WAGO-I/O-SYSTEM 750

Fieldbus Independent I/O Modules

Steppercontroller 750-671

Manual

Version 1.0.2

INNOVATIVE CONNECTIONS

WAGO Kontakttechnik GmbH & Co. KG
Hansastraße 27
D-32423 Minden

Phone: +49 (0) 571/8 87 – 0
Fax: +49 (0) 571/8 87 – 1 69

E-Mail: info@wago.com

Web: http://www.wago.com

Technical Support
Phone: +49 (0) 571/8 87 – 5 55
Fax: +49 (0) 571/8 87 – 85 55

E-Mail: support@wago.com

Every conceivable measure has been taken to ensure the correctness and completeness of this documentation. However, as errors can never be fully excluded, we would appreciate any information or ideas at any time.

E-Mail: documentation@wago.com

We wish to point out that the software and hardware terms as well as the trademarks of companies used and/or mentioned in the present manual are generally trademark or patent protected.

CONTENT

1 Important Notes

This section includes an overall summary of the most important safety requirements and notes that are mentioned in each individual section. To protect your health and prevent damage to devices as well, it is imperative to read and carefully follow the safety guidelines.

1.1 Legal Bases

1.1.1 Copyright

WAGO Kontakttechnik GmbH & Co. KG reserves the right to provide for any alterations or modifications that serve to increase the efficiency of technical progress. WAGO Kontakttechnik GmbH & Co. KG owns all rights arising from the granting of patents or from the legal protection of utility patents. Third-party products are always mentioned without any reference to patent rights. Thus, the existence of such rights cannot be excluded.

1.1.2 Personnel Qualifications

The use of the product described in this Manual requires special personnel qualifications, as shown in the following table:

Activity	Electrical specialist	Instructed personnel*)	Specialists**) having qualifications in PLC programming
Assembly	X	X	
Commissioning	X		X
Programming			X
Maintenance	X	X	
Troubleshooting	X		
Disassembly	X	X	

*) Instructed persons have been trained by qualified personnel or electrical specialists.

**) A specialist is a person, who – thanks to technical training – has the qualification, knowledge and expertise to meet the required specifications of this work and to identify any potential hazardous situation in the above listed fields of activity.

All responsible persons have to familiarize themselves with the underlying legal standards to be applied. WAGO Kontakttechnik GmbH & Co. KG does not assume any liability whatsoever resulting from improper handling and damage incurred to both WAGO´s own and third-party products by disregarding detailed information in this Manual.

1.1.3 Use of the 750 Series in Compliance with Underlying Provisions

Couplers, controllers and I/O modules found in the modular WAGO-I/O-SYSTEM 750 receive digital and analog signals from sensors and transmit them to the actuators or higher-level control systems. Using programmable controllers, the signals can also be (pre-)processed.

The components have been developed for use in an environment that meets the IP20 protection class criteria. Protection against finger injury and solid impurities up to 12.5 mm diameter is assured; protection against water damage is not ensured. Unless otherwise specified, operation of the components in wet and dusty environments is prohibited.

1.1.4 Technical Condition of Specified Devices

The components to be supplied Ex Works, are equipped with hardware and software configurations, which meet the individual application requirements. Changes in hardware, software and firmware are permitted exclusively within the framework of the various alternatives that are documented in the specific manuals. WAGO Kontakttechnik GmbH & Co. KG will be exempted from any liability in case of changes in hardware or software as well as to non-compliant usage of components.

Please send your request for modified and new hardware or software configurations directly to WAGO Kontakttechnik GmbH & Co. KG.

1.2 Standards and Guidelines for Operating the 750 Series

Please adhere to the standards and guidelines required for the use of your system:

- The data and power lines shall be connected and installed in compliance with the standards required to avoid failures on your system and to substantially minimize any imminently hazardous situations resulting in personal injury.
- For assembly, start-up, maintenance and troubleshooting, adhere to the specific accident prevention provisions which apply to your system (e.g. BGV A 3, "Electrical Installations and Equipment").
- Emergency stop functions and equipment shall not be made ineffective. See relevant standards (e.g. DIN EN 418).
- The equipment of your system shall be conform to EMC guidelines so that any electromagnetic interferences will be eliminated.
- Operating 750 Series components in home applications without further measures is permitted only if they meet the emission limits (emissions of interference) in compliance with EN 61000-6-3. You will find the detailed information in section "WAGO-I/O-SYSTEM 750" → "System Description" → "Technical Data".
- Please observe the safety precautions against electrostatic discharge in accordance with DIN EN 61340-5-1/-3. When handling the modules, please ensure that environmental factors (persons, working place and packaging) are well grounded.
- The valid standards and guidelines applicable for the installation of switch cabinets shall be adhered to.

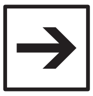
1.3 Symbols

Danger
Always observe this information to protect persons from injury.

Warning
Always observe this information to prevent damage to the device.

Attention
Marginal conditions that must always be observed to ensure smooth and efficient operation.

ESD (Electrostatic Discharge)
Warning of damage to the components through electrostatic discharge. Observe the precautionary measure for handling components at risk of electrostatic discharge.



Note
Make important notes that are to be complied with so that a trouble-free and efficient device operation can be guaranteed.

Additional Information
References to additional literature, manuals, data sheets and internet pages.

1.4 Safety Information

When connecting the device to your installation and during operation, the following safety notes must be observed:

Danger
The WAGO-I/O-SYSTEM 750 and its components are an open system. It must only be assembled in housings, cabinets or in electrical operation rooms. Access is only permitted via a key or tool to authorized qualified personnel.

Danger
All power sources to the device must always be switched off before carrying out any installation, repair or maintenance work.

Warning
Replace defective or damaged device/module (e.g. in the event of deformed contacts), as the functionality of field bus station in question can no longer be ensured on a long-term basis.

Warning
The components are not resistant against materials having seeping and insulating properties. Belonging to this group of materials is: e.g. aerosols, silicones, triglycerides (found in some hand creams). If it cannot be ruled out that these materials appear in the component environment, then the components must be installed in an enclosure that is resistant against the above mentioned materials. Clean tools and materials are generally required to operate the device/module.

Warning
Soiled contacts must be cleaned using oil-free compressed air or with ethyl alcohol and leather cloths.

Warning
Do not use contact sprays, which could possibly impair the functioning of the contact area.

Warning
Avoid reverse polarity of data and power lines, as this may damage the devices.

ESD (Electrostatic Discharge)
The devices are equipped with electronic components that may be destroyed by electrostatic discharge when touched.

Warning
For components with ETHERNET/RJ-45 connectors:
Only for use in LAN, not for connection to telecommunication circuits.

1.5 Font Conventions

italic	Names of paths and data files are marked in italic-type. e.g.: *C:\Programs\WAGO-IO-CHECK*
italic	Menu items are marked in italic-type, bold letters. e.g.: ***Save***
\	A backslash between two names characterizes the selection of a menu point from a menu. e.g.: ***File \ New***
END	Pushbuttons are marked as bold with small capitals e.g.: **ENTER**
< >	Keys are marked bold within angle brackets e.g.: **<F5>**
`Courier`	The print font for program codes is Courier. e.g.: `END_VAR`

1.6 Number Notation

Number code	Example	Note
Decimal	100	Normal notation
Hexadecimal	0x64	C notation
Binary	'100' '0110.0100'	In quotation marks, nibble separated with dots (.)

1.7 Scope

This manual describes the Special Module 750-671 Steppercontroller of the modular WAGO-I/O-SYSTEM 750.

Handling, assembly and start-up are described in the manual of the Fieldbus Coupler. Therefore this documentation is valid only in the connection with the appropriate manual.

2 I/O Modules

2.1 Special Modules

2.1.1 General Description

2.1.1.1 Safety Information

Observe the following information and safety notices to prevent injury and/or equipment.

Danger
Take appropriate measures, such as cordoning off appropriate areas with screens/enclosures, to prevent bodily contact with the system's moving parts.

Danger
Enact and install an EMERGENCY OFF procedure and system that adheres to locally valid regulations and applicable engineering practices.

Notice
Install appropriate hardware limit switches that can directly disengage power to the system if a restricted area of movement has been breached.

Notice
Install appropriate equipment to protect motors and power electronics, such as motor circuit breakers or fuses.

2.1.1.2 Structure of Positioning Controller

The following figure illustrates the structure of a typical positioning controller, along with its basic elements:

- Control section,
- Power section,
- Drive section,
- Mechanical section.

