nominal 61, 66
- Adaptations 35
- Analog 52
- Analog input 22
- Analog inputs 45, 52
- Analog outputs 23
- Application example 72
- ASCII interface 44
- Auto calibration 63, 65, 67
- Auto calibration control 63
- Auto start program 10
- Average 53
- Axes 43, 57
  - activated 43
- Axis
  - actual value input 58
  - controlling mode 57
  - decimal point 59
  - feedback control 59
  - feedback control factor 60
  - nominal value output 59
  - ramps 67
  - signals 65
  - virtual 58
- Axis configuration 57
- Axis control 7, 35
- Backup 40
- Baud rate 44
- Binary file 38
- Brake 56, 62, 66
- Braking 60
- Calibration 9, 50
- Calibration (→ Auto calibration)
- Calibration edge 50
- Calibration input 50
- Calibration value 9, 51
- CAN 1/2 45
- CAN bus 19, 45
  - actual value inputs 51
  - actual value source 51
  - CAN 1 19
  - CAN 2 19
  - CAN Link 46
  - nominal value destination 55
  - nominal value outputs 55
  - object 46
  - scan 32, 46
  - state 31
- CAN Link 46, 77
- CAN network 46
- CAN Remote I/O 33, 46
- CANopen 33, 46, 77
- CoDeSys 7, 36
  - interface 44
  - version 36
- COM 1/2 44
- COM port 38
- Commissioning 34, 35
- Company logo 24, 43
- Configuration 29
- Configuration menu 41
- Connections 14
- Connector coding 15
- Control factor 58
- Control factor (→ Feedback control)
- Control panel cutout 11
- Control sampling time 58, 76
- Control start delay 62, 66
- Control stop delay 62
- Control time 43
- Controlling mode 57
- Copyright 31
- Counter frequency 52
- Counting pulses 9, 49
- Dangerous situation 5
- Data format (Field bus) 69
- Data transmission failure 30
- Dead range 54
- Deceleration
  - maximum 60
  - working 61
- Decimal point 9, 27, 49, 53, 59
- Default 42
- Delivered state 35
- DeltaS 28, 59, 61
- Destination 55
- Device file 36
- Device information 30
- Device number 44
- DeviceNet 69
- Digital inputs 21
- Digital outputs 23
- Dimensional diagram 75
- DIP switch 18, 19
- Direction 54, 55, 56
- Direction recognition 49
- Display windows 26
- Download 39
- Dynamics 60
- Edge evaluation 9, 27
- EMC 14, 78
- Encoder 17
  - inputs 20
  - supply voltage 20
- Encoder, separate 55
- Error message 49
- Ethernet 70
- Example projects 36
- Extension module 69
- Extension slot 77
- Factory defaults 35, 42
- Failure diff. 49
- Failure list 30
- Failure suppression 49
- Failure tolerance 49
- Failures 30
- Fast jog 28, 65, 67
- Fast/slow jog 56, 58, 65
- Feedback control 59
  - actual=nominal state 62
  - auto calibration 63
  - brake 62
  - control factor 60
  - jerk limitation 61
  - manual operation 62, 63
  - max. tracking error 61
  - maximum acceleration 60
  - maximum deceleration 60
  - maximum speed 60
  - maximum velocity 60

- principle 59
- stop state 62
- tolerance 61, 66
- working acceleration 60
- working deceleration 61
- working velocity 60
- Feedback position control 55, 62
- Field bus 69, 77
- Field bus module 12
- Firmware 5
- Flash memory 38, 42
- Function block 57
- Function elements 7
- Function keys 24, 25
- Function module 12
- Gate time 45, 53
- HMI 36
- Home offset 9
- Home offset adjustm. 50
- HW-limit switch 64
- Hydraulic drive 58
- Increm./revolution 55
- Incremental encoder 20
- Inputs
  - analog 22
  - digital 21
  - encoder 20
- InterBus-S 69
- Internet 37
- Inverse inputs 20
- Jerk 61
- Jerk limitation 61
- Jog 28
- Jog control 63
- Jog direction 50
- Jog direction change 65
- Jog polarity assignment 63, 66
- Jogging forward 63, 66
- KSP (→ Feedback control, control factor)
- LB2 protocol 30, 44, 79
- LD 2000 33, 51, 55
- LED 21, 23
- Library 7, 36, 46
  - external 36
  - internal 36
- Library functions 8
- limit switch 63
- LingiMon 37
- Logic state 28
- Main menu 6, 29
- Main window 27
- Main window (axes) 27
- Main window (I/O) 28
- Manual keys 63, 66
- Manual operation 62
- Master 45, 52
