FAULHABER

Motion Controller with
Sine Wave Commutation for EC-Motors

Series MCBL2805

FAULHABER GROUP
We create motion

Instruction Manual

Table of Contents

General Information

The MCBL2805 was designed for Faulhaber brushless motors with linear hall sensors. This means that, even without an external encoder, low speed and positioning with a resolution up to 1/1000th of a revolution are possible. The motion controller is based on a powerful 16-bit microcontroller with excellent filtering quality.

This intelligent motion controller performs the following tasks:

- **Velocity Control:** High performance speed synchronization and the lowest possible torque variance even in the most demanding of applications. A PI-Controller makes sure that the command velocity is accurately achieved.
- **Velocity Profiles:** Ramping, triangle, trapezoidal and more complicated combined velocity profiles are available to the user.
- **Positioning Mode:** Positioning with high resolution including **limit switches and zero referencing.**
- **Additional Modes:** For example, Stepper Motor Mode to synchronize multiple motors.
- **Torque Controlling:** Achieved through current regulation.
- **Saving and Running Program Sequences:** For stand-alone positioning applications and to relieve the host PC.

Inputs and Outputs:

- **Command Value Input:** For inputing command speed values in digital or analog form. This input can also interpret a reference mark signal. Depending on the Mode, a frequency signal or a incremental encoder can be connected at this input.
- **Fault Output (Open Collector):** This output can also be programmed as a direction of rotation input, a digital input, or reference mark input.
- Third digital input
- **RS-232 Ports:** For connecting a host PC and controlling program sequences.

Setting the Parameters and Configurations

All filter settings can be set over the RS-232 port and saved to the onboard memory (EEPROM). Configuration settings (operating modes, in/outputs, etc.) can likewise be saved. Therefore, depending on the operating mode, the motion controllers can function fully in stand-alone applications.

Programming

A complete ASCII command set is available for programming and operating the motion controllers. All that is necessary is a terminal program such as the Windows Hyperterminal.

The **"FAULHABER Motion Manager"** program is available for users of Windows 95/98/NT. This program makes configuration and operation easy and allows the user to perform an online graphic analysis of the performance of the drive.

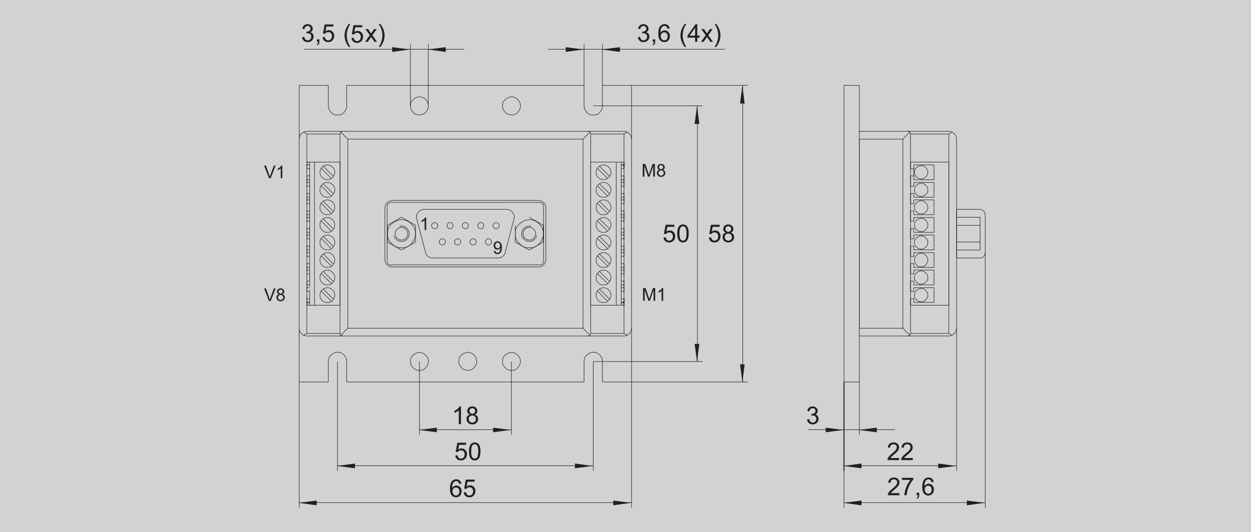
General Information

Area of Application

Ease of installation, integrated technology, compatability, size, stand-alone capability and the numerous connection possibilities allow this motion controller to perform to the highest standards in a wide range of applications, for example in decentralized automated production systems like handling or tooling machines.

Options

- FAULHABER offers the optional Multiplexer Board to operate multiple motion controllers from one RS-232 port.
- To accomadate our customers specialized needs we offer factory preconfiguring of Modes and Parameters to fit the application.
- The **"FAULHABER Motion Mananger"** is available for download from the internet at www.faulhaber.de or upon request.
- The adapters allow for simple attatchment to mounting railings.



Connections

Supply Side:

No.	Function
V1	RS-232 TXD
V2	RS-232 RXD
V3	Analog GND
V4	Fault Output
V5	Analog Input
V6	+24V
V7	GND
V8	Input No. 3

SUB-D Connector:

No.	Function
2	RS-232 RXD
3	RS-232 TXD
5	GND

Motor Side:

No.	Function	Color
M1	Phase C	yellow
M2	Hall Sensor A	green
M3	VCC (+5V)	red
M4	Signal GND	black
M5	Hall Sensor B	blue
M6	Hall Sensor C	grey
M7	Phase B	orange
M8	Phase A	brown

Data Sheet

General Specifications			
Supply Voltage	U_B	12 ... 28	V DC
PWM Switching Frequency	f_{PWM}	31,25	kHz
Efficiency	η	95	%
Max. Continuous Current	I_{cont}	5	A
Max. Peak Current	I_{max}	10	A
Current Consumption in the Electronics	I_{el}	0,06	A
Velocity Range		10 ... 15000	rpm
Out put Voltage for External Use	VCC	5	V DC
– Max. Load Current	I_{CC}	60	mA
Input No. 1[1]	Input Resistance	18	kΩ
Command Analog Velocity	Voltage Signal	± 10	V
	Slope of the Curve	1000[2]	rpm/V
Command Digital Velocity	PWM Signal	low 0 ... 0,5 / high 4 ... 30	V
	Frequency Range	100 ... 2000	Hz
	Pulse Duty Ratio 50%	0	rpm
	Pulse Duty Ratio <50%	left turning	
	Pulse Duty Ration >50%	right turning	
External Encoder / Step Frequency	f_{max}	150	kHz
Fault Output (Input No. 2)	Open collector	max. U_B / 30 mA	
	No Error	Switched to GND	
	Programmed as an input	low 0...0,5 / high 4...U_B	V
Input No. 3	Logic Level	low 0...0,5 / high 4...30	V
Port	RS-232	9600 (1200,2400,4800,19200)	Baud
Memory for Programs	Serial EEPROM	7936	Bytes
Operating Temperature Range		0 ... +70	°C
Storage Temperature Range		–25 ... +85	°C
Weight		110	g

[1] Can be set over the RS-232 port. (Factory configuration: Command Analog Velocity)

[2] Preset value. Can be changed over the RS-232 port.

Ordering Infomation:

The Motion Controllers were developed for and function only with the following motors:

1628T012B K1155	1628T024B K1155
2036U012B K1155	2036U024B K1155
2444S024B K1155	3056K012B K1155
3056K024B K1155	3564K024B K1155

The data sheets for the standard versions of these motors can be found in the catalog or the internet.

The code K1155 means that these motors have been fitted with linear Hall sensors to function with the MCBL2805

Note that the no load speed decreases to 80% of the standard version

Cables and Connections

Power Supply Requirements

The power supply should deliver at least 5 A. It is important to be careful and connect the controller with the correct polarity. Incorrect polarity will damage the protection fuse and it can only be replaced at the factory.

Analog Input (Analog Input, Analog GND = AGND)

The analog input is a differential input.

The analog-GND should be connected to the supply-GND. This avoids the effects of the voltage drop in the supply leads on the given speed value.

The analog input has, according to configuration, various applications:

- Velocity control with a voltage over the analog input (factory-installed setting)
- Velocity control with PWM through the analog input
- Zero-Referencing (Limit Switch) when used as a motion controller
- Input for the external encoder in Encoder Mode (Analog input to ground: Channel A / Analog-GND to ground: Channel B)

RS-232 Connections

The RS-232 hardware cabling consists of the TXD, RXD, and Supply-GND connectors.
The built in RS-232 port allows for a direct connection to the host PC.

Fault Output

The system is outfitted with a fault output through which system errors are signalled.

Fault Output Characteristics:

- Switch to ground (open collector)
- Output Resistance: switched through (low level): 47 Ohms, open (high level): 10 kOhms
- In the case of a system error, the switch is open (the LED is not lit.)
- Output current maximum 30 mA, voltage in open condition may not exceed supply voltage

The fault output is activated as a result of the following situations:

- Dynamic current limiting active
- Low voltage (by voltage under 10 V)
- Overvoltage protection active (by supply voltage over 32 V)
- Overtemperature protection active

The fault output port can also be configured to perform other functions:

- Pulse output
- Digital output
- Limit switch input
- Direction of rotation input

Input No. 3

The third input is a digtal input and can be used to evaluate homing points or as a control input for program sequences.

First Steps

Upon delivery the MCBL2805 is configured for the motor type 5 (2444S024B K1155).

Important: For successful operation of the motion controller with the motor of choice please read the following chapters carefully:

- Choosing the Motor Type (Page 13)
- Fine Tuning the Digital Filter (Page 19)
- Current Controller and Current Limiting (Page 40)

When operating the MCBL2805 in one of the Additional Modes please read the corresponding chapters carefully. In particular the folllowing:

- Operating Mode: Using an External Encoder to Measure Actual Position … Interfacing the external encoder (Command: ENCRES)
- Setting the Fault Output as an Input: Use the command REFIN or DIRIN and then EEPSAV before a signal is interfaced with the Fault-Pin.

If these First Steps are ignored it can lead to component damage.

Analog Velocity Control

The standard MCBL2805 is layed out as a Velocity Controller. The command velocity is given as a voltage signal at the analog input.

In this operating mode the RS-232 is not required but can be used to alter configurations. More on the topic of "Altering Configurations" to come.

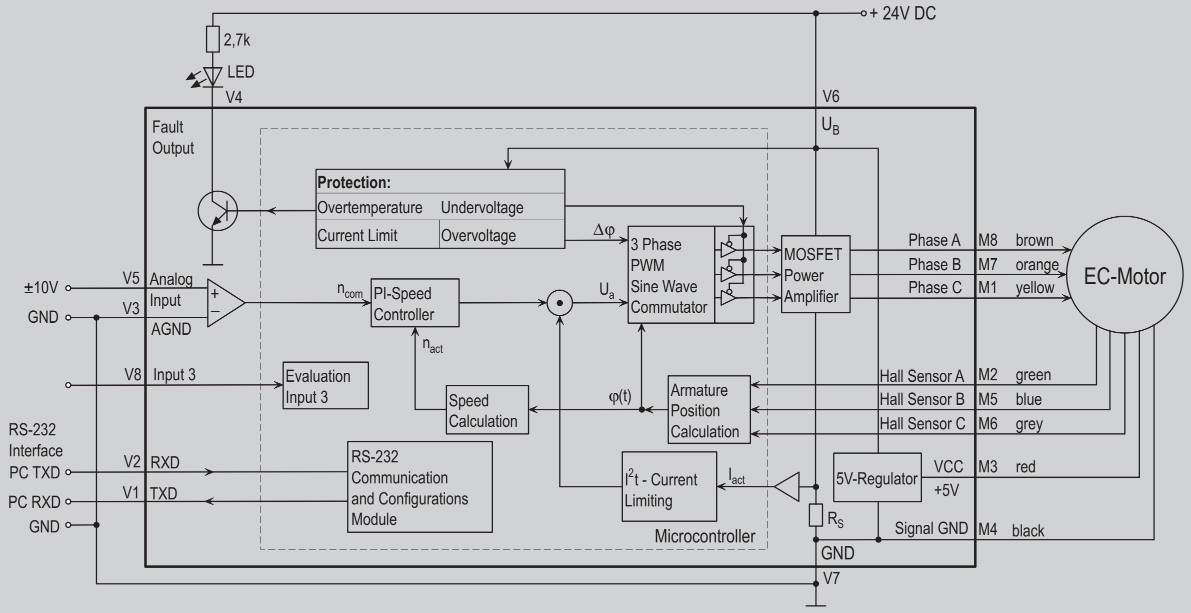


Diagram 1:
Velocity control with command velocity from the analog input.