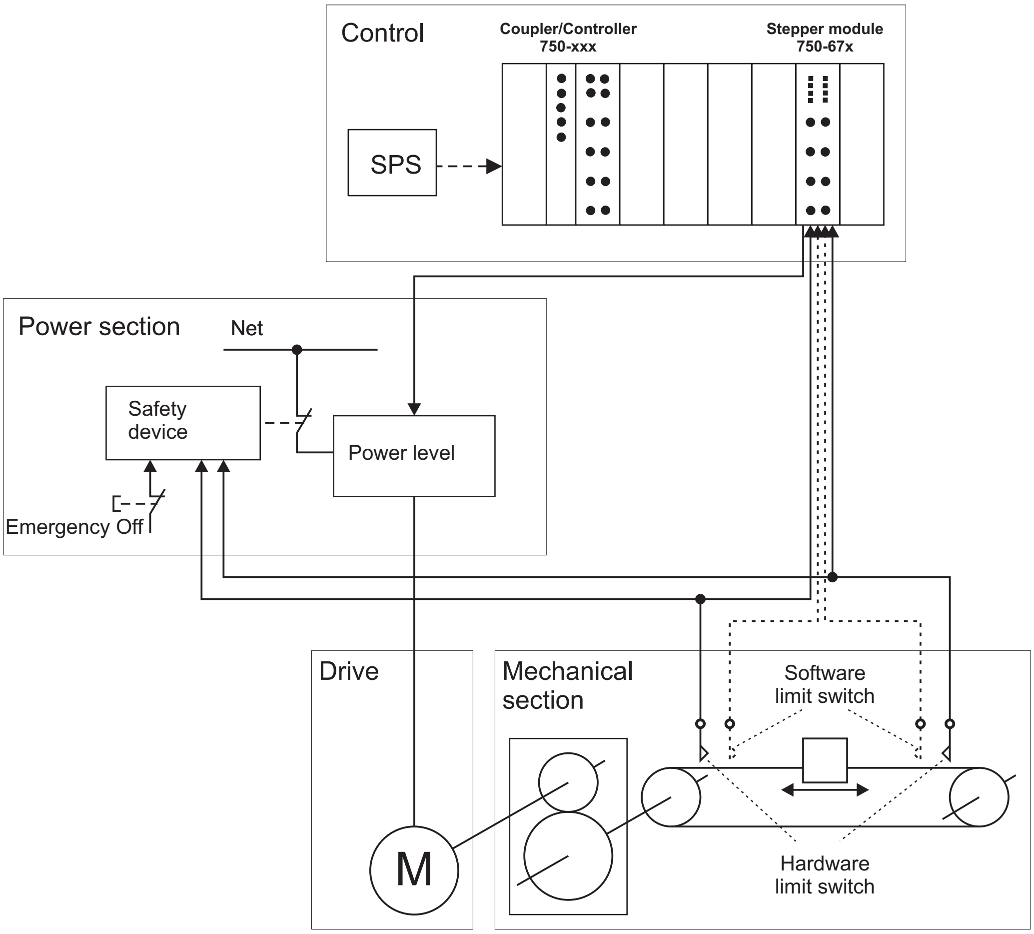


Fig. 2.1.1-1: Structure of position control system g067x00e

2.1.1.2.1 Control Section

The control section consists of a PLC for process control and stepper module 750-67x for positioning, FM and PWM functions.

2.1.1.2.2 Power Section

The power stage generates drive currents from the pulses for the specific motor. Any type of output stage equipped with a pulse direction or incremental encoder interface can be used with the 750-670 stepper module This also allows output stages for 3- or 5-pole stepper motors, DC or AC servo motors to be used. Stepper modules 750-671, -672 and -673 are equipped with an integrated output stage for regulating 2-phase stepper motors.

2.1.1.2.3 Drive

Stepper motors are simple and economical drives that execute highly precise tasks for a wide range of applications.

The shaft of a stepper motor rotates by a defined angle at each pulse; a rapid succession of pulses transforms the stepping motion into a continuous turning motion. The stepper motor's natural resonance is suppressed largely by high-revolution microstepping, which produces extremely smooth operation. This is characteristic of WAGO modules 750-671, -672 and –673, which feature 64-fold microstepping.

The figures below illustrate possible types of connections for stepper motors:

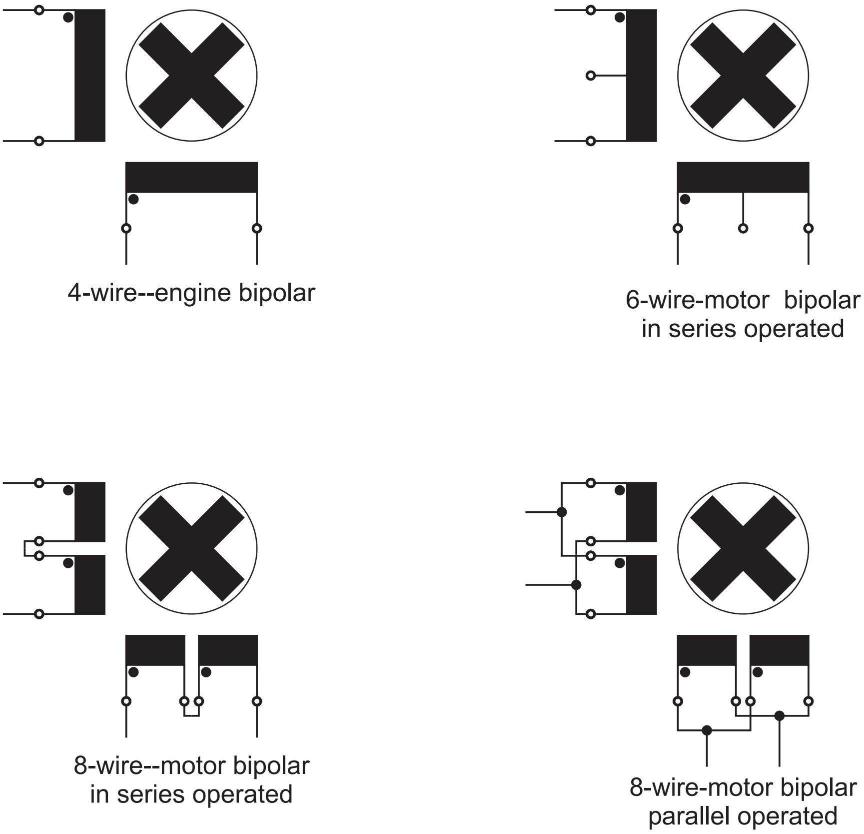


Fig. 2.1.1-2: Types of connections for stepper motors g067x02e

The following must be considered when selecting an appropriate motor:

- type of connection and number of phases
- required torque progression over speed
- required motor current
- winding resistance
- motor inductance.

The application dictates the torque progression and speed; experience has shown that a torque margin of approx. 25 %, depending on the mechanical system properties, is useful. This should be considered when accounting for any dynamic effects (resonance in mechanical systems).

The positioning process sequences also determine both the average and peak power supplied to the motor; special attention must be given to the total power loss and motor temperature.

Depending on the motor model and design, a corresponding current must be present in order to be transferred from the output stage into the motor. The required voltage depends on the winding resistance, motor inductance and speed (anti-EMC). It may be necessary to have considerably higher voltage levels for the specific current level, particularly at high speeds, than that specified by the motor data. The manufacturer-provided motor data is based on motor standstill (ohmic winding resistance). The power output stages for stepper modules 750-671, -672 and –673 are equipped with power control systems. For example, it is possible to run 12 V motors with 24 V supply systems as long as the current, power loss and motor temperature remain within acceptable limits. Consult the motor manufacturer with any questions/concerns.

2.1.1.2.4 Mechanical Section

Motor data can be calculated based on the requirements for the load to be moved, and any additional bearings, transmissions, deflection systems, damping elements, etc. that may be required. Important parameters here are:

- moment of inertia
- starting torque
- holding torque
- torque at the maximum required speed,
- cycle times for positioning
- requisite acceleration
- required torque (where applicable) when passing through mechanical resonance fields - particularly when mechanical components such as long drive belts, spring elements or vibration buffers (couplings) are used.

Please note that there must be no step losses if the required mechanical torque does not exceed the torque supplied by the motor (taking inertia into account).

2.1.1.3 Positioning

A distinction is made here between absolute and relative positioning. Additionally, a difference is also made between a reference run and the Jog mode.

2.1.1.3.1 Absolute Positioning

Positioning from the absolute position X to absolute position Y.

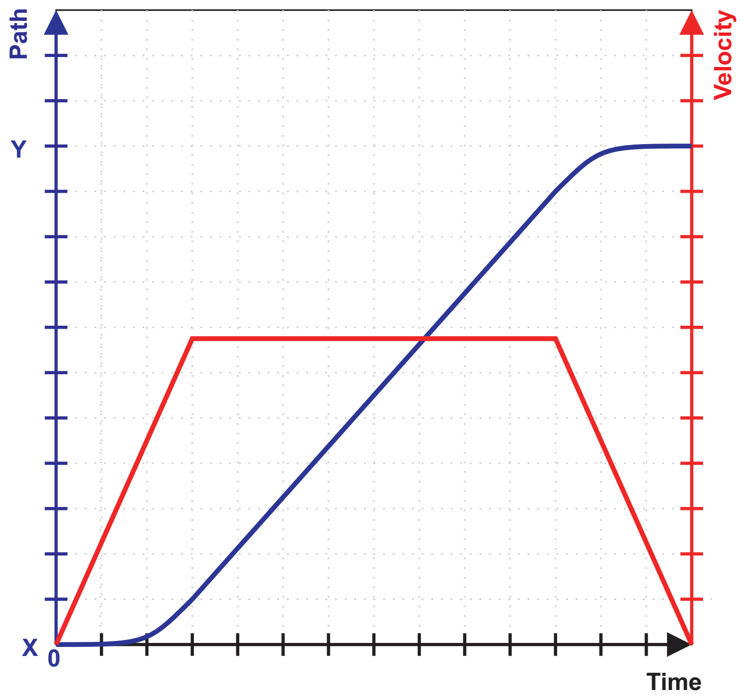


Fig. 2.1.1-3: Absolute positioning g067x03e

Potential applications:

- Positioning shafts
- Transfer carriages
- Pick & Place

2.1.1.3.2 Relative Positioning

Positioning from absolute position X to absolute position Y by the difference x; it is also possible as a command during positioning (on the fly).

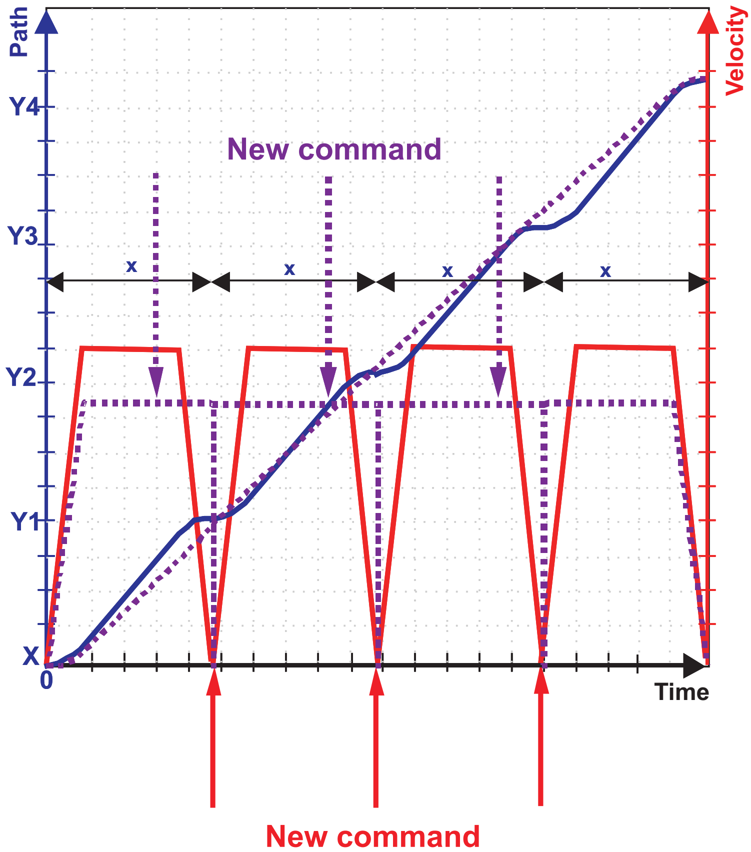


Fig. 2.1.1-4: Relative positioning g067x02e

Potential applications:

- Incremental dimensions
- Variable reference points

2.1.1.3.3 On-the-Fly Positioning

Termination of ongoing positioning (such as Move to Y) and execution of the new positioning command (Move to Y-n).