- Master bypass 47
- Max. contour. error 61
- Maximum speed 60
- Maximum voltage 54
- MC8230\_Basic22.lib 36, 46
- MC8230\_HMI\_Basic22.lib 36
- MC8230\_HMI\_Techno22.lib 36
- MC8230\_LD100\_Basic22.lib 37
- Memory
  - Controller 76
  - PLC 77
- Menu keys 24, 25
- Menu level 24, 25, 26
- Menu structure 26
- Miniature key 39
- Minimum voltage 54
- MotionController
  - library 36
  - operating system 37
  - switch on 35
- Moving state 27
- Multiplier 9, 27, 50
- Node address 46
- Node Guarding 32
- Nominal position 61, 66
- Nominal value output 59
  - analog 54
  - CAN bus 55
  - decimal point 53
  - destination (CAN) 55
  - direction 54, 55, 56
  - maximum voltage 54
  - minimum voltage 54
  - Object/PDO OUT 55
  - reverse direction (signals) 57
  - signal output 56
  - signals 56
  - stepper motor control 57
  - switch-off point 55
  - type 53
  - voltage range 54
- Nominal value outputs 53
- Null modem cable 38
- Object
  - node address 46
  - type 46
- Object/PDO IN 33, 51
- Object/PDO IN/OUT 46
- Object/PDO OUT 33, 55
- Offset 9
- Operating data 27
- Operating system 7
  - update 37
- OPMODE 33, 55
- Outputs
  - analog 23
  - digital 23
  - nominal value outputs 53
- Parameter
  - characteristic 41
  - copy 48, 53, 57
  - designation 42
  - system 5, 6, 9, 34, 35, 41, 42
  - value 41
- parameter editor 6
- Parameter editor 91
- Parameter image 42
- Paraminator (→ Parameter editor)
- Password 29, 32, 34
  - request 34, 43
- Path profile 58
- PDF file 8
- PDO 77
- PLC 43
  - start input (Run) 21, 35
  - stop 44
- PLC program 7, 8, 35, 38, 56, 57
  - CAN Link 46
  - start 35
- Positioning 60
- Positioning characteristic 61
- PROFIBUS-DP 69
- Program cycle time 58
- Programming environment 36
- Protocol
  - user 46
- PT100 22
- Pulse width modulation 20
- Ramps 58, 67
  - braking process 67
  - curve shape 67
  - function 67
  - principle 68

- Readme.txt 8
- Real time 54, 56
- Reference point 63
- Reference value
  - jog direction 50
  - set 50
- Resolution 9
- Resolver 55
- Rev. count. direction 50
- Reverse direction 57
- Routines 7
- RS 232 C 18, 79
- RS 422/485 18, 79
- Run out 66
- Run time counter 31
- Scan 32
- Scope of display 6
- Scroll bars 25
- Scroll keys 24
- SDO 77
- Select keys 24
- Serial communication 44
- Serial interface 18
- Service 30, 34
- Service mode
  - parameter 29, 43
- Service mode ON/OFF 34
- Servo amplifier 33, 46, 51, 55
- Setting menus 26
- Shut down 65
- Signal edge 50
- Signal evaluation 49
- Signal output 54, 56
- Signal sequence 65
- Signal states 27
- Signal tracks 77
- Signals 65
  - acceleration 65
  - brake 66
  - calibration 67
  - fast jog 65
  - nominal value outputs 56
  - run out 66
  - shut down 65
  - slow jog 65
  - tolerance 66
- Simulation 52
- Single-turn bits 51
- Slave 45, 52
- Slide-in strip 24, 75
- Slow jog 28, 65
- Source 51
- SPC8230.pro 36
- Specifications 76
- Speed pre-control 59
- SSI
  - code 51
  - type 51
- SSI encoder 20
- State
  - actual value input 48
  - actual=nominal 62
  - analog input 45
  - CAN bus 31
  - stopped 62
- Stepper motor 20, 27, 52, 53, 57, 58
- Stepping frequency 57
- Stop state 62
- Stored failures 30
- SW-limit switch 63
- Sync 32
- System parameters (→ Parameter)
- Technical data 75
- Terminating resistor 18, 19
- Tolerance 61, 66
- Tool correction 61
- Tracking distance 28, 30
- Tracking error 61
- Transmission procedure 39
- Transmission rate 38
- Turning point 63
- U max. (→ Maximum voltage)
- U min. (→ Minimum voltage)
- Upload 40
- User measuring units 8, 27, 59
- V jog 62
- V max. (→ Velocity, maximum)
- V pos. (→ Velocity, working)
- Variant 41
- Velocity
  - maximum 60
  - working 60
- Velocity profile 58
- Version info 31
- Virtual function 27, 58
- Virtual1 52
- Virtual2 52, 59
- Voltage range 54
- Voltage supply 17
- Wiring 14