Analog Velocity Control

Simple Possibilities to Set Command Velocity with a Potentiometer

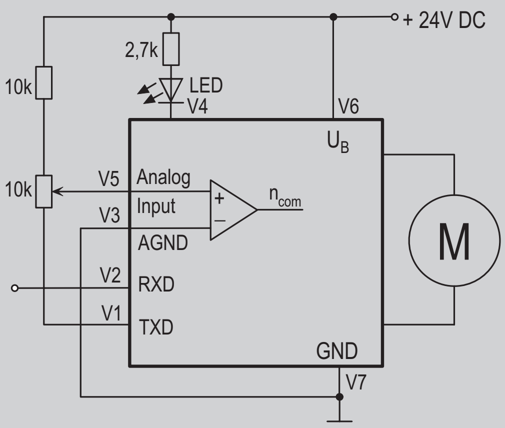


Diagram 2

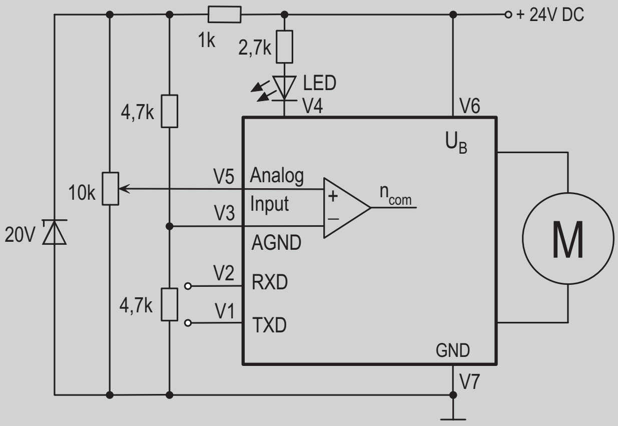


Diagram 3

Diagram 2 shows the simplest possibility, but note the following:

- The command velocity depends on the supply voltage.
- The command velocity is not very accurate due to the changes in voltage at the TXD connector.
- The RS-232 port cannot be used.

Diagram 3 shows a more precise circuit, but note the following:

- The analog GND is separate from the supply GND.
- The RS-232 is available.

Some Comments about the Input Circuit

The input circuit at the analog input is layed out as a differential amplifier. If the analog input is "open" there is already a voltage of 2 V. That means in this case that the motor would be turning at a speed of about 2000 rpm. In order to set 0 rpm the input must be connected over a low ohm resistor to the analog ground (AGND) or connected to the AGND-voltage level.

RS-232 Port and the ASCII Commands

The RS-232 port allows the MCBL2805 to be connected to a personal computer as well as various digital controllers, like for example an SPS or an IPC.

Using the RS-232 Port

- To configure the motor
- Online data analysis
- Online communications with the motor during operation

Default Configuration of the RS-232 Port

■ 9600 Baud	■ 8 data bits
■ 1 stop bit	■ No Parity

When working with a terminal program on a PC one should activate "local echo" and "carriage return".

The ASCII Commands

Communication with the PC usually takes place with a simple ASCII terminal program like the one provided with the Windows operating system. Operation becomes more elegant with the available "Moman" operating program which provides real time graphics generation for values like actual speed and position.

Building blocks of the ASCII Commands

1.) Node Address (option...just necessary in a network)

2.) Command word: a character string, letters only

3.) Number: in many cases the command word is followed by a number

4.) The conclusion is always a "carriage return". In a terminal program, for example, the enter or return key.

Example:

V 500 [enter]	... Switch to velocity control mode and rotate with 500 rpm
GST [enter]	... Get status
LA200 [enter]	... Set absolute position at 200

Spaces (blank characters) are ignored and capital and lower case letters are allowed.

The answer to the return information command is always an ASCII Character string.
At the end a "carriage rerturn" [CR] symbol (Return, decimal code 13) and a LF symbol (Line Feed, decimal code 10).

Example:

Request the actual postion (POS-Command)
Enter: POS [CR]
Answer: 50000 [CR][LF]
➔ This means that the motor is now at position 50000, which means 50 turns from zero.

RS-232 Port and the ASCII Commands

Saving Configurations

Parameters and configurations can be saved in an on board EEPROM chip. That means that saved programs and configurations are not lost in case of a loss of power. Upon connection to the supply voltage the motor runs under the setup saved in the EEPROM.

To save parameters in the EEPROM use the ASCII Command **EEPSAV.**

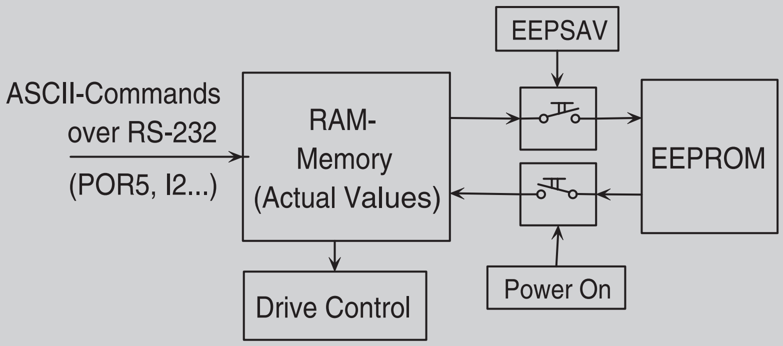


Diagram 4: The operation of the EEPROM

Important:

Setup can be lost in the case of a loss of power during programming.

Comments about the command table:

Commands that are marked in the instruction manual with a *) will be saved with the **EEPSAV** command.

Changing the Baud Rate

The baud rate can be set to the following values: 1200, 2400, 4800, 9600, 19200 Baud.

Command	Function	Description	Example
BAUD *)	Select Baud Rate	Sets the baud rate for the RS-232 port	BAUD9600

Default Setting: BAUD9600
In order to continue working after the baud rate has been changed in the motor, the baud rate must also be changed at the PC.

RS-232 Port and the ASCII Commands

Setting the Node Address

With the assistance of the RS-232 Multiplexer Board it becomes possible to drive multiple motors from one host. Use the ASCII addressing command to address the individual motors.

Command	Function	Description	Example
NODEADR *)	Define Node Address	Defines the node address from 0 to 255	NODEADR5
GNODEADR	Get Node Address	Calls up node address at the host	GNODEADR ➔ 5

Default Setting: NODEADR0

Important Note:

Only one motor may be connected at the time of addressing. Otherwise, multiple motors will receive the same node address.

Careful when hosting multiple motors:

If commands are sent without a node address, **all the motors in the network will receive the command.** If one calls up the status of a motor without entering the node address an error will occur because all the motors will attempt to answer at the same time.

Turning off the Asynchronus Answer Commands:

In the case of the Asynchronus Answer Commands problems can occur even if the node address is given because the answer is not sent directly after the command is given.
Example: NP20000 ... Notify Position (details to come)
The answer “p” will come only after the motor has reached position 20000. Other nodes (motors) could be answering at the same time. If this occurs the data (answers) can be lost.

Command	Function	Description	Example
ANSW *)	Asynchron Answer On/Off	ANSW0 ... automatic answering deactivated ANSW1 ... automatic answering active	ANSW1

Choosing the Motor Type

The Motion Controller comes standard set to the 2444S024B K1155 motor. The motor type can be changed with the following commands.

Command	Function	Description	Example
MOTTYP *)	Motor Type	Sets the type of motor to be used. (see table)	MOTTYP4
GMOTTYP	Get Motor Type	Sends the type over the RS-232 port	GMOTTYP ➔ 8

RS-232 Port and the ASCII Commands

Optional Motor Types and Parameters

No.	Motor Type	P-portion (POR)	I-portion (I)	Peak Current (mA)	Continuous Current (mA)
1	1628T012B K1155	3	6	3000	770
2	1628T024B K1155	4	10	3000	410
3	2036U012B K1155	5	6	3000	980
4	2036U024B K1155	6	5	3000	480
5	2444S024B K1155	4	15	5000	1370
6	3056K012B K1155	6	10	7000	1940
7	3056K024B K1155	12	12	3000	930
8	3564K024B K1155	8	20	8000	2800

These parameters can be individually changed or reset to the default parameters of the individual motor with the command RN.

Important: By setting the Motor Type other internal parameters will also be changed. Therefore it is important, when changing motor types on a Motion Controller, to set the type with the MOTTYP command.

Configuring the Velocity Control

Upon delivery the controller is set up as a velocity controller. The input signal is a voltage at the analog input, for example, a potentiometer. By changing the configuration of the motor many other sources for controlling the velocity can be chosen.

Sources for Velocity

Command	Function	Description	Example
SOR *)	Source for Velocity	Sources for the Velocity SOR 0: Command velocity at the RS-232 port SOR 1: Command velocity with a voltage at the analog input SOR2: Command velocity with a PWM signal at the analog input	SOR1

a.) Command Velocity with Voltage at Analog Input

In Analog Velocity Mode other setup values can be changed.

Setting the Maximum Velocity

Command	Function	Description	Example
SP *)	Load Maximum Speed	Loads new maximum velocity. Arguement in rpm (from 0 to 30000). Settings apply to all modes.	SP4000
GSP	Get Maximum Speed	Calls up maximum speed.	GSP ➔ 2500

Example: SP5000 [return] ➔ The maximum velocity is set to 5000 rpm. That means that 10 V at the Analog Input represents a command velocity of 5000 rpm

Default Setting: SP10000

Setting the Minimum Velocity

The lowest possible velocity, at minimum analog voltage, can be set.

Command	Function	Description	Example
MV *)	Minimum Velocity	Sets the minimum speed	MV100
GMV	Get Minimum Velocity	Calls up the minimum speed over the RS-232 port.	GMV ➔ 0

Default Setting: MV0

Configuring the Velocity Control

Setting the Minimal Analog Voltage

Command	Function	Description	Example
MAV *)	Minimum Analog Voltage	Sets the minimum analog voltage.	MAV500
GMAV	Get Minimum Analog Voltage	Calls up the minimum analog voltage over the RS-232 port.	GMAV ➔ 25

Example: MAV100 [return] ➔ 100 mV is the necessary minimum starting voltage. The motor will not turn if the voltage ranges from –99 mV to 99 mV.

Default Setting: MAV25

Advantages: While 0 mV at the analog input is normally very difficult to achieve, 0 rpm can not be accurately set. Setting the minimal analog voltage > 0 prevents this from being a problem. The resulting dead zone is also useful because the motor will not start up when small disturbance voltages occur.

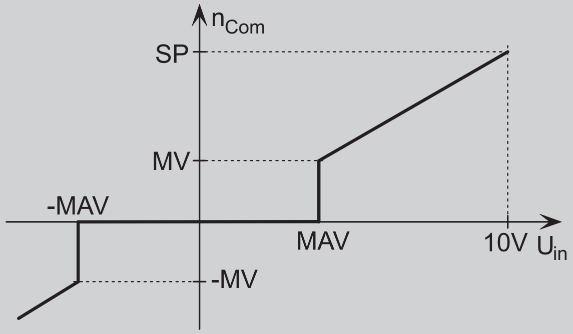


Diagram 5: Characteristic curve with a given analog velocity

Configuring the Velocity Control

Setting the Direction of Rotation

Command	Function	Description	Example
ADL *)	Analog Direction Left	Armature rotates left with positive voltage at the analog input	ADL
ADR *)	Analog Direction Right	Armature rotates right with positive voltage at the analog input	ADR

Default Setting: ADR ... Right rotation with positive voltage.

b.) Command Velocity with Pulse Width Modulation (PWM) at the Analog Input

The factory setting are:

■ Pulse Duty Ratio > 50%	➔ Rotation Right
■ Pulse Duty Ratio = 50%	➔ Rotation Stop
■ Pulse Duty Ratio < 50%	➔ Rotation Left

The commands SP, MV, MAV, ADL, and ADR can also be used in this mode.

c.) Command Velocity at the RS-232 Port

Command	Function	Description	Example
V *)	Select Velocity Mode	Activates velocity control mode and rotates with the given value.	V2000
GV	Get Velocity	Calls up the command velocity	GV ➔ 500

Directly following the SOR0-Command the motor continues rotating with the current command velocity.

Special Setting: Constant Velocity

SOR0 [enter]	➔ Switches to RS-232 Mode
V500 [enter]	➔ Enter desired velocity
EEPSAV [enter]	➔ Saves setting to EEPROM

Now the motor will always start with the saved velocity.