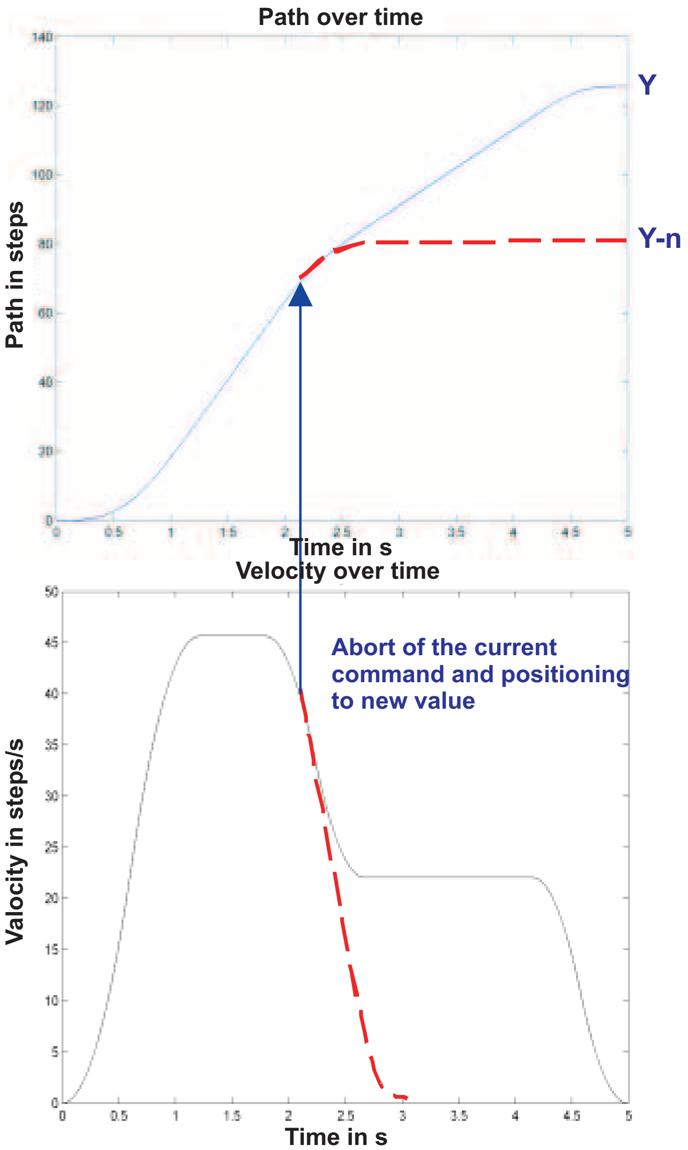


Fig. 2.1.1-5: On-the-fly positioning g067x05e

Potential applications:

- Event-dependent changing of target position
- Collision avoidance
- Process optimization

2.1.1.3.4 Referencing

Referencing is the setup of a measuring system. A distinction is drawn here between referencing to a limit switch and referencing to a special reference switch. A high degree of reproducible accuracy is essential for referencing. Referencing should always be performed from the same end.

Referencing involves searching for the reference switch at the set setup speed and then moving toward that point from the correct end of any position with the movement range.

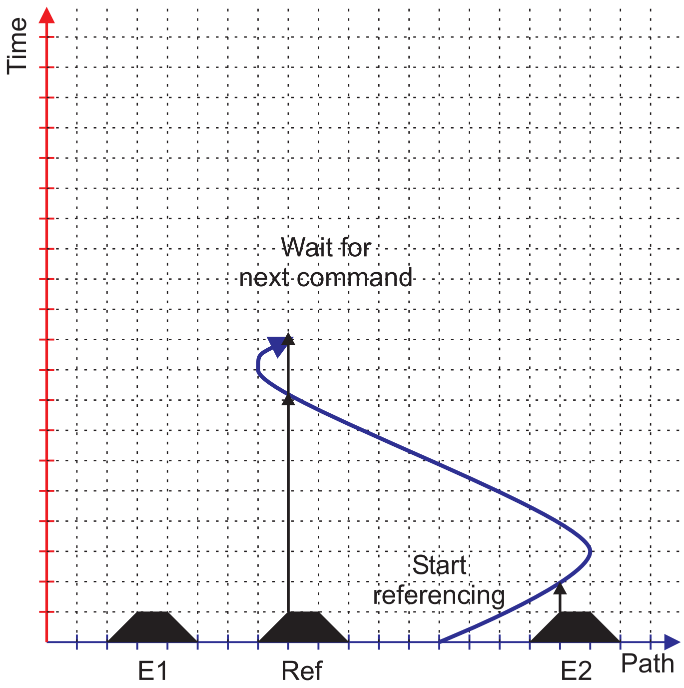


Fig. 2.1.1-6: Referencing g067x09e

The reference value (usually 0) is accepted at the corresponding edge.

2.1.1.3.5 Jogging Mode

The drive is run at the setup speed via defined input, or a control bit, as long as either the input is active or the bit is set. A time limit can be activated for the moving process.

2.1.1.3.6 Rotary Axis

The value range (such as -10000 ... +10000) is converted for rotation (360°) around either a real or virtual axis for a rotary axis. Overrun is automatically taken into account; i.e., when 360° is exceeded, counting restarts at 0°. Based on the example values, after exceeding the +10000 position, the next position would be -10000.

Potential applications:

- Belt control
- Label control
- Control of rotary tables

2.1.1.3.7 Types of Acceleration

2.1.1.3.7.1 Constant Acceleration

Acceleration has a constant value during the acceleration phase.
Both the onset and completion of acceleration phase jolt the mechanical system; this phenomenon is comparable to the jolt a vehicle experiences when stepping on/stepping off the accelerator.

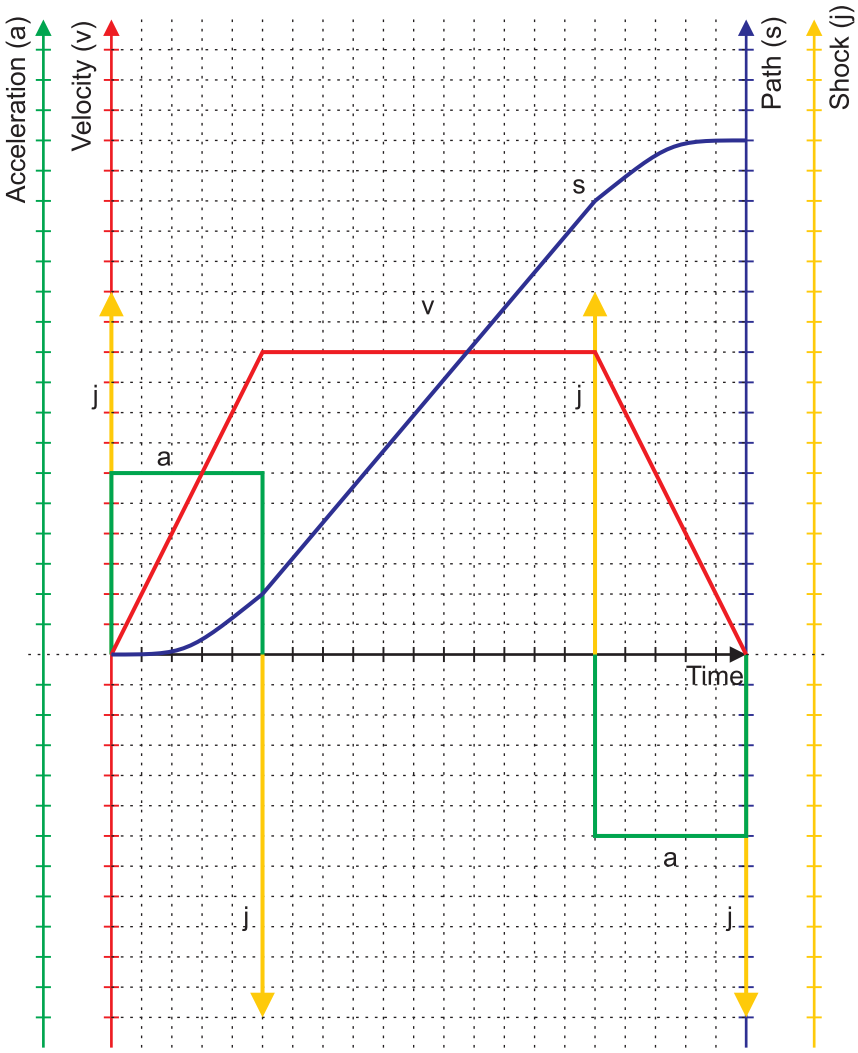


Fig. 2.1.1-7: Constant acceleration g067x06e

Potential applications:

- Peak acceleration at specified acceleration value,
- Linear path/time response.

2.1.1.3.7.2 Linear Acceleration

Acceleration increases and decreases during the acceleration phase with a linear gradient, reducing the jolt experienced by the mechanical system.

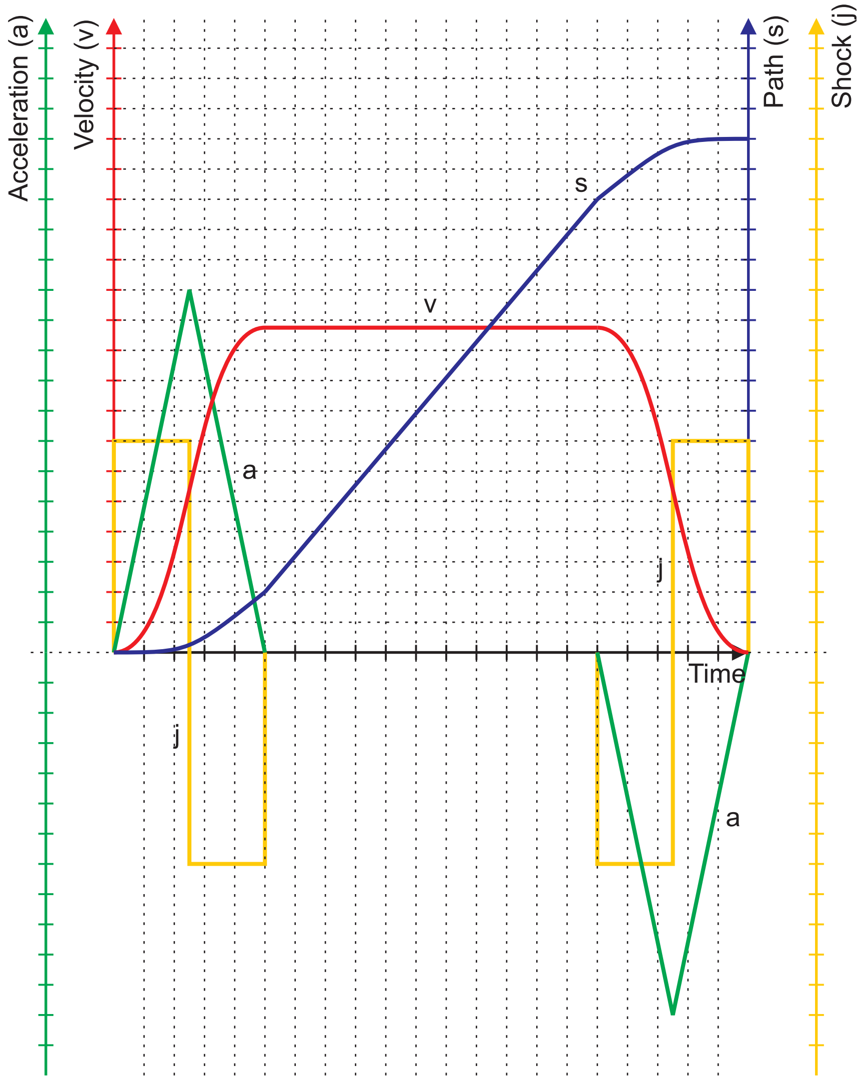


Fig. 2.1.1-8: Linear acceleration g067x07e

Potential applications:

- Soft start (jolt reduction)
- Reduction of step losses
- Linear (constant) acceleration moment
- Maximum acceleration, particularly with flexible drive systems (belts)

2.1.1.3.7.3 sin²*t Acceleration

The acceleration value progresses according to a $\sin^2 * t$ curve during the acceleration phase.
This minimizes the jolt experienced by the mechanical system, reducing any remaining harmonic waves present during linear acceleration.

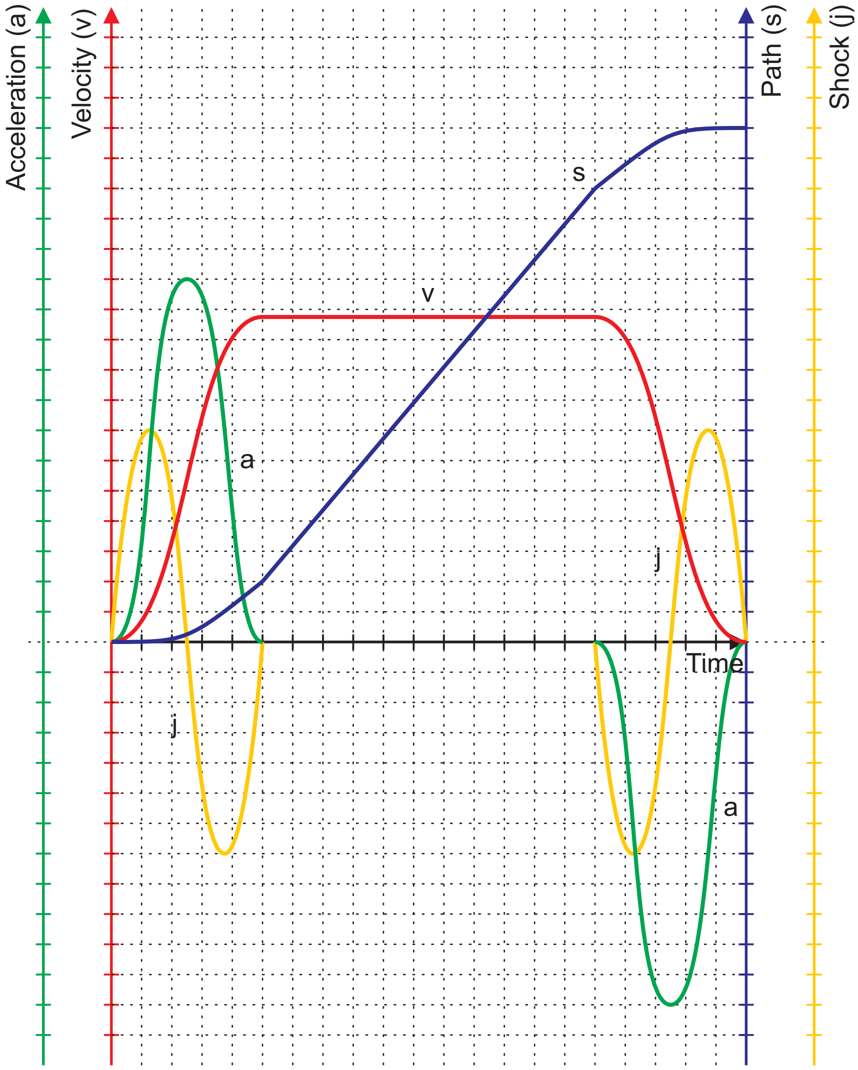


Fig. 2.1.1-9: $\sin^2 * t$ acceleration g067x08e

Potential applications:

- Soft start (jolt reduction)
- Reduction of step losses
- Maximum acceleration, particularly with flexible drive systems (belts)

2.1.1.3.7.4 Adjustable Acceleration

The acceleration and brake ramps can be adjusted individually.

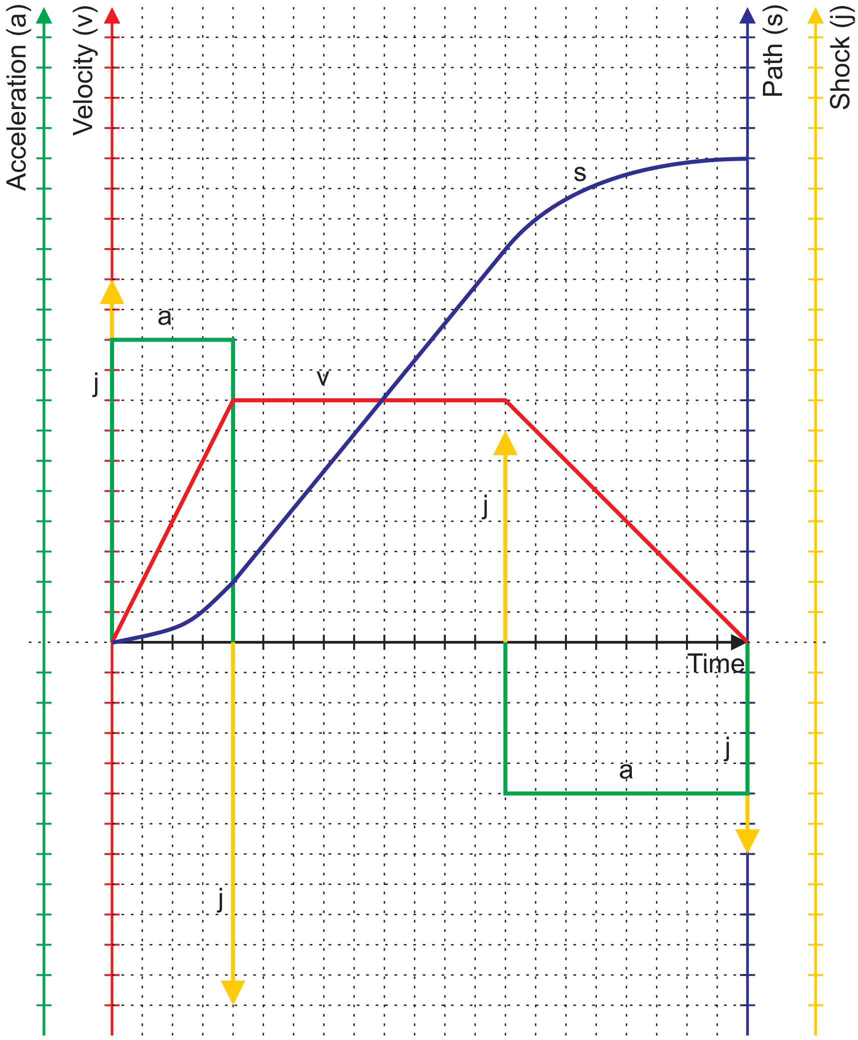


Fig. 2.1.1-10: Adjustable acceleration g067x28e

Potential applications:

- Defining if the acceleration must be done with or against additional external torques
- Asymmetric retaining forces of toolings (grippers)

The acceleration can be defined as

- Acceleration time
- Acceleration path or
- Acceleration (steps/s^2)

Potential applications:

- Cycle-time dependent applications
- Simple path calculation
- Definition of acceleration torques

2.1.1.4 Current Control

The current depends on:

- Acceleration
- Constant speed
- Delay
- Stop (holding torque)

The current is adjustable in % of the nominal value.