Configuring the Velocity Control

Setting an Acceleration

Command	Function	Description	Example
AC *)	Load Command Acceleration	Loads a new value for acceleration. Arguement in Rev/s^2	AC100
GAC	Get Acceleration	Calls up current acceleration value	GAC → 1000

Default Setting: 30000 Rev/s^2

This acceleration value makes soft acceleration and braking in Velocity Control Mode possible.

Direction of Rotation Input

The Fault Output can be configured to serve as a direction of rotation switch.

Command	Function	Description	Example
DIRIN *)	Direction Input	Sets the fault output to function as a direction of rotation input. (activates the limiter switch function)	DIRIN

Level and Direction:

Low: 0 to 0,5 V	... Rotation Left
High: 4 to Supply Voltage	... Rotation Right

The logic level at the direction of rotation input is dominant to changes made with the ADR and ADL commands.

When the Fault Output is used as a Direction of Rotation Switch, it functions with all sources after the setup is initially given over the RS-232 port.

To take the position limits into account

The position limits (LL-command) will become applicable with the input of the command **APL1.** With the input of the **APL0** command position limits will be ignored.

Configuring the Velocity Control

Fine Tuning the Digital Filter

The digital filter parameters can be adjusted to improve the dynamic performance. These parameters should be carefully chosen to fit the application because they have a great influence on performance.

Command	Function	Description	Example
POR *)	Load Proportional Term	Loads controller amplification (values: 0-255)	POR20
I *)	Load Integral Term	Loads controller integral term (values: 0-255)	I10
GPOR	Get Proportional Term	Calls up controller amplification.	GPOR ➔ 8
GI	Get Integral Term	Calls up the setting for the integral term	GI ➔ 20

Instructions:

1.) Set output configuration

- SOR0 [enter] ➔ Sets Source for Command Velocity: RS-232
- POR 8 [enter] ➔ Proportional Term = 8 (example)
- I20 [enter] ➔ Integral Term = 20 (example)
- V1000 [enter] ➔ Sets Velocity to 1/3 of Maximum Application Speed (only an example value)
- AC10000 [enter] ➔ Sets Acceleration to Maximum Application Value (only an example value)

2.) Raise the Proportional Term
POR 13 [enter] ➔ Proportional Term = 13
(Step Width 5, smaller later)

3.) Velocity Jump
V2000 [enter] ➔ Velocity jumps from 1/3 to 2/3 the Maximum Application Speed
(only an example value)

4.) Velocity Jump
V1000 [enter] ➔ Velocity jumps from 2/3 to 1/3 the Maximum Application Speed
(only an example value)

5.) Repeat steps 2-4 until the controller becomes unstable. Then reduce the Proportional Term until the stabilty is achieved.

6.) Repeat steps 2-5 with the Integral Term.

Position Control

The following Command Sequence is necessary to switch from Speed Control Mode (factory setting) to Positioning Mode:

SOR0 [enter] ➔ Switches to RS-232 Communication in Speed Control Mode

LR0 [enter] ➔ Load Relative Position 0

M [enter] ➔ Switches to Positioning Mode

Positioning commands:

Command	Function	Description	Example
M *)	Initiate Motion	Activates positioning mode and starts positioning	M
LA	Load Absolute Position	Loads the new value for the absolute position. Arguement: 1000 indicates 1 revolution.	LA100000
LR	Load Relative Position	Loads a new relative position	LR5000
SP *)	Load Maximum Speed	Loads a new maximum velocity. Arguement: rpm from 0 to 30000	SP4000
AC *)	Load Command Acceleration	Loads a new value for acceleration. Arguement: Rev/s^2 from 0 to 30000	AC100
NP	Notify Position	A "p" will appear on the screen when the motor has rotated beyond a given position.	NP10000 asynch ➔ p
NV	Notify Velocity	A "v" will appear when a given velocity has been achieved.	NV5000 asynch ➔ v

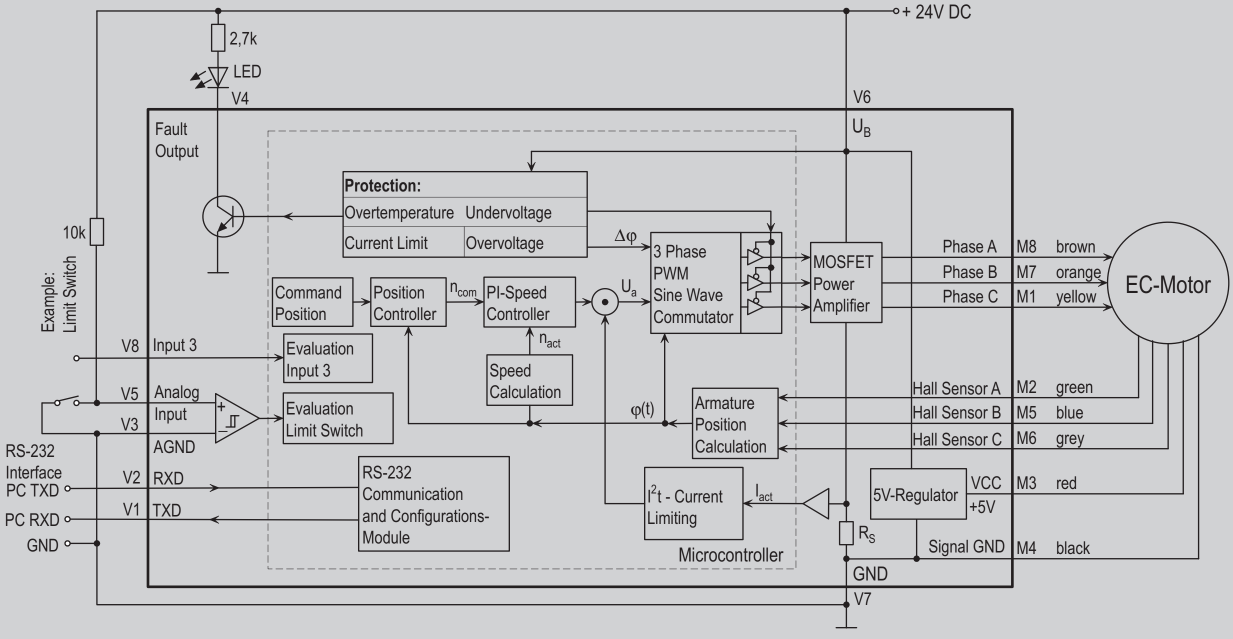


Diagram 6: Position control with a reference switch at the analog input

Position Control

Standard Positioning Sequences

1.) Enter the acceleration and the maximum velocity (rpm)

AC50 [enter] ➔ Sets acceleration to 50 Rev/s^2

SP3000 [enter] ➔ Sets maximum velocity to 3000 rpm

These values are set until they are changed or the motor is turned off.

2.) Set Command Position

Either:

a.) **LA40000 [enter]** ➔ Sets command position to Absolute Position 40000

Or:

b.) **LR10000 [enter]** ➔ Adds 10000 to the current Command Position

3.) Start Positioning Sequence

M [enter] ➔ Depending upon the choice of a.) or b.) from step 2.) the positioning sequence will start from either 40000 or current command position +10000. The motor will turn to this position with the given acceleration and maximum velocity.

By repeating steps 2.) and 3.) one can set the motor to rotate to other positions one after the other.

Combined Motion Profiles

Through well chosen values (Maximum Velocity, Acceleration, end Position) entered during a positioning sequence one can create complex motion profiles. After any values have been changed during a positioning sequence a new motion-start (the M command) must be initiated. The commands NP (notify position) and NV (notify velocity) can be used to aid in controlling the sequence.

Sequence (corresponding command sequences after the notify requirements have been met)

Start:	a.)	b.)	c.)	d.)
LA[POS3]	AC[AC2]	AC[AC1]	SP[SP2]	AC[AC4]
AC[AC1]	NV[V2]	NP[POS1]	AC[AC3]	NP[POS3]
SP[SP1]	M	M	NP[POS2]	M
M			M	

Setting the Digital Filter

The digital filter settings in Positioning Mode can also be optimized just as in Speed Control Mode (see above).

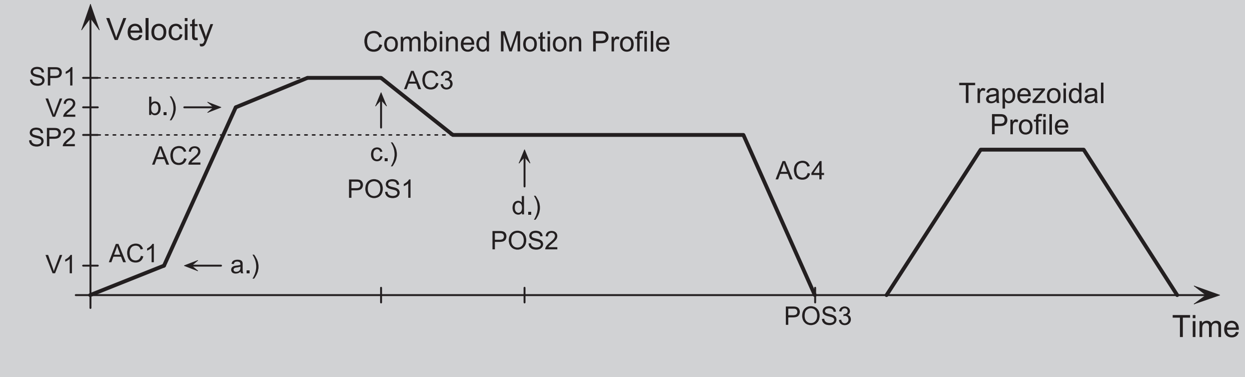


Diagram 7: Combined motion profile in comparison to a trapezoidal profile

Evaluating the Homing Points and Limit Switch

The following are the available inputs and signals:

- Analog Input
- Fault Pin (programmed as input)
- Input No. 3
- Zero Index of Hall Sensor

The inputs can be evaluated in varying ways:

- Direct programming with HA, HL, HN commands
- Programmable Homing Sequence
- Hard Blocking Function
- Hall Index Sequence

The Arguement in the Limit Switch Commands

The arguement portion of the Limit Switch Commands (HA,HL,HN,HP,HD and HB) is a number which defines the appropriate limit switch with binary code.

Allocation :

Name of Limit Switch and Input Address	Input Number	Binary	Number	Answer Symbol
Analog Input	1	0	1	h
Fault Pin	2	1	2	f
Input No. 3	3	2	4	t

A binary ‘1’ usually means an activation of that input.
A ‘0’ deactivates that input.

Example:

HA0 ... Deactivate Home Arming at all Limit Switches

HA1 ... Activate Home Arming at Analog Input, deactivate all others

HA2 ... Activate Home Arming at the Fault Pin, deactivate all others

HA3 ... Activate Home Arming at both the Fault Pin and the Analog Input

HA4 ... Activate Home Arming at Input 3

HA5 ... Activate Home Arming at Input 3 and the Analog Input

HA6 ... Activate Home Arming at Input 3 and the Fault Pin

HA7 ... Activate Home Arming at all limit switches

The command table describes only what a “1” and a “0” mean at the appropriate binary point.

Programming the Fault Pin as a Limit Switch

Command	Function	Description	Example
REFIN *)	Reference Input	Sets the Fault Output to function as a limit switch input.	REFIN
ERROUT *)	Error Output	Switches to Fault Output mode	ERROUT

Important: The Fault Pin input function is only active if the REFIN is activated. (Save setups with the EEPSAV command !)

Evaluating the Homing Points and Limit Switch

Setting the Edge and Polarity

Example circuit diagrams for the Fault Pin Limit Switch and the Analog Input are pictured in Diagram 2 and Diagram 5. The trigger edge and polarity must be chosen depending on how the limit switch is connected.

Command	Function	Description	Example
HP *)	Hard Polarity	Sets the trigger edge and polarity. of the limit switches 1: Rising edge active and logic level to High 0: Falling edge active and logic level to Low	HP3

Default Settings: HP7 ... all inputs triggered on the rising edge.

Direct Programming with the HA, HL, and HN Commands

One can define certain special actions that are triggered at the homing input with the chosen trigger edge.

- The programmed limit switch function remains until the chosen trigger edge is triggered.
- The programming can be changed with a new command before triggering at an edge.
- In the event of a power loss the HA, HL and HN-programming is erased and all limit switches are inactive.