Values to 150 % are possible (Boost)!

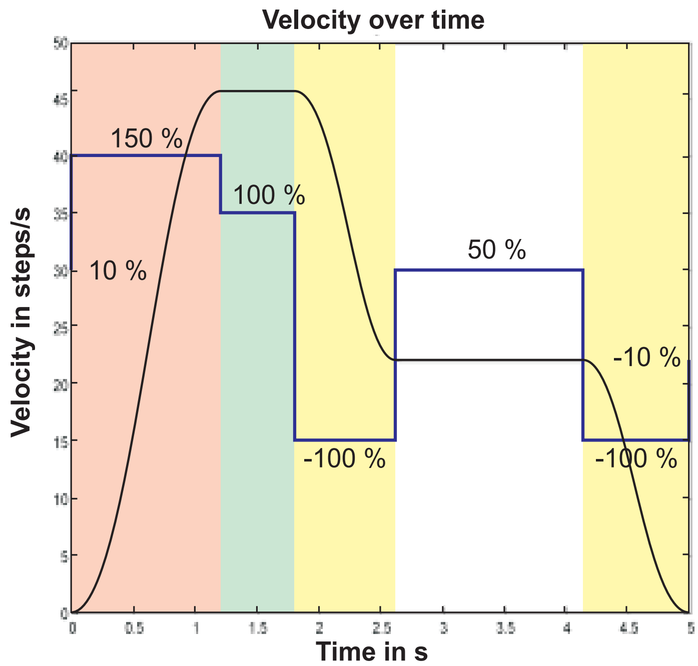


Fig. 2.1.1-11: Current Control g067x31e

Potential applications:

- Power loss limitation
- Torque control

2.1.1.5 Rotational speed

The rotational speed is regulated by speed control. Achieving a specified position is not relevant here.

Potential applications:

- Simple interfaces for ready-made application programs
- Belt drives, conveyor systems

2.1.1.6 Camshaft controller

The camshaft controller allows to set an output or bits in a position window. The position window can be defined absolutely or relatively

- Set output/bit from Xn to Yn
- Set output/bit von Xn to ΔYn

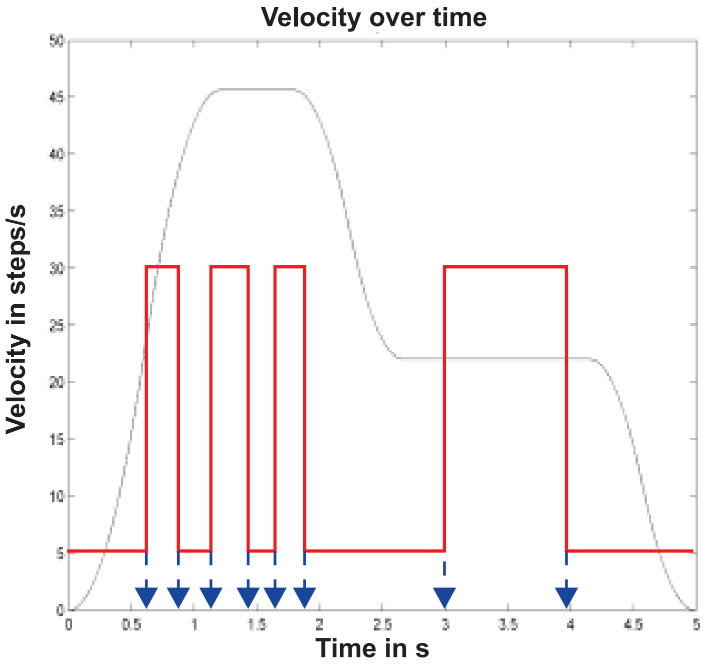


Fig. 2.1.1-12: Camshaft controller g067x32e

Potential applications:

- Setting of glue dots
- Length feeding
- Stamp positions
- Tool operation

2.1.1.7 Frequency modulation

The frequency can be set directly and be changed during operation.

The pulse duty factor is fixed at 50 %.

Maximum frequency: 500 kHz

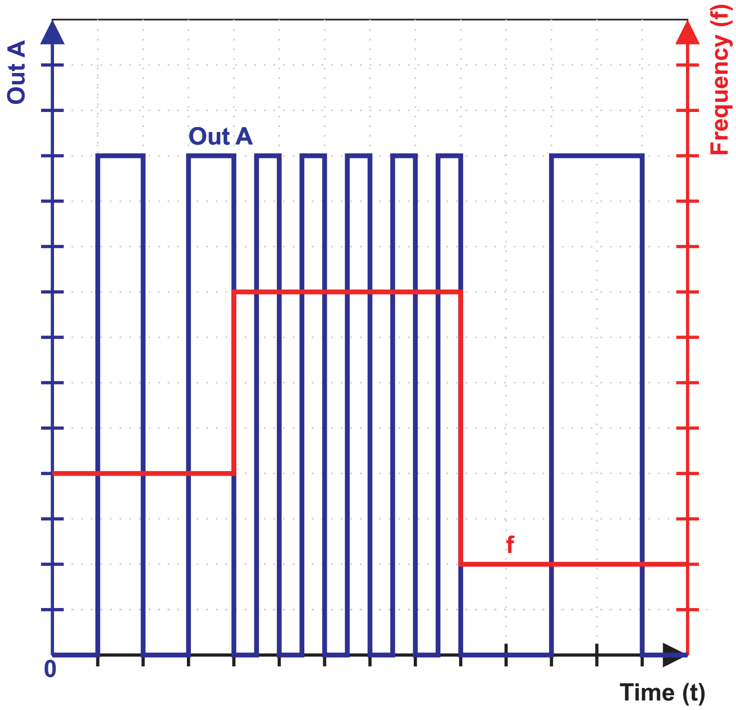


Fig. 2.1.1-13: Frequency modulation g067x34e

Potential applications:

- Rotational speed setting
- Digital set point value transmission

2.1.1.8 PWM

The frequency is preset and constant.

The pulse duty factor is variable and can be adjusted between 0 % and 100 %.

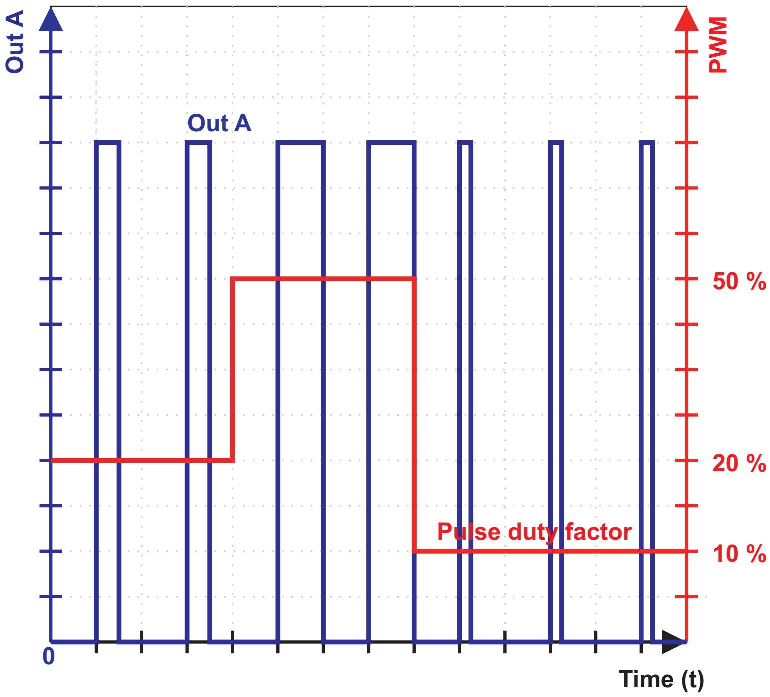


Fig. 2.1.1-14: PWM g067x36e

Potential applications:

- Temperature control
- Power control
- Lamp control
- Defining rotational speed with current control for drives

2.1.1.9 Single Shot

The pulse duration is preset.

Only one pulse is generated.

The time from the activating event up to the pulse generation is adjustable.

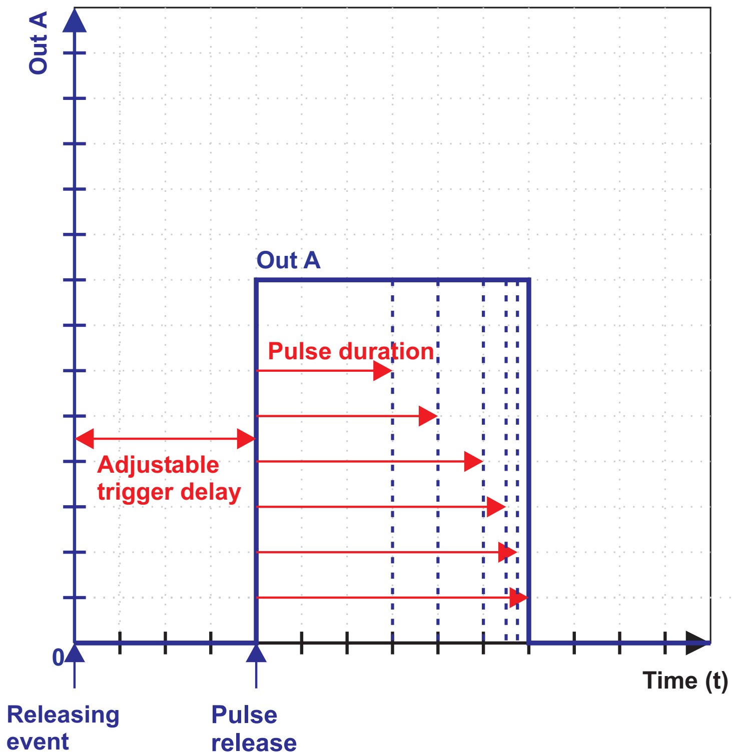


Fig. 2.1.1-15: Single Shot g067x37e

Potential applications:

- Valve opening times
- Power control
- Precise opening times

2.1.1.10 Brake Control

Brake OFF (Output=1) ΔtOff before the start of the positioning.