Command	Function	Description	Example
HA	Home Arming	When edge triggers, the position is set to 0. 1: activate 0: deactivate	HA1
HL	Hard Limit	When edge triggers, the motor stops 1: activate 0: deactivate	HL3
HN	Hard Notify	When edge triggers, the motor sends a symbol 1: activate 0: deactivate	HN2 asynch ➔ f

About the HL Command:

Speed Control Mode: Upon triggering, the motor will brake with the programmed acceleration. That means that it will run out beyond the homing point. With the command “M” at the end of a sequence the motor will run out and stop directly at the reference point. **Advantage:** No abrupt changes in motion.

Positioning Mode: Upon triggering, the motor runs to the homing point at maximum speed and stops.

The HA, HL and HN commands can all be simultaneously active.

Evaluating the Homing Points and Limit Switch

Programmable Homing Sequence

The programmable homing sequence has the following advantages:

- When programmed the sequence can be called up at any time with a single command.
- When upon activation, the limit switch is already active, the motor will first run out of the switch.
- The homing sequence can be set as the first action upon activation. This makes it possible to run positioning sequences even when the motor is not connected at the RS-232 port.
- The homing sequence runs at the programmed velocity (HOSP). The direction of rotation is set with the sign in the HOSP command.

Configuring the Homing Sequence

1. Set the special homing sequence actions with the HA, HL, HN commands.
2. Save the special actions in the intermediate memory with the CAHOSEQ (Capture Homing Sequence) command
3. (Optional) Set the homing sequence to start after each POWER ON with the POHOSEQ1 (Power On Homing Sequence)
4. Further configuration commands (if necessary)
5. Save the commands to the EEPROM with the EEPSAV command. (The commands in the intermediate memory will be tranferred as well)

To make changes in the Homing Sequence repeat steps 1-5 with the new values.

POHOSEQ0 (Power On Homing Sequence Inactive) deactivates the Homing Sequence in the event of POWER ON. Save the settings to the EEPROM with the EEPSAV command.

Command	Function	Description	Example
CAHOSEQ*)	Capture Homing Sequence	Saves homing sequence to the intermediate memory. Actions that are defined with the HL,HN, and HA commands will be saved.	CAHOSEQ
POHOSEQ*)	Power On Homing Sequence	POHOSEQ1: activates homing sequence upon motor activation POHOSEQ0: deactivates homing sequence upon motor activation	POHOSEQ1
GOHOSEQ*)	Go Homing Sequence	Runs homing sequence regardless of the current mode settings. (If a Homing Sequence has been progammed)	GOHOSEQ
HOSP*)	Load Homing Speed	Loads the speed for the homing sequence Values: –30000 to 30000 rpm	HOSP100
GHOSP*)	Get Homing Speed	Calls up the current set speed for the homing sequence	GHOSP ➔ 100

Default Setting: HOSP100 ... Direction of Rotation, right, with 100 rpm.

Evaluating the Homing Points and Limit Switch

Hard Blocking Function

To ensure that the motor does not run past the limit switch, it is possible to program the limit switch as follows:

- If the drive system is at a limit switch, one direction of rotation will be blocked, that means the drive system can only move back out of the limit switch. The speed remains at 0 rpm even if the command velocity has been given with the incorrect direction of rotation.

Command	Function	Description	Example
HB *)	Hard Blocking	Activates or deactivates hard blocking at a given limit switch. 1: activate 0: deactivate	HB3
HD *)	Hard Direction	Sets which direction will be blocked. 1: Blocked right 0: Blocked left	HD2

The logic level (high, low) of the limit switch is set with the HP command.

The hard blocking function can be simultaneously active with the other limit switch commands.

Hall Indexing Sequence

The motor will run to the Hall Zero Point with the GOHIX command. The actual position value is then also set to 0.

Command	Function	Description	Example
GOHIX	Go Hall Index	Motor runs to Hall zero index position and sets position to 0.	GOHIX

The command switches automatically to the CONTMOD.

The Homing Sequence is carried out with the given Homing Speed (HOSP).

Additional Modes

Additional “special modes” were developed in order to fulfill the requirements of as many different applications as possible. There are also Modes programmable, which can work independent of the RS-232 Port connection. To increase the resolution of the system it is possible to connect an external encoder.

Command	Function	Description	Example
CONTMOD*)	Continuous Mode	Switches motor back to Continuous Mode from any of the additional modes.	CONTMOD
GMOD	Get Mode	Calls up current mode: c ... Continuous Mode s ... Stepper Motor Mode a ... Analog Positioning Mode e ... Encoder Mode, actual speed registered with an external encoder. h ... Encoder Mode, actual speed registered with Hall Sensors. g ... Gearing Mode v ... Voltage Regulator Mode	GMOD ➔ c

Stepper Motor Mode

In Stepper Motor Mode the motor rotates one “step”, a programmable angular value, for each pulse at the analog input. In this way the MCBL2805 simulates the function of a stepper motor.

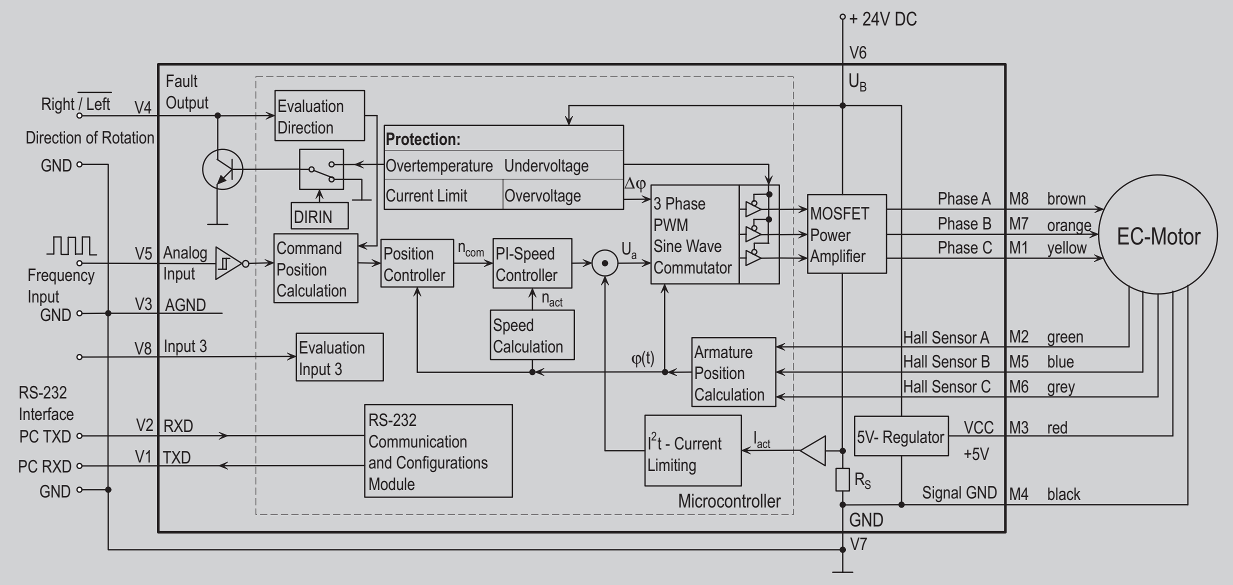


Diagram 8: Operation as stepper motor with a direction of rotation input

Additional Modes

Advantages over a conventional stepper motor:

- The step count per revolution is programmable and has very high resolution
- The step width is programmable
- No torque losses due to cogging
- The full dynamic capabilities of a brushless motor
- The motor is very quiet
- The motor monitors actual position so that no steps are "lost"
- No current flows through the motor after it has reached the predefined position
- Very efficient

Input:

Maximum input frequency: 150 kHz
Level: low 0 ... 0.5 V / high 4 ... 30 V
This mode makes true-position speed control possible and allows the user to set the ratio of input frequency to rpm with the step width and step number commands.

The reduction ratio can be calculated with the help of the following formula:

$$\text{Revolutions} = \text{Pulses} \cdot \frac{\text{STW}}{\text{STN}}$$

Revolutions	... output revolutions
Pulses	... number of pulses at the frequency input (= number of steps)
STW	... step width (Step width factor ... number of steps per pulse at the frequency input)
STN	... step number (number of steps per revolution)

STN and STW range: 0 to 65535

Command	Function	Description	Example
STEPMOD*)	Stepper Motor Mode	Switches to Stepper Motor Mode	STEPMOD
STW *)	Load Step Width	Sends the step width to the motor	STW1
STN *)	Load Step Number	Loads the number of steps per revolution	STN1000
GSTW	Get Step Width	Calls up the current step width	GSTW ➔ 1
GSTN	Get Step Number	Calls up current step number	GSTN ➔ 1000

Default settings: STN1000 and STW1

Once the motor is configured in Stepper Motor Mode, the RS-232 is no longer required.

In the event that the direction of rotation input is not yet programmed, it can be programmed with the ADR (rotate right) and ADL (rotate left) commands.

To change the direction of rotation externally, the Fault Pin can be programmed as the direction of rotation input with the DIRIN command.

In stepper motor mode the **AC**- and **SP**-values (see Position Control) are also applicable making soft acceleration and braking possible.

The command **APL1** applies position limits (LL-Command) to the sequence. With the command **APL0** the position limits will be ignored (Default setting).

Additional Modes

Gearing Mode

In Gearing Mode it is possible to connect an external encoder to provide the command position value.

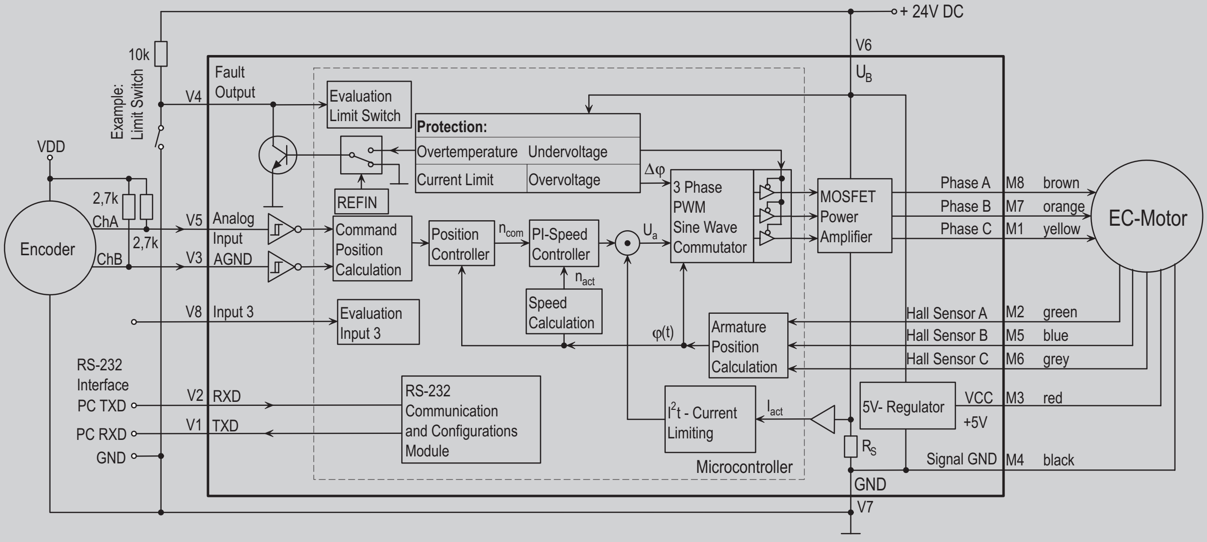


Diagram 9: Gearing Mode with a reference point at the Fault Pin

The reduction ratio is calculated just as in Stepper Motor Mode.

Command	Function	Description	Example
GEARMOD *)	Gearing Mode	Switches to gearing mode	GEARMOD

The direction of rotation can be programmed with the ADR (rotate right) and ADL (rotate left) commands or it can be set by an external signal at the Fault Pin (DIRIN command).

In gearing mode the **AC**- and **SP**-values (see Position Control) are also applicable making soft acceleration and braking possible.

The command **APL1** applies position limits (LL-Command) to the sequence.
With the command **APL0** the position limits will be ignored (Default setting).

Additional Modes

Position Control with a Voltage at the Analog Input

In this mode the command position value can be regulated with an adjustable resistor (a potentiometer), or any other adjustable analog voltage source.

Command	Function	Description	Example
APCMOD *)	Analog Position Control Mode	Switches to analog positioning mode	APCMOD
LL *)	Load Position Range Limits	Loads position limit: Provides the maximum position at the maximum analog voltage of 10 V.	LL10000

The LL command sets the position range limits.
A voltage of 10 V conforms to the predefined upper position limit.
At –10 V the motor will position at the maximum in the other direction.
Independent of the limits set with the LL command, the outer positioning limit is set in APCMOD at 1,000,000.
Important: The resolution of the analog input is limited to 10 Bit (1024 Steps).

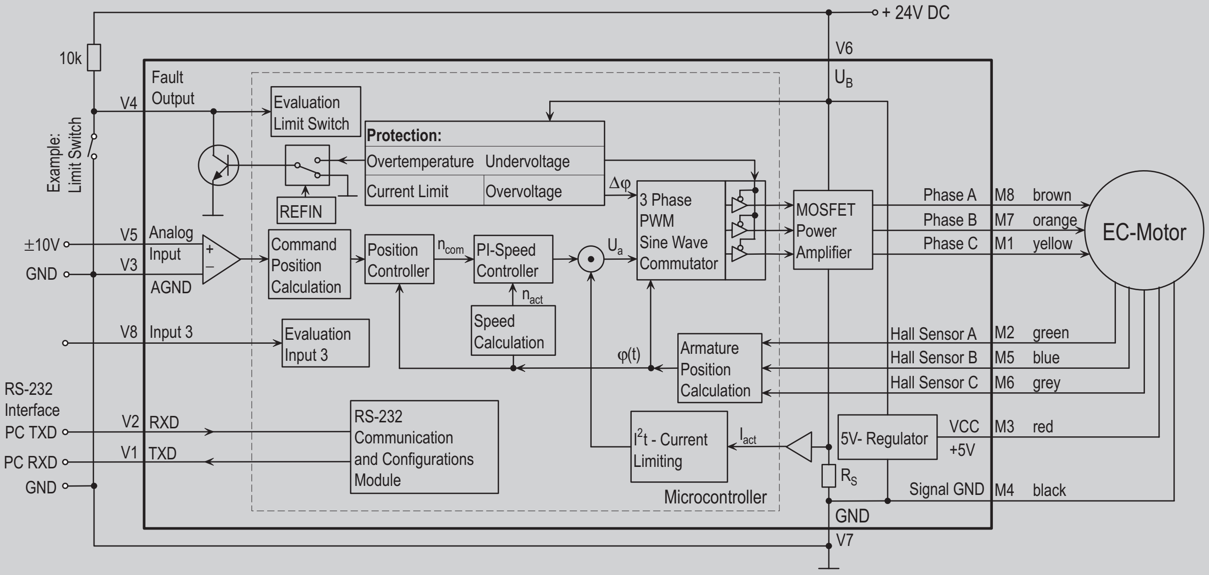


Diagram 10: Position control with an analog voltage

Additional Modes

The direction of rotation is set with the ADR and ADL commands.

ADR ➔ with a positive voltage the motor rotates in the positive direction (right).

ADL ➔ with a negative voltage the motor rotates in the negative direction (left).

The **AC-** and **SP**-values are also applicable in analog positioning mode making soft acceleration and braking possible.

Positioning with a Pulse-Width Modulated Signal

Positioning will be controlled by a PWM from the analog input when the commands **APCMOD** and **SOR2** are given one after the other. The pulse duty ration sets the command position value. A pulse duty ratio of 0.5 (ON time = OFF time) will set the command position to 0. A pulse duty ratio larger than 0.5 leads to a positive command position value and a ratio less than 0.5 leads to a negative value.

Special Function:

The presence of a Hall Sensor makes it possible to register the absolute position of the armature inside of one revolution. That means that in the event of a power loss the exact position can be recalled after the power is restored (if the armature has only moved within one revolution).

With the following commands it is possible to position the motor inside one revolution with a voltage ranging from 0 - 10 V even after the power has been turned off and then on again.

ADL	...With a positive voltage the motor will rotate to a negative position (after power is restored to the motor it is at a negative position value ranging from 0 to –1000)
APCMOD	...Switches to analog positioning
LL1000	...Sets maximum position to one revolution.
EEPSAV	...Saves the configuration.

Using an External Encoder to Measure Actual Position

A processed signal from an external encoder can be used to precisely measure actual position.

- The positioning resolution depends on the resolution of the encoder.
- Depending on the application, the speed can either be measured at the encoder or the Hall Sensor.
- The external encoder can be fastened directly to the motor shaft. The encoder can also be fastened directly to the output shaft of the complete system enabling highly accurate positioning that can be optimized at the output stage.
- The analog Hall Sensors provide the motor commutation.
- All other positioning mode functions, for example, combined motion profiles, can also be used in conjunction with the external encoder.

Additional Modes

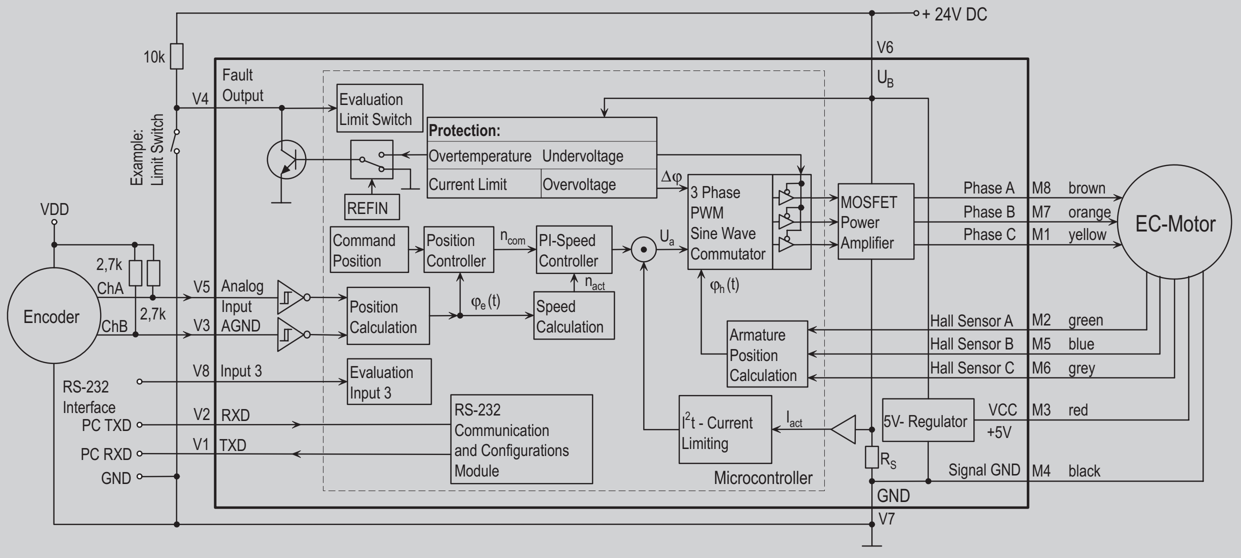


Diagram 11: External encoder a position and speed signal source

The maximum position, which is set with the LL command, can range from 0 to +2,000,000,000 and in the negative, 0 to –2,000,000,000.

Input:
Maximum Input Frequency:
150 kHz
Level: low 0 ... 0.5 V /
high 4 ... 30 V

Command	Function	Description	Example
ENCMOD*)	Encoder Mode	Switches to encoder mode. Position is registered by an external encoder.	ENCMOD
HALLSPEED*)	Hall Sensor as Speed Sensor	Speed registered by the Hall Sensors.	HALLSPEED
ENCSPEED*)	Encoder as Speed Sensor	Speed registered with an encoder signal	ENCSPEED
ENCRES*)	Load Encoder Resolution	Loads the encoder resolution to the motor. From 0 to 65535 (4 x Pulses/Revolution)	ENCRES 2048
GENCRES	Get Encoder Resolution	Calls up current encoder resolution.	GENCRES ➔ 2048

Default Setting: ENCRES2048

Important:
Because four edges are measured for every pulse, the pulse number programmed with the ENCRES command must be multiplied by four. That means, for example, that an encoder with a resolution of 1024 pulses translates to 4096 “steps” per revolution and maximum resolution can be given with the command, ENCRES4096.

Additional Modes

Voltage Regulator Mode

The controller can be configured with the VOLTMOD command to serve as a voltage regulator. The motor voltage is proportional to the input voltage at the analog input. The current limiting remains active.

Command	Function	Description	Example
VOLTMOD*)	Set Voltage Mode	Activates voltage regulator mode	VOLTMOD

In this mode it is possible to use an external controller.
The controller on the motor than functions as a power amplifier.
It is important to keep in mind that the sampling frequency of the regulator is 1,866 ms and the resolution at the analog input is 10 Bit.

Handling Errors and the Error Output

The standard error functions and the way that these errors can be rectified are described in detail in the chapter "Technical Information". Aside from these error functions, which serve mainly to protect the motor from being damaged, there is the possibility to program the unit to handle application problems.

Show Deviation from Command Speed as Error

In certain applications a greater degree of deviation from the given command speed is unacceptable. Therefore the unit should be programmed to react by displaying an error.

When the actual speed deviates to the given unnacceptable degree from the command speed, an error will be displayed. The error display and reaction can be programmed with the NE, GES and ERI commands.

The degree of deviation can be programmed:

Command	Function	Description	Example
DEV*)	Load Deviation	Loads the allowable degree of deviation of the actual to the command speed value.	DEV500
GDEV	Get Deviation	Calls up the programmed degree of deviation	GDEV ➔ 200

Default Setting: DEV30000 … This degree of deviation cannot possibly be reached. It represents the deactivation of deviation error display.

Delayed Activation of the Error Display

The delay time before displaying an error, during which the current limiting, over voltage protection or the deviation error are active can be programmed with the "Delayed Current Error" function. This makes it possible to ignore short periods of overload if, for example, in the acceleration phase the current exceeds the given limits for a short period of time.

Handling Errors and the Error Output

Command	Function	Description	Example
DCE*)	Delayed Current Error	Delayed activation of the error display for current limiting, over voltage protection, and deviation error. Given in 1/100th seconds.	DCE100
GDCE	Get Delayed Current Error	Calls up the delayed current error value	GDCE ➔ 200

Default Setting: DCE200 … 2 second delay

Example:

DCE400 ➔ The delay time is set to 4 seconds.

Short periods of overload as can occur in the acceleration phase will be ignored. If the motor is in current limiting for more than four seconds the error will be activated.

The Error Output as a Digital Output

Aside from the ability to function as a limit switch input (REFIN) and a direction of rotation input (DIRIN), the error output can also function as a digital output. This makes it possible, for example, to control a valve directly or to send a signal as a reaction to a certain event.

Command	Function	Description	Example
DIGOUT *)	Digital Output	Programs the error output as a digital output. The output is set to logic level LOW	DIGOUT
CO *)	Clear Output	Sets digital output to logic level LOW	CO
SO *)	Set Output	Sets digital output to logic level HIGH	SO

The last setting will be saved with the EEPSAV command.

Handling Errors and the Error Output

Pulse Output

Pulses from the Hall Sensor are processed and are then sent out at the fault output.

Properties of the pulse output:

- Maximum Pulse Frequency: 2000 Pulses per second
- At speeds in excess of the maximum pulses per second the maximum pulse frequency will be transmitted
- The programmed number of pulses will be reached exactly. That means you can get the right position by counting the pulses (no drift).
- The output pulses can exhibit a variance over time (lagtime).
- Because the output has only one channel the direction of rotation cannot be registered.

Command	Function	Description	Example
LPN *)	Load Pulse Number	Sets pulse number with a range of 1 to 255	LPN16
GPN	Get Pulse Number	Calls up current number of pulses	GPN ➔ 16
ERROUT *)	Error Output	Switches to Fault Output Mode	ERROUT
ENCOUT *)	Encoder Output	Switches to Pulse Output Mode	ENCOUT

Example:

LPN64 ➔ ... Sets 64 pulses per revolution

ENCOUT ➔ ... Sets pulse output mode

That means that at 1800 rpm: (1800/60)*64 = 1920 pulses per second.

At 3000 rpm: (3000/60)*64 = 3200 pulse per second. In this case, since the maximum output pulses at the fault output is 2000, drift will occur. Accurate positioning at pulse numbers above 2000 is not possible.

Saving and Running Programs

For stand-alone and partially autonomous applications, sequences can be programmed in a standard text-editor (Windows-Editor, Word) and sent to the drive with a terminal program. The programs are saved in the on-board memory (the EEPROM).

Important: The handshake control (XON/XOFF) must be set in the terminal program. To receive data from the motor (GPROGSEQ), it is best to activate the "return upon receiving" in the terminal program.