Brake ON (Output=0) ΔtOn before reaching the target position.

If the brake was switched on, the execution of the next positioning is delayed by ΔtOff.

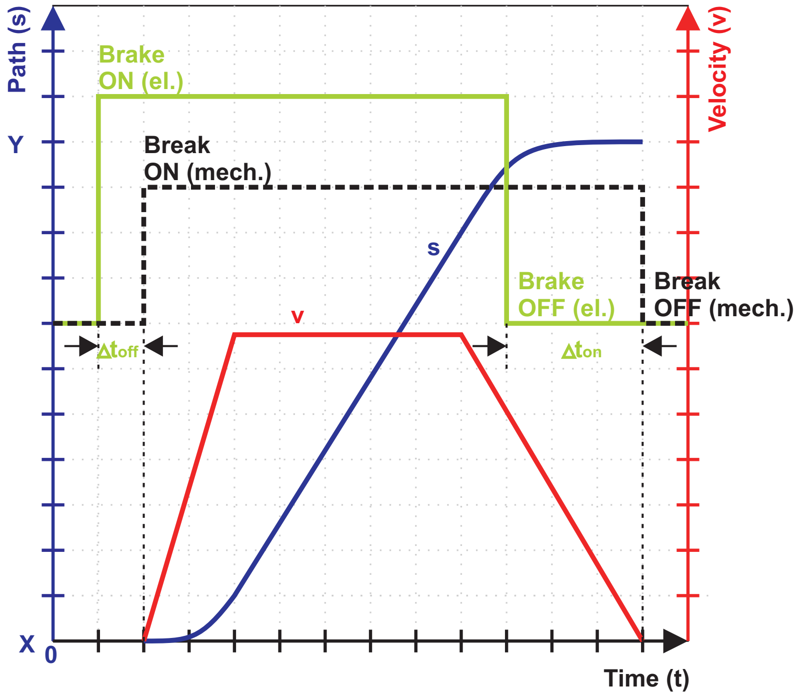


Fig. 2.1.1-16: Brake Control g067x33e

Potential applications:

- Lifting axis
- Parking brakes

2.1.1.11 Command tables

In the command tables, a complex positioning sequence can be stored and executed independently according to the appropriate sequential list of individual commands.

The command sequence can be changed or stopped depending on external or internal (PLC) events.

The possible operatings mode are:

- Cyclic (Repeat after End Of List)
- Event driven (Digital, analog, time, command)
- Direct adressable (Start command at any position of the active list)
- Jumps to other entry of list
- On the fly (Cancel actual & execute other command)

Two tables are available that can alternately be switched over: An offline table (program run) and an online table (program up-/download).

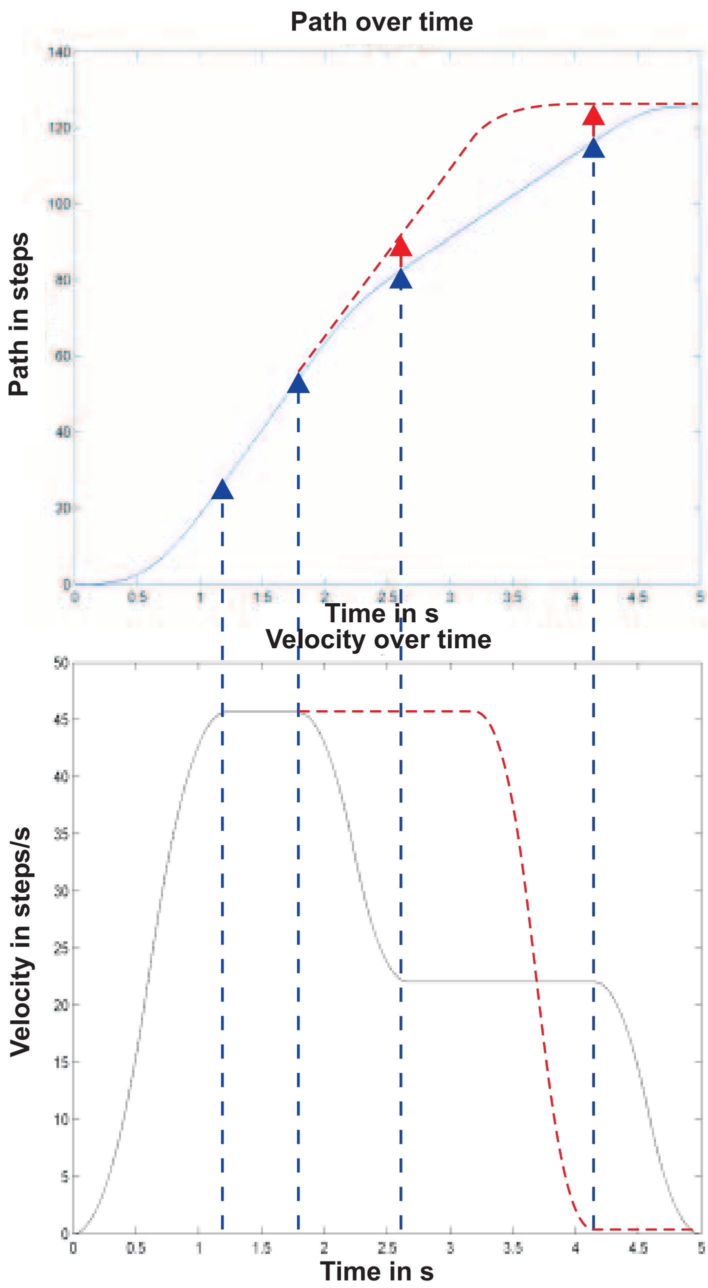


Fig. 2.1.1-17: Command tables g067x29e

Potential applications:

- Relieving the PLC
- Reduction of response times
- Encapsulating the application

2.1.2 750-671 [Steppercontroller]

2.1.2.1 View

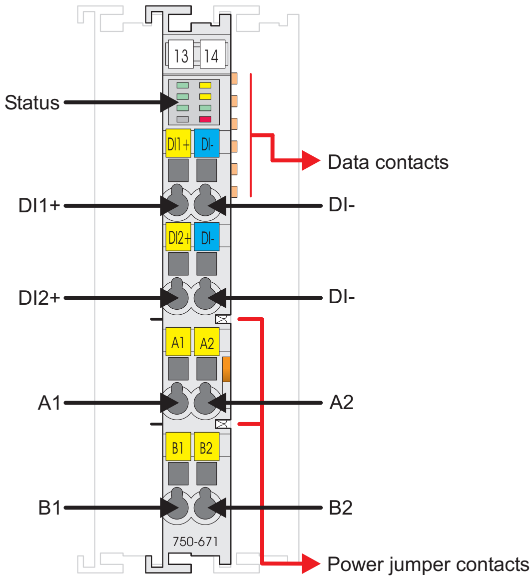


Fig. 2.1.2-1: View g067100e

2.1.2.2 Description

The stepper controller 750-671 is an intelligent controller with integrated output stage. This allows stepper motors to be connected directly. Two configurable inputs for Start/Stop, limit switches, reference cams, Jog/Tip, etc., are evaluated directly and without any further delay by the internal software.

Versatile functions, such as positioning with different acceleration slopes, command tables, auto referencing and other event-dependent properties provide this controller with a wide spectrum of possible uses

Two different applications are implemented in the stepper controller 750-671.

- Positioning,
- Speed control

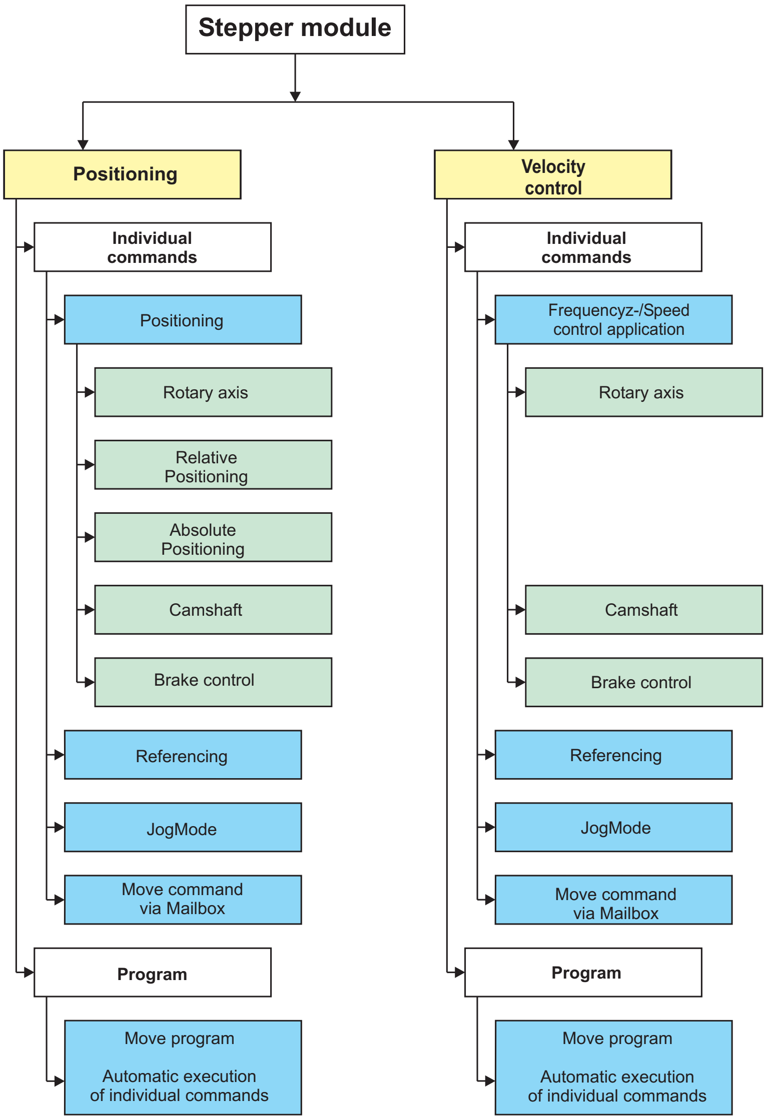


Fig. 2.1.2-2: Stepper controller applications and operating modes g067120e

There are five operating modes available in each of the applications Positioning and Velocity Control:

- Positioning and velocity definition,
- Referencing,
- Jog Mode,
- Move command via Mailbox,
- Move program.