Note: The **FAULHABER MOTION MANAGER** program provides a comfortable way to program sequences and configure the motor.

Sequence Programming Commands

Command	Function	Description	Example
PROGSEQ	Program Sequence	Defines the beginning of a program. All commands given there after will be sent directly to the EEPROM. (Important: Do not cut the supply power to the controller during programming). The command END defines the end of the program. All commands given after the END will be immediately carried out by the controller.	PROGSEQ *LA1000* *NP1000* *M* END
GPROGSEQ	Get Program Sequence	Calls up the program sequence at the host PC.	GPROGSEQ ➔ <Program>
ENPROG *)	Enable Program	Starts the program. This command can also be saved with the EEPSAV command. and the program will then run directly after turning the power to the controller on.	ENPROG
DIPROG *)	Disable Program	Deactivates the program.	DIPROG

Commands can be sent over the RS-232 Port even while a program is running.
When programming a sequence almost all the ASCII commands are available.

Saving and Running Programs

Controlling a Program Sequence

The following are added commands for controlling a program sequence while it is running.
These commands are only available while the program sequence is running.

When the following commands are used the current program is interrupted until a certain criteria is fulfilled:

NP ... Notify Position ... The program is interrupted at next "M"-Command until a certain position is achieved

HN ... Hard Notify ... The program is interrupted at 'GOHOSEQ'-Commands or at next "M"- or "V"-Command until the limiter switch has been reached

NV ... Notify Velocity ... The program is interrupted at next "M"- or "V"-Command until a certain speed is achieved

GOHIX ... Go Hall Index ... The program is interrupted at "GOHIX"-Command until the Hall zero index is reached

Additional Commands (Most are only available while programming a sequence):

Command	Function	Description	Example
DELAY	Delay	Stops the sequence for a period of time defined with: Arguement: in 1/100th of a second Range: 0 to 65535	Delay100
TIMEOUT	Timeout	The motor waits a defined period of time when notify commands are being carried out and then the sequence will continue. Arguement: 1/100th of a second. Range: 0 to 65535 An answer of "o" at the RS-232 host means that the notify requirements were not met. The command can also be programmed with the RS-232	TIMEOUT5 asynch ➔ o
JMP	Jump	Jumps to a given address. The command can also be programmed with the RS-232.	JMP5
JPH	Jump if Hard Input Activated	Jump to a given address if the analog input is active (HP command defines the polarity.)	JPH3
JPF	Jump if Fault Input Activated	Jumps to a given address when the Fault Pin Input is active. (The HP command defines the polarity) Fault Pin must already be configured as an input) (REFIN)	JPF4
JPT	Jump if Third Input Activated	Jump to a given address if the third input is active (HP command defines the polarity.)	JPT53
SETx	Set Variable x	Set variable x (A,B,C) to a value in range 0 to 65535.	SETA10
GETx	Get Variable x	Calls up current settings of variable x	GETA ➔ 123
DxJNZ	Decrement x, Jump if not Zero	Decreases the value of x by 1 and jump if the value has not reached 0.	DBJNZ8
ERI	Error Interrupt	An error interrupt is activated. That means that when an error occurs the program will jump to an error subroutine in order to correct the problem. The subroutine must end with a JMP or RETI command.	ERI5

Saving and Running Programs

Continuation: Additional Commands

Command	Function	Description	Example
RETI	Return Error Interrupt	Jumps back from the error subroutine to the main program. Important: The interrupted command will not repeat upon jumping back to the main program even if it hadn't fully been carried out before the interrupt.	RETI
DIERI	Disable Error Interrupt	The ERI command is deactivated. That means the error subroutine will not run in the event of an error.	DIERI
CALL	Call Subroutine	Calls up a subroutine	CALL17
RET	Return from Subroutine	Return to the main program from a subroutine. Note that it is not possible to make a sub-subroutine. Only one level is possible.	RET
A	Define Address	Defines the actual position in the program as the jump-in point. Range: 0 to 63	A37

More About Commands and Functions

The Jump Command

The jump command **(JMP)** makes it possible to control the program sequence directly.

The jump command can also be used while connected at the RS-232 port. This becomes useful when different programs are to be called up on the computer.

Example:

```
PROGSEQ
A1
JMP1  ... continuous loop
A2    ... Program sequence 2 ... can only be called
          up with the JMP2 command at the
          RS-232.
LA10000
NP
M
JMP1  ... jumps back to continuous loop
A3    ... Program sequence 3 ... can only be
          called up with the JMP3 command at
          the RS-232.
LA-10000
NP
M
JMP1  ... jumps back to the continuous loop.
END
```

The program sequences after the A2 and A3 can only be called up at the RS-232 terminal. A JMP2 in this case means that the motor will run to position 10000 and stop.

The **DxJNZ** commands serve to create programming loops with predefined numbers of runs.

The commands **JPH**, **JPF** and **JPT** make jumps possible when a specific predefined input is activated. This makes it possible to call up programs over an external switch.

The **A command** is used to define the jump destination. The program sequence will then continue after the jump destination.
The value range for jumps is from 0 to 63. That means that 64 possible jump destinations can be set with the JMP, JPx, ERI and CALL commands.

Saving and Running Programs

About the ERI Command

At first after entering the ERI command nothing happens. The command only takes effect after an error occurs. The program then jumps to the given address.

In this way a continuation of the program even after an error occurs is possible.

With the RETI command, the program jumps back to the position it was in before the error occured. Note that the command that caused the error (or the point at which the error occurred in the program) will not be repeated.

The program will continue from the next part of the sequence. Once an error has occured no other errors can be registered until the RETI or the JMP command have been executed. Therefore the error routine must be programmed to make sure that the error "disappears" or is corrected before the program resumes. If not, the error routine will be called up repeatedly.

About the Homing Sequence

With the HN command the program can be stopped until the limiter switch has been reached. In order to be able to use the GOHOSEQ command effectively during a program, the HN command must be set correctly in the Homing Sequence.

This is especially important when a Power On Homing sequence is to be run (POHOSEQ1).

About the Notify Commands

With the Notify commands it is possible to create very complex motion profiles.

Example:

```
LA10000
SP5000
AC50
NV1000
M
AC100
NV2000
M
AC50
NP
M
```

The acceleration is increased when the motor reaches 1000 rpm and then is decreased again when it reaches 2000.

Note: When the NP command is used without an arguement the program is interrupted until the predefined position is achieved.

About the CALL Command

The CALL command makes it possible to call up subroutines at any time from various sources.

Once a subroutine is called up all commands are available except the CALL command.
To return to the main program, use the RET command.

In General

After a program has been run and is complete (and without a JMP command at the end) the motor sends an 'n' to the RS-232 if the ANSW1 is set.

To create a continuous loop, for example for stand-alone applications, a JMP command at the end of the program is necessary.

Memory Size: 7936 Bytes

Commutation with a Sine Wave

Sine wave commutation means that the rotating magnetic field is always ideally positioned to the armature. This minimizes torque variance even when the motor is rotating very slowly. The motor is also very quiet.

Current Controller and Current Limiting

The MCBL2805 comes equipped with a current controller which makes torque limiting possible.

It is possible to program 2 values for current.

1.) Peak Current

ASCII Command: LPC8000 ... Limits peak current at 8000 mA

The current will be limited at this value as long as the calculated thermal current model remains noncritical.

2.) Continuous Current

ASCII Command: LCC2800 ... Sets continuous current to 2800 mA

If the thermal model goes critical the motor will automatically switch from the peak to the continuous current value.

Operating Mode of the Current Controller

When the power is turned on to the motor, the current controller sends the value for peak current as the command current value.

With increasing load the current will increase until the peak current has been reached. The current controller then limits the current at the peak level.

An approximate thermal model is calculated directly from the actual current through the motor during operation. If the thermal model rises above a critical value the controller will automatically limit the current to the lower continuous current value.

The current limiting will switch back to the peak value only after the motor temperature has fallen below the critical value. This protects the motor from overheating but also allows for extreme loading over short periods of time for extremely dynamic applications.

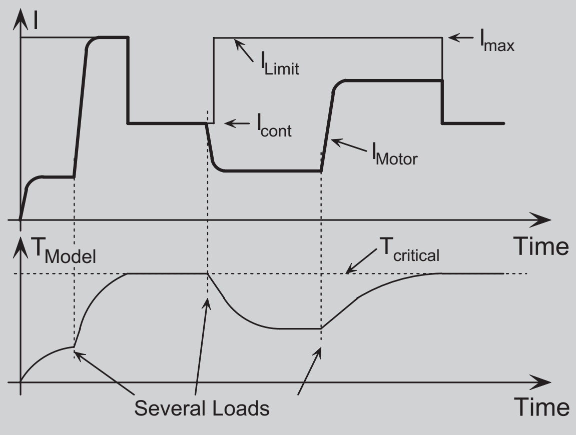


Diagram 12: The I²t current limiting

Technical Information

Analog command current

- The **SOR3** command can be used to switch to the analog command current input. The current limit is proportional to the voltage at the analog input. The I²R current limiting is no longer active.
- The set current is compared to the maximal current (LPC-value). When 10 V are registered at the analog input the maximum current will be limited accordingly.
- The command speed can be given with the V-command. Positioning can be done with the M-command. Additional modes are not possible here due to the fact that they normally use the analog input. (Exception: IXRMOD)
- Negative voltage values at the analog input also limit the current in proportion to the voltage. The direction of rotation remains the same as with positive voltages.

Overtemperature Protection

The Motion Controller will automatically shut down if the temperature in the Power MOSFETs increases beyond the predefined peak value. The following criteria must be fulfilled in order to reactivate the motor:

- The temperature must fall below the peak value
- The command velocity must be set to 0
- The actual motor speed must be lower than 50 rpm

About Measuring the Temperature of the MOSFET

The temperature of the Motion Controller is measured on the casing and compared to the measured current value to calculate the power loss.

With the help of a thermal model the controller calculates the MOSFET temperature. In most cases this method performs as over temperature protection for the MCBL2805.

Undervoltage Supervision

If the supply voltage falls below 10 V the motor deactivates and an error is shown at the fault output (LED goes dark). The electronics remain active. When the supply voltage rises above 11 V the motor will reactivate.

Overvoltage Protection

If the motor is driven as a generator it produces energie. Normally, voltage sources are not capable of reabsorbing this added energie.

For this reason the supply voltage to the motor increases and depending on the speed of the motor this can lead to a voltage value above the motors specified limits.

The motor is equipped with an overvoltage protection to avoid damaging or destroying any internal components.

If the voltage rises above 32 V the polar angle is changed and the energie produced by the motor will decrease limiting the voltage to 32 V.

Appendix

Electromagnetic Compatability (EMC)

The motion controller MCBL2805 was tested and measured for EMC according to the european guideline (89/336/EWG).

The System fulfills the following requirements (norms) during operation at nominal values:

EMC emissions according to the range defined by VDE 0839 part 81-2 (EN50081-2)

EMC immunity according to VDE 0839 part 82-2 (EN50082-2):

- Electrostatic discharges of 4kV (through contact) and 8 kV (through air) according to VDE 0847 part 4-2 (EN61000-4-2)
- HF-fields according to VDE 0847 part 4-3 (EN61000-4-3)
- Fast transients according to VDE 0847 part 4-4 (EN61000-4-4)
- Cable bourne HF-currents according to VDE 0847 part 4-6 (EN61000-4-6)

The forementioned requirements are met when the following conditions are fulfilled:

- Operation according to the instruction manual and given system data
- The supply cables should be wound through a ferrite tube, at least one winding, for example, through a Kitagawa R1-16-28-7, nearly to the controller
- Use of shielded motor supply cables (if cable length is more than 30 cm)
- All components are grounded

Appendix

The RS-232 Multiplexer Board

Multiple motors can be controlled from one host with the help of the Multiplexer Board.

On board are an input SUB-D connector to connect to the host PC and an output SUB-D connector to connect with the next motor.

It is possible to chain up to 255 motors. It is also possible to chain the 3564K024B C with other Faulhaber motion controllers, for example, the MCDC2805 and the MCBL2805.

It is important to remember that when controlling multiple motors from one host a strict master-slave structure must be applied.

That means that the host PC sends the commands to the motors and motors only answer when called.

Therefore asynchronus answers must be prevented with the ANSW0 command.

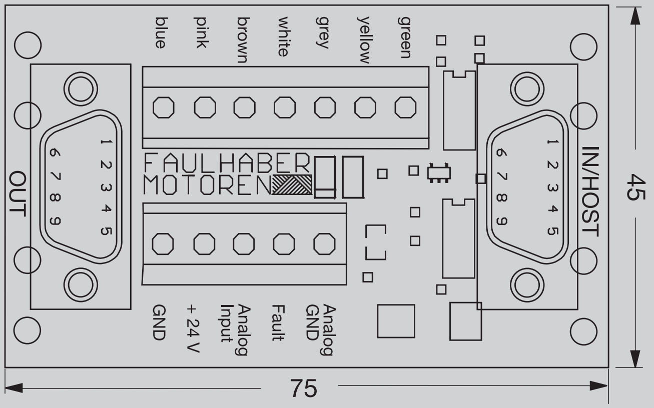


Diagram 13: The RS-232 Multiplexer Board

Appendix

The ASCII Command Set

- All commands that are marked with a *) will be saved with the **EEPSAV** command.
- In the following list, examples of possible answers to request commands are indicated with an arrow.
- Answering Commands, whitch take effect only after a certain criteria has been met, are marked by the word “asynch”.

Commands for Basic Settings

Command	Function	Description	Example
DI *)	Disable Drive	Deactivates the motor (Power Stage)	DI
EN *)	Enable Drive	Activates the motor (Power Stage)	EN
BAUD *)	Select BAUD Rate	Sets the baud rate for the RS-232 port.	BAUD9600
NODEADR *)	Define Node Address	Defines the node address anywhere form 0 to 255.	NODEADR5
GNODEADR	Get Node Address	Calls up the node address.	GNODEADR ➔ 5
MOTTYP	Motor Type	Sets the motor type (see table)	MOTTYP 4
GMOTTYP	Get Motor Type	Calls up the motor type	GMOTTYP ➔ 4
ANSW *)	Asynchronus Answer On/Off	ANSW0... no automatic answers. ANSW1... activate automatic answering	ANSW1
LL *)	Load Position Range Limits	Loads position range limits. The motor will not pass these limits in positioning mode. Positive values give the upper limit and negative values the lower limit.	LL2000000 LL-5000000
APL *)	Activate (Deactivate) Position Limits	APL1 : Activate position limits even while in speed control and stepper motor mode APL0: Deactivate Limits (default)	APL1
GPL	Get Positve Limit	Calls up the upper positioning limit at the RS-232.	GPL ➔ 60000000
GNL	Get Negative Limit	Calls up the lower positioning limit at the RS-232.	GNL ➔ -30000000
RN *)	Reset Node	Resets the parameter at the node address to the original settings (current, acceleration, controller parameters, maximum speed, position limits etc.)	RN
EEPSAV	Save to EEPROM	Saves the actual parameters and configurations to the EEPROM. The settings remain saved even when the supply voltage is turned off. Upon the next power on the setting are recalled.	EEPSAV
FCONFIG	Factory Configuration	All configurations and values will return to the factory default values. The motor will then deactivate. It can be reactivated by turning the power on again.	FCONFIG

Appendix

Commands for Motion Control

Command	Function	Description	Example
M *)	Initiate Motion	Activates positioning mode and starts positioning	M
LA	Load Absolute Position	Loads new absolute position. Arguement: 1000 is one rotation.	LA100000
LR	Load Relative Postion	Loads a new relative position	LR5000
V *)	Select Velocity Mode	Activates velocity control mode and rotates with the given value.	V2000
GV	Get Velocity	Calls up command velocity	GV ➔ 500
NP	Notify Position	Motor sends a “p” to the host terminal when the given position has been passed. No arguement: “p” is sent when target position is reached.	NP10000 asynch ➔ p
NV	Notify Velocity	When the motor reaches the given velocity it transmits a “v” to the host.	NV5000 asynch ➔ v
NPOFF	Notify Position Off	Deactivates a given Notify Position Command that has not yet been carried out.	NPOFF
NVOFF	Notify Velocity Off	Deactivates a given Notify Velocity Command that has not yet been carried out.	NVOFF

Commands for Velocity Control

Command	Function	Description	Example
MV *)	Minimum Velocity	Sets the minimum speed in rpm	MV100
GMV	Get Minimum Velocity	Calls up the minimum velocity on the host terminal	GMV ➔ 0
MAV *)	Minimum Analog Voltage	Sets the minimum analog voltage	MAV500
GMAV	Get Minimum Analog Voltage	Calls up the minimum analog voltage on the host terminal.	GMAV ➔ 25
ADL *)	Analog Direction Left	The armature rotates left when positive voltage is applied at the analog input.	ADL
ADR *)	Analog Direction Right	The armature rotates right when positive voltage is applied at the analog input.	ADR
SOR *)	Source For Velocity	Chooses source for velocity. SOR 0: Command velocity from RS-232 port SOR 1: Command velocity from a voltage at the analog input. SOR 2: Command velocity with a PWM signal at the analog input SOR 3: Command current limit value at the analog input (command speed is given with the V-command at the RS-232 port)	SOR1

Appendix

Commands For Evaluating Homing Points and Limit Switches

Command	Function	Description	Example
HO	Define Home Position	With no Arguement: Sets the actual position to 0 With Arguement: Sets actual position to the given value	HO
HP *)	Hard Polarity	Sets the trigger edge and the polarity for the limiter switches: 1**): rising edge and high level 0**): falling edge and low level	HP3
HA	Home Arming	At an edge the position value will be set to 0 1**): activate 0**): deactivate	HA1
HL	Hard Limit	At an edge the motor will stop 1**): activate 0**): deactivate	HL3
HN	Hard Notify	At an edge a symbol will be sent to the host over the RS-232. 1**): activate 0**): deactivate	HN2 asynch ➔ f
CAHOSEQ *)	Capture Homing Sequence	Saves homing sequence to the intermediate memory. Actions that are defined with the HL,HN, and HA commands will be saved.	CAHOSEQ
POHOSEQ *)	Power On Homing Sequence	Activate the homing sequence upon turning the motor on. 1**): activate 0**): deactivate	POHOSEQ1
GOHOSEQ	Go Homing Sequence	Executes the programmed homing sequence without regard to the current mode.	GOHOSEQ
HOSP *)	Load Homing Speed	Loads the homing speed. Range: -30000 to 30000 rpm	HOSP100
GHOSP	Get Homing Speed	Calls up the homing speed at the RS-232	GHOSP ➔ 100
HB *)	Hard Blocking	Activates the hard blocking function for the given limit switch 1**): activate 0**): deactivate	HB3
HD *)	Hard Direction	Sets the direction to be blocked. 1**): right rotation blocked 0**): left rotation blocked	HD2
GOHIX	Go Hall Index	Motor runs out to Hall zero and sets the position value to 0	GOHIX
HS	Hard Status	A number 1-7 indicating which limit switch/es has/have activated. (binary code)	HS ➔ 3

Appendix

Continuation: Commands For Evaluating Homing Points and Limit Switches

Command	Function	Description	Example
GAHS	Get Actual Homing Status	5 numbers with values between 0 and 7 are sent to the host over the RS-232. They indicate the status of the homing switches. 1. HA value 2. HL value 3. HN value 4. HB value 5. HD value	GAHS ➔ 33300
GHSC	Get Homing Sequence Configuration	3 numbers with values between 0 and 7 are sent to the host over the RS-232. They indicate the settings of the homing sequence. 1. HA value 2. HL value 3. HN value	GHSC ➔ 220

**) 0 or 1 at the given binary position

Commands for the Actual Parameters

Commands	Function	Description	Example
SP *)	Load Maximum Speed	Loads a new maximum velocity. Arguement in rpm. Range: 0 to 30000 For use in all modes.	SP4000
GSP	Get Maximum Speed	Calls up actual maximum velocity.	GSP ➔ 10000
AC *)	Load Command Acceleration	Load new acceleration. Arguement in Rev/s^2 (Range: 0 to 30000)	AC100
GAC	Get Acceleration	Calls up acceleration	GAC ➔ 500
POR *)	Load Proportional Term	Load controller amplification (Range: 0 to 255)	POR8
GPOR	Get Proportional Term	Calls up the controller amplification	GPOR ➔ 8
I *)	Load Integral Term	Load integral term. (Range: 0 to 255)	I20
GI	Get Integral Term	Calls up integral term	GI ➔ 20
LPC *)	Load Peak Current Limit	Loads peak current limit Range: 0 to 12000 mA	LPC8000
GPC	Get Peak Current	Calls up peak current	GPC ➔ 8000
LCC *)	Load Continuous Current Limit	Loads continuous current limit Range: 0 to 12000 mA	LCC2800
GCC	Get Continuous Current	Calls up continuous current	GCC ➔ 2800

Appendix

Commands for Additional Modes

Command	Function	Description	Example
CONTMOD *)	Continuous Mode	Switches from the present further mode back to continuous operating mode. (Hall Sensor gives actual position. Communication over the RS-232)	CONTMOD
STEPMOD *)	Steppermotor Mode	Switches to stepper motor mode.	STEPMOD
STW *)	Load Step Width	Sends a step width to the motor.	STW1
STN *)	Load Step Number	Send a step number to the motor	STN1000
GSTW	Get Step Width	Calls up step width	GSTW ➔ 1
GSTN	Get Step Number	Calls up step number	GSTN ➔ 1000
GEARMOD *)	Gearing Mode	Switches to gearing mode	GEARMOD
APCMOD *)	Analog Position Control	Switches to analog positioning mode	APCMOD
ENCMOD *)	Encoder Mode	Switches to encoder mode. Uses external encoder for actual position. (Upon switching the position is automatically set to 0)	ENCMOD
HALLSPEED *)	Hall Sensor as Speed Sensor	Actual speed is given by the Hall Sensors.	HALLSPEED
ENCSPEED *)	Encoder as Speed Sensor	Actual speed is given by the external encoder.	ENCSPEED
ENCRES *)	Load Encoder Resolution	Sends the encoder resolution to the motor. Range: 0 to 65535 [4 x (Pulse/Revolution)]	ENCRES512
GENCRES	Get Encoder Resolution	Calls up the resolution of the external encoder	GENCRES ➔ 512
VOLTMOD *)	Set Voltage Mode	Switches to voltage regulator mode.	VOLTMOD
GMOD	Get Mode	Calls up the present mode. c ... normal operating mode s ... stepper motor mode a ... analog positioning mode e ... encoder mode, speed at the external encoder h ... encoder mode, speed at the Hall Sensors g ... gearing mode v ... voltage regulator mode	GMOD ➔ c

Appendix

Commands for Configuration of the Error Functions and Error Output:

Command	Function	Description	Example
DIRIN *)	Direction Input	Programs the fault output as a direction of rotation input. (The limiter switch is also thereby activated)	DIRIN
REFIN *)	Reference Input	Programs the fault pin as a limit switch	REFIN
ERROUT *)	Error Output	Switches to error ouput mode.	ERROUT
ENCOUT *)	Encoder Output	Switches to pulse output mode.	ENCOUT
LPN *)	Load Pulse Number	Sets the number of pulses. Range: 1 to 255	LPN16
GPN	Get Pulse Number	Calls up the number of pulses.	GPN ➔ 50
DIGOUT *)	Digital Output	Programs the fault output as a digital output. The level is automatically low upon activation.	DIGOUT
CO *)	Clear Output	Sets digital output level to low.	CO
SO *)	Set Output	Sets digital output level to high.	SO
DCE *)	Delayed Current Error	Delayed activation of the error display for current limiting, over voltage protection, and deviation error. Given in 1/100th seconds.	DCE100
GDCE	Get Delayed Current Error	Calls up the delayed current error value	GDCE ➔ 200
DEV *)	Load Deviation	Loads the allowable degree of deviation of the actual to the command speed value.	DEV500
GDEV	Get Deviation	Calls up the programmed degree of deviation	GDEV ➔ 200

Commands for Programming Sequences

Command	Function	Description	Example
PROGSEQ	Program Sequence	Defines the beginning of a program. All commands given thereafter will be sent directly to the EEPROM. (Important: Do not cut the supply power to the motor during programming). The command END defines the end of the program. All commands given after the END will be immediately carried out by the motor.	PROGSEQ LA1000 NP1000 M END
GPROGSEQ	Get Program Sequence	Calls up the program sequence at the host PC.	GPROGSEQ ➔ <Program>
ENPROG *)	Enable Program	Starts the program. This command can also be saved with the EEPSAV command and the program will then run directly after turning the power to the motor on.	ENPROG
DIPROG *)	Disable Program	Deactivates the program.	DIPROG

Additional commands for programming sequences can be viewed in the chapter "Saving and Running Programs".