The stepper controller function is defined by various tables, with the configuration table and the Bit I/O table playing a particularly important role.

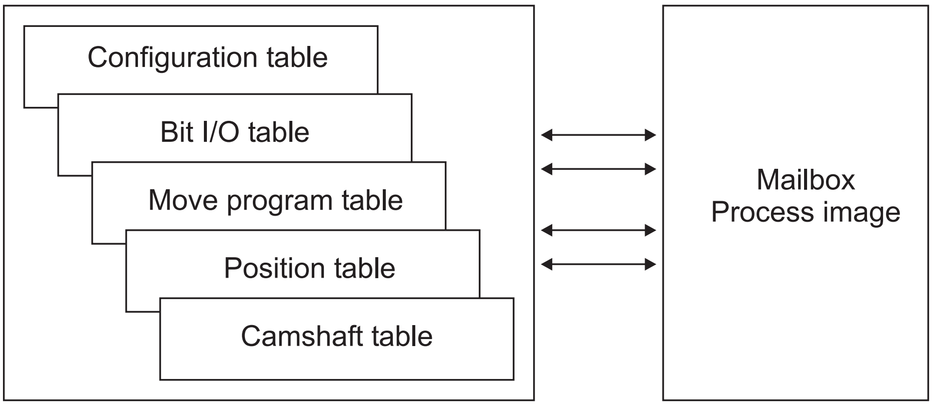


Fig. 2.1.2-3: Tables in the stepper controller g067x01e

The stepper controller is equipped with two digital 24 V inputs DI1+ and DI2, with the reference potential DI-, enabling connection of two-wire sensors or switches.

Input DI1+ is used as the enable input and input DI2+ as the reference input for standard applications; these inputs can also be assigned to dedicated applications and to other functions.

In addition, the stepper module has connections for motor windings. The windings are connected to A1–A2 and B1-B2.

The outputs are not short-circuit proof. However, the output stage is protected against overload. Reverse voltage protection of the 24 V supply is available.

The signal status for the digital inputs and the power supply status are each indicated by a dedicated green LED.
Two yellow LEDs, one green LED and one red LED indicate the active mode, status, readiness for operation and errors in the standard applications.

Field and system signals are electrically isolated.

The individual I/O modules can be arranged in any combination when configuring the fieldbus node. An arrangement in groups is not necessary.

The stepper controller receives the 24 V supply voltage for the field level via an upstream I/O module or a supply module. Power connections are made automatically from module to module via the internal power jumper contacts when snapped onto the DIN rail.

Caution
The current at the power jumper contacts may be a maximum of 10 A. When configuring the system ensure that this total current is not exceeded. If this should happen, an additional supply module has to be used.

The stepper controller can be operated at the following WAGO I/O SYSTEM 750 couplers and controllers:

Bus system	Coupler/Controller	Item No.	Hard-ware vers.	Soft-ware vers.	Max. number of modules
ETHERNET TCP/IP	Fieldbus coupler	750-341	03	06	8
		750-342	04	17	3
	Programmable fieldbus controller	750-841	03	17	16
		750-842	04	12	8
		750-843	12	01	8
		750-871	03	05	16
		750-873	02	02	16
CANopen	Fieldbus coupler	750-337	09	10	8
		750-338	02	16	8
	ECO Fieldbus coupler	750-347	01	04	1
		750-347	01	06	2
		750-348	01	04	1
		750-348	01	06	2
	Programmable Fieldbus controller	750-837	07	12	8
		750-838	02	12	8
DeviceNet	Fieldbus coupler	750-306	12	4J	8
	ECO Fieldbus coupler	750-346	02	07	2
	Programmable fieldbus controller	750-806	04	09	8
LON	Fieldbus coupler	750-319	xx	05	3
	Programmable fieldbus controller	750-819	xx	09	8
PROFIBUS	Fieldbus coupler	750-303	xx	from 08	3
		750-333	12	from 07	8
	ECO Fieldbus coupler	750-343	03	from 06	2
	Programmable fieldbus controller	750-833	16	10	8
Powerlink	Fieldbus coupler	750-350	07	01	8
BACnet	Programmable fieldbus controller	750-830	01	01	8
KNX	Programmable fieldbus controller	750-849	xx	04	16
SERCOS III	Feldbuskoppler	750-351	02	03	08

Bus system	Coupler/Controller	Item No.	Hard-ware vers.	Soft-ware vers.	Max. number of modules
WAGO-IPC	IPC	758-870/ 000-xxx	10	03	16
	IPC	758-874/ 000-xxx	10	03	16
	IPC	758-875/ 000-xxx	10	03	16
	IPC	758-876/ 000-xxx	10	03	16

Other couplers/controllers upon request

Notes
The following must be observed when using the stepper module with CANopen bus couplers 750-337, 750-338, 750-837, 750-838, 750-347 and 750-348.
The CANopen master accesses the mailbox and process data in the coupler/controller using process data objects (PDOs).
In the default configuration the stepper module data are mapped in consecutive PDOs, with each PDO able to accommodate up to eight (8) bytes of data. The 12-byte process image for the stepper module contains 2 PDOs, one with 8 and one with 4 bytes.

Problem:
The specified and actual values for the positioning data is distributed among 2 PDOs during positioning using the cycling process image, which could result in the data not being transferred consistently.

Remedy:
- For positioning via the Mailbox mode, the mailbox data are transferred consistently in PDO1 and the control bits in PDO2.
- Use of 16-bit specified/actual values or
- Omission of "on-the-fly" specified/actual values, i.e. initiation of the function only after setting of the specified values or reading out of the 24-bit actual values has been fully completed and only in the "Standstill" status.

This description is valid for the XXXX0101... and XXXX0102... hardware and software versions. The version is specified in the manufacturing number, which is part of the lateral marking on the module.

Connection Elements

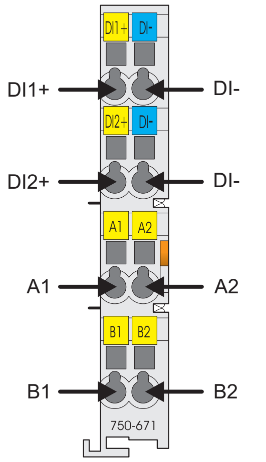


Fig. 2.1.2-4: Connecting elements g067103x

Connection	Designation	Standard configuration *)
DI1+	Digital input 1	Enable input
DI2+	Digital input 2	Referencing input
DI-	0 V for DI1+ and DI2+	Reference potential for Enable and Referencing input (electrically isolated from all other potentials)
A1	Motor winding A	
A2	Motor winding A	
B1	Motor winding B	
B2	Motor winding B	

*) The given configuration applies only to standard applications. Adaptation for other applications is described in the corresponding sections.

2.1.2.3 Indicators

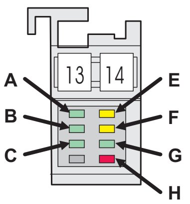


Fig. 2.1.2-5: Indicators g067002x

LED	Link	Designation	Status	Function
A	fixed	Status DI 1	off	Input DI 1: Signal level (0)
			green	Input DI 1: Signal level (1)
B	fixed	Status DI 2	off	Input DI 2: Signal level (0)
			green	Input DI 2: Signal level (1)
C	fixed	Status System voltage	off	No system voltage present
			green	System voltage present

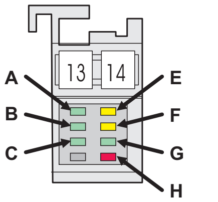


Fig. 2.1.2-6: Indicators
g067002x

LED	Link	Designation	Status	Function
E	Busy	Busy		The selected operating mode is active and not yet finished. This operating mode may have been discontinued.
		Positioning	off	Positioning not active, drive motionless
			yellow	Positioning active, drive in operation
		Move program	off	Move program not active.
			yellow	Move program active
		Referencing	off	Referencing not active, drive motionless
			yellow	Referencing active, drive in operation
		Jog Mode	off	Jog mode not active.
			yellow	Jog mode active, motor has been started using Direction_Pos or Direction_Neg. LED flashes briefly
		Mailbox mode	off	Mailbox active, but no command active, drive motionless.
			yellow	Mailbox active, and command active, drive in operation.
F	M_Program_ACK	Move program	off	No Move program being processed
			yellow	A Move program is currently in progress.
G	Stop_N_ACK	Drive Stop inverted	off	The bit Stop1_N or Stop2_N is 0. In addition, the motor is at standstill and frequency output is 0. Startup using Start is not possible.
			green	The bits Stop1_N and Stop2_N are both set to 1, or the drive is braking the unit.
H	Error	Write access to EEPROM/ Error code Group error	red flashing 10 Hz	Write access to EEPROM
			red, Blink-code	Group error, Error message (cf. Chapter 3.3, “Error Blink Codes”) issued

2.1.2.4 Operating Elements

The stepper controller 750-671 is not equipped with any operating elements. Changes to the configuration or parameters are made using the higher-order control too, or the WAGO-I/O-CHECK configuration tool.