Appendix

More Request Commands

Command	Function	Description	Example
POS	Get Actual Position	Calls up the actual position	POS ➔ 500000
GN	Get N	Calls up the actual speed	GN ➔ 4000
GCL	Get Current Limit	Calls up the actual current limit value	GCL ➔ 2800
GRC	Get Real Current	Calls up the actual current	GRC ➔ 2000
TEM	Get Temperature	Calls up the actual temperature of the motor casing.	TEM ➔ 35
VER	Get Version	Calls up the version the software in use.	VER ➔ ...305008..
NE	Notify Error	In the event of an error the host will be signalled. NE1: activate. (Sends an "r" to the host terminal) NE0: deactivate	NE1 asynch ➔ r
GST	Get Status	Calls up the actual status (7 Bits) From left to right: Bit 0: 1 ... Position controller active 0 ... Velocity controller active Bit 1: 1 ... Velocity is analog or PWM 0 ... Velocity given at the RS-232 Bit 2: 1 ... Velocity is PWM (Bit1 = 1) 0 ... Velocity is analog (Bit1 = 1) Bit 3: 1 ... Drive enabled 0 ... Drive disabled Bit 4: 1 ... Command position has been reached 0 ... Command position has not yet been reached. Bit 5: 1 ... Positive edge at limit switch is active 0 ... Negative edge at limit switch is active Bit 6: 1 ... Limit switch set to high level 0 ... Limit switch set to low level	GST ➔ 0101011
GFS	Get Fault Status	Calls up the fault status. (4 Bits) 0 ... No Error 1 ... Error From left to right: Bit 0: Overtemperature Bit 1: Current Limiting Bit 2: Undervoltage Bit 3: Overvoltage	GFS ➔ 0000
GAST	Get Actual Status	Calls up the actual status at the host RS-232 (4 Bits) From left to right: Bit 0: 1 ... Limit switch 2 at high level 0 ... Limit switch 2 at low level Bit 1: 1 ... Limit switch 3 at high level 0 ... Limit switch 3 at low level Bit 2: 1 ... Direction of rotation right by positive values 0 ... Direction of rotation left by positive values Bit 3: 1 ... Power On Homing Sequence is running 0 ... Power On Homing Sequence has ended	GAST ➔ 1100

Continuation: More Request Commands

Command	Function	Description	Example
GSCS	Get Special Configuration Set	Calls up configuration at the host RS-232 (8 Bits) From left to right: Bit 0: 1 ... Power On Homing Sequence is active 0 ... Power On Homing Sequence is inactive Bit 1: 1 ... Fault Pin is an input 0 ... Fault Pin is an output Bit 2: 1 ... Pulse output at fault pin (Bit1 = 0) 0 ... Error signal at fault pin (Bit1 = 0) Bit 3: 1 ... Bit1=1: Fault pin is a direction of rotation input Bit1=0: Fault pin is a digital output 0 ... Bit1=1: Fault pin is limiter switch input 2 Bit1=0: Fault pin is not a digital output Bit 4: 1 ... Rising edge at limit switch 2 is active 0 ... Falling edge at limit switch 2 is active Bit 5: 1 ... Rising edge at limit switch 3 is active 0 ... Falling edge at limit switch 3 is active Bit 6: 1 ... Program sequence is active 0 ... Program sequence is inactive Bit 7: 1 ... Automatic answering is active 0 ... Automatic answering is inactive	GSCS → 00000001
GES	Get Enhanced Status	Send 5 status Bits to the RS-232 port. From left to right: Bit 0: 0 Bit 1: 0 Bit 2: 1... Analog command current active 0... No analog command current Bit 3: 1... Position limits in all modes active 0... Position limits only in positioning mode active Bit 4: 1... Deviation error is given 0... No deviation error is given	GES → 00001

Example Configurations and Programs

The following examples are programmed from the basis of the factory settings.

Velocity Control: Command value received as a PWM signal at the analog input.

Goals:

- Velocity control with a PWM signal at the analog input.
- Limit acceleration to 500 Rev/s^2
- Set scaling so that a maximum speed of 5000 rpm is possible.
- Set the parameters, proportional term to 30 and the integral term to 15

ASCII Command form:
SOR2
AC500
SP5000
POR30
I15
EEPSAV ➔ Saves configuration in EEPROM

Velocity Control: Command value received at the RS-232 port

Goals:

- Velocity control at the RS-232 port.
- Two active limiter switches (high level active) cannot be passed.
- Set peak current to 3 A

ASCII Command form:
SOR0
REFIN ➔ Defines the fault pin as a limiter switch input

HP3 ➔ Both limiter switches are set to high level

HB3 ➔ Both limiter switches are set to Hard Blocking

HD1 ➔ Blocks right direction of rotation at the analog limiter switch and left direction of rotation at the Fault Pin limiter switch.

LPC3000

LCC1000

V0 ➔ Switches to velocity control mode and stops.

EEPSAV ➔ Saves configuration to the EEPROM

Appendix

Position Control

Goals:

- Position control
- Limit acceleration to 300 Rev/s^2
- Reference point at falling edge
- Controller Parameters: Proportional 25 and integral 8

ASCII Command form:

AC300

HP0

POR25

I8

SOR0 → Switches to receive commands at the RS-232

LA0 → Loads new position

M → Activates positioning mode and positions

EEPSAV → Saves configuration to the EEPROM

Note: The positioning should always take place at the end so that it runs with the desired parameters.

Stepper Motor Mode

Goals:

- Operation as a stepper motor
- Step Width: 3
- Step Number per Revolution: 557
- Active Power On Homing Sequence with the fault pin as reference input
- Reference point at falling edge
- Homing sequence runs with direction of rotation left and 300 rpm

ASCII Commands:

STEPMOD

STW3

STN557

REFIN → Programs fault pin as the reference input

HA2 → When edge triggers at limiter switch 2 set the position to 0

HL2 → When edge triggers at limiter switch 2 the motor stops

CAHOSEQ → Programs the homing sequence in the intermediate memory

POHOSEQ1 → Activate Power On Homing sequence

HP0 → Sets falling edge at all reference inputs

HOSP-300 → The negative velocity value indicates direction of rotation left

EEPSAV → Saves the configuration to the EEPROM

Note: The RS-232 is no longer necessary after the configuration has been saved to the EEPROM

Program Sequence: Calling up Various Positioning Routines at the RS-232

This program makes it possible to call up various programs at the RS-232 port.

- Homing: The motor first runs to a limiter switch then to the Hall index zero. This type of homing sequence is repeatable with a high degree of positioning accuracy.
- Sequence 1: Runs to 0 position and stops.
- Sequence 2: The motor attempts to reach a position with very little current limiting. It can be that the motor doesn't reach the given position due to some blocking obstacle. After 5 seconds the motor should be stopped. (Further evaluation then takes place in the external computer)
- Sequence 3: The motor runs the following sequence 1000 times: 10 revolutions forward, 1 second pause, 5 revolutions backward, than 0.5 second pause.

Configuration:

SOR0	➔ Switches to receive command speed at the RS-232
LR0	➔ Motion stop
M	➔ Switches to position control (no motion)
HA1	
HL1	➔ Analog input is limiter switch
HN1	
CAHOSEQ	➔ Saves homing sequence in the intermediate memory
HOSP200	➔ Loads homing speed
HP1	➔ Rising edge is trigger edge
ENPROG	➔ Run program immediately following power on
ANSW0	➔ No asynchronus answering
EEPSAV	➔ Save configuration

Program:

A1	
JMP1	➔ Continuous loop
A2	➔ Jump-in address for the homing sequence
GOHOSEQ	➔ Run homing sequence
GOHIX	➔ Run Hall index homing sequence
JMP1	➔ Jump to continuous loop
A3	➔ Jump-in address for sequence 1
LA0	➔ Sets command position to 0
NP	➔ Notify when command position is achieved... Sequence suspended until the command position has been reached.
M	➔ Start positioning
JMP1	➔ Jump to Continuous loop

```
A4           ➔ Jump-in address for sequence 2
LPC500       ➔ Set peak current to 500mA...
               (continuous current ≤ peak
               current)
LA1000000
NP
TIMEOUT500   ➔ After 5 seconds delay the
               program will start even if the
               given position has not yet been
               reached.
M            ➔
V0           ➔
LR0          ➔
M            ➔ Stops motor
JMP1         ➔ Jumps to continuous loop

A5           ➔ Jump-in address for sequence 3
SETA1000     ➔ Define variable
A6           ➔
LR10000
NP
M
DELAY100
LR-5000
NP
M
DELAY50
DAJNZ6       ➔ Repeat the loop 1000 times
JMP1         ➔ Jump to continuous loop
```

Commentary:

- The homing sequence is called up by sending the JMP2 command from the RS-232. The other routines are called up in a similar manner.
- The NP command without an arguement makes sure that the sequence stops at the M command until the command position has been achieved.

Program: Sequence Controlled by the Digital Input

Goals:

- After power on the motor should run first to the limiter switch and then the Hall index zero.
- At a rising edge at the digital input (fault pin has been reprogrammed) the motor will make 5 forward revolutions. (Then stop if the logic level is then low).
- If the logic level is still high after 5 revolutions then the motor should run to position 0.

Configuration:

```
SOR0         ➔ Switches to speed control over
               the RS-232
LR0          ➔ No motion
M            ➔ Switches to position control
REFIN        ➔ Reprograms fault output as an
               input
HA1
HL1          ➔ Analog input is limiter switch
HN1
CAHOSEQ      ➔ Saves homing sequence to the
               intermediate memory
POHOSEQ1     ➔ Activates power on homing
               sequence
```

Appendix

HOSP-200	→ Loads homing speed (backward)
HP1	→ Rising edge will be registered at the limit switch
ENPROG	→ Program will activate after power on
ANSW0	→ No asynchronus answering
EEPSAV	→ Saves configuration

Program:

GOHIX	→ Homing with the Hall index zero
A1	
HP3	→ Logic level high is registered at input 2 (Fault Pin Input)
A2	
JPF2	→ Continuous loop until the logic level is low at input 2
HP1	→ Logic level low is registered at input 2 (Fault Pin Input)
A3	
JPF3	→ Continuous loop until logic level is high at input 2 ... evaluation of the positive edge
LR5000	
NP	
M	→ 5 revolutions forward
DELAY50	→ 0.5 second delay then evaluate input 2
JPF1	→ When low logic is registered at input 2, jump to the beginning of the program
LA0	
NP0	
M	→ Run to position 0 if high logic level at input 2
JMP1	→ Jump to the beginning

Commentary:

- After programming is complete the RS-232 connection is no longer necessary.
- The program is started with a short pulse at the digital input and interrupted with a long pulse.

Appendix

Factory Configuration

The commands in the factory configuration are as follows:

EN	→ Drive is active
V0	→ Command speed 0
SOR1	→ Source for velocity is the analog input
ADR	→ Positive voltage means direction of rotation right
HP7	→ Rising edge registered at all limit switches
ERROUT	→ Fault Pin is fault output
CONTMOD	→ Normal mode (speed control with the Hall Sensors)
HALLSPEED	→ Hall Sensors register speed
HB0	→ No hard blocking
HD0	→ Left direction of rotation is blocked at all limit switches
HA0	→
HL0	→
HN0	→
CAHOSEQ	→ Deavctivate homing sequence
POHOSEQ0	→ Power on homing sequence is deactivated
HOSP100	→ Homing speed is 100 with direction of rotation right
ANSW1	→ Asynchronus answering is active
DIPROG	→ Program will not start when power is turned on
BAUD9600	→ Baud rate is 9600
NODEADR0	→ Sets node address to 0
MOTTYP5	→ Sets to the 2444S024B K1155 motor
RN	→ Represents the following commands: LPC5000 LCC1370 AC30000 I15 POR4 SP10000 MV0 MAV25 LL600000000 LL-600000000 DCE200 DEV30000 LPN16 STW1 STN1000 ENCRES2048

Commentary:

The command FCONFIG will restore the factory settings with the exception of the MOTTYP8 (also other parameter values!)

Notes

Notes

FAULHABER

From Software Version 305010A and upwards

MA05002, english, 1. Edition, 13. 06. 01

DR. FRITZ FAULHABER
GMBH & CO. KG
Antriebssysteme

Daimlerstraße 23
71101 Schönaich · Germany
Tel. +49(0)70 31/638-0
Fax +49(0)70 31/638-100
info@faulhaber.de
www.faulhaber.de

REGELMANN