2.1.2.5 Schematic Diagram

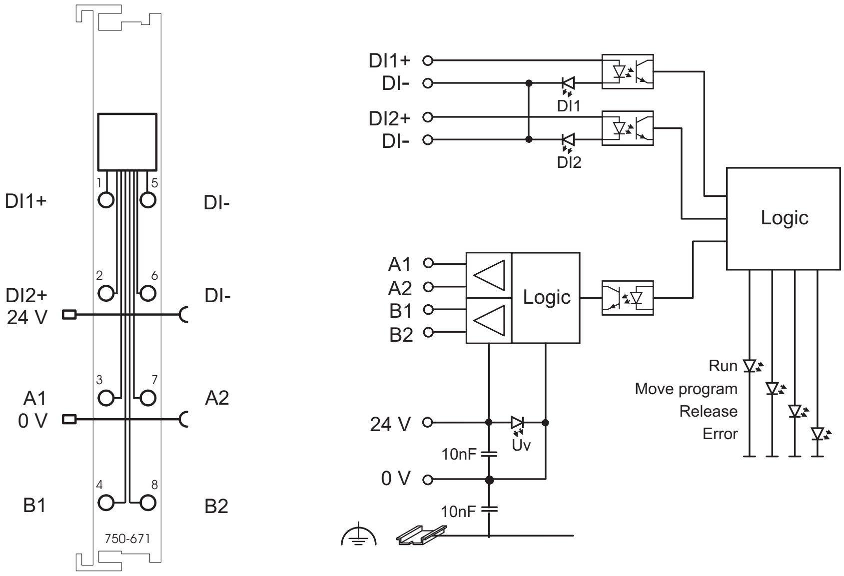


Fig. 2.1.2-7: Schematic diagram g067101e

2.1.2.6 Technical Data

Inputs		
Number of inputs		2 (DI1+, DI2+)
Input characteristik		Type 1 accordance with IEC61131-2 (2004)
Input voltage		DC -3 V ... 30 V
Signal voltage (0)		DC -3 V ... +5 V
Signal voltage (1)		DC 15 V ... 30 V
Input filter		100 µs
Input current $_{typ.}$		2.8 mA
Outputs		
Number of outputs		2 Outputs A1, A2 and B1, B2 for the connection of motor windings
Operating modes		Positioning, Referencing, Jog Mode, Move program
Resolution	Path	23 Bit + sign
	Velocity	15 bit + sign
	Acceleration	15 bit + sign
Microstepping		64 times internally
Output current$_{max.}$		1 A Nominal current[1)] per phase 1.5 A Short-time current [1)] per phase The derating factor is 1 % Kelvin from an ambient operating temperature of 25 °C (see Fig. 1.1.1 8: Derating graph)
Protective functions		- Short circuit of motor windings is permitted - Short circuit of motor windings against 0 V or 24 V results in the destruction of the output stage - Reverse polarity of the 24 V supply - Excess temperature of the output stage results in shutdown [2)]
Module Specific Data		
Voltage supply		via system voltage internal bus (5 V DC) and power jumper contacts (24 V DC)
Current consumption (system voltage 5 V DC) $_{ca.}$		85 mA
Current consumption (power jumper contacts 24 V DC) $_{ca.}$		0 mA + load
Voltage via power jumper contacts		24 V DC (-15 % ... +20 %) Protected against reverse polarity
Current via power jumper contacts $_{max.}$		10 A
Electrical isolation		500 V DC system voltage / field level (power jumper contacts)

Data width, internal	12 bytes input/output	
Dimensions W x H* x D * (from upper edge of rail)	12 mm x 64 mm x 100 mm	
Weight	approx. 50 g	
Standards and directives (see section 2.2 in manual on coupler / controller)		
EMC Immunity to interference	in acc. with EN 61000-6-2 (2001)	
EMC Emission of interference	in acc. with EN 61000-6-3 (2001)	
Approvals (See Section 2.2 of the Coupler/Controller Manual)		
CE	Conformity marking	

1) The figures are peak values
2) Detection system is not available

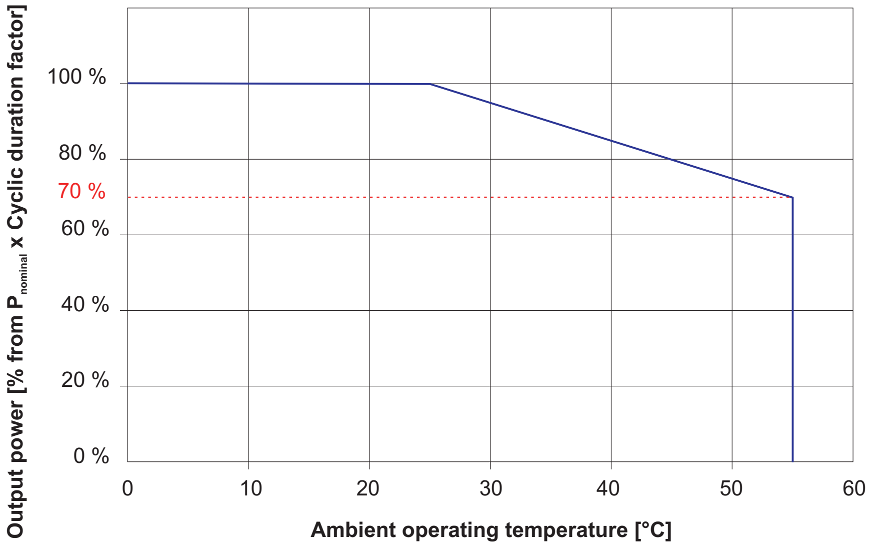


Fig. 2.1.2-8: Derating graph g067121e

2.1.2.7 Process Image

The 750-671 I/O module provides the fieldbus coupler/controller 12 bytes input and output process image via 1 logical channel. The data to be sent and received are stored in up to 7 output bytes (D0 … D6) and 7 input bytes (D0 … D6), depending on the operating mode. Output byte D1 and input byte D1 are reserved and have no function assigned. 1 I/O module control and status byte (C0, S0) and 3 application control and status bytes (C1 ... C3, S1 ... S3) provide the control of the data flow.

Please note
Mapping the process data of some I/O modules or their variations into the process image is specific for the fieldbus coupler/controller used. You will find both this information and the specific configuration of the relevant control/status bytes in the section on "Fieldbus Specific Configuration of Process Data" which describes the process image of the particular coupler/controller.

2.1.2.7.1 Overview

A basic distinction is drawn between the cyclic process image and the mailbox process image.

<table>
<tr><th></th><th colspan="2">Cyclic process image
(Mailbox deactivated)</th><th colspan="2">Mailbox process image
(Mailbox activated)</th></tr>
<tr><th>Off-set</th><th>Input Data</th><th>Output Data</th><th>Output Data</th><th>Input Data</th></tr>
<tr><td>0</td><td>Status byte S0</td><td>Control byte C0</td><td>Status byte S0</td><td>Control byte C0</td></tr>
<tr><td>1</td><td>Reserved</td><td>Reserved</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>2</td><td rowspan="7">Process data
D0 ... D6</td><td rowspan="7">Process data
D0 ... D6</td><td rowspan="6">Mailbox
MB0 ... MB5</td><td rowspan="6">Mailbox
MB0 ... MB5</td></tr>
<tr><td>3</td></tr>
<tr><td>4</td></tr>
<tr><td>5</td></tr>
<tr><td>6</td></tr>
<tr><td>7</td></tr>
<tr><td>8</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>9</td><td>Status byte S3</td><td>Control byte C3</td><td>Status byte S3</td><td>Control byte C3</td></tr>
<tr><td>10</td><td>Status byte S2</td><td>Control byte C2</td><td>Status byte S2</td><td>Control byte C2</td></tr>
<tr><td>11</td><td>Status byte S1</td><td>Control byte C1</td><td>Status byte S1</td><td>Control byte C1</td></tr>
</table>

Switching between the two process images is conducted through bit 5 in the control byte (C0 (C0.5). Activation of the mailbox is acknowledged by bit 5 of the status byte S0 (S0.5).

2.1.2.7.2 Control Byte 0, Status Byte 0

Control byte C0							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
0	0	MBX	0	0	0	0	0

MBX	Mailbox mode 0: Mailbox deactivated. 1: Mailbox activated.
0	Reserved

Status byte S0							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
0	ERR	MBX	X	X	X	X	X

MBX	Mailbox mode 0: Mailbox deactivated. 1: Mailbox activated.
ERR	Error signaling ERR (Status byte 0, Bit 6) follows the general error bit Error (Status byte 2, Bit 7). ERR can be enabled via a bit in the configuration table. ERR is not enabled in the default state. This means that errors will not result in a bit being set. 0: No error present. 1: Error present.
X	Reserved

Configuration of the control and status bytes C1 ... C3 and S1 ... S3 depends on the set operating mode; this is described in the associated sections.

2.1.2.7.3 Cyclic Process Image

The process image appears as follows when the mailbox is deactivated (C0.5 = 0):

Off-set	Input Data		Output Data	
0	S0	Status byte S0	C0	Control byte C0
1	Reserved		Reserved	
2	D0	Process data	D0	Process data
3	D1	Process data	D1	Process data
4	D2	Process data	D2	Process data
5	D3	Process data	D3	Process data
6	D4	Process data	D4	Process data
7	D5	Process data	D5	Process data
8	D6	Process data	D6	Process data
9	S3	Status byte S3	C3	Control byte C3
10	S2	Status byte S2	C2	Control byte C2
11	S1	Status byte S1	C1	Control byte C1

The configuration of the process data depends on the set operating mode; this is described in the associated sections.

A basic distinction is drawn between the following process images:

- Positioning,
- Jogging,
- Move program,
- Velocity.

2.1.2.7.4 Mailbox Process Image

The process image appears as follows when the mailbox is activated (C0.5 = 1):

Off-set	Input Data		Output Data	
0	S0	Status byte S0	C0	Control byte C0
1	Reserved		Reserved	
2	MB0	Opcode	MB0	Opcode
3	MB1	Status_Mbx	MB1	Control_Mbx
4	MB2	Reply Parameter byte 1	MB2	Request Parameter byte 1
5	MB3	Reply Parameter byte 2	MB3	Request Parameter byte 2
6	MB4	Reply Parameter byte 3	MB4	Request Parameter byte 3
7	MB5	Reply Parameter byte 4	MB5	Request Parameter byte 4
8	Reserved		Reserved	
9	S3	Status byte S3	C3	Control byte C3
10	S2	Status byte S2	C2	Control byte C2
11	S1	Status byte S1	C1	Control byte C1

The individual applications can be set using opcodes.

Opcodes are assigned to different topical areas and are described in the sections that follow.

The control byte and status byte for the mailbox have the following function:

Control_MBX							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Toggle-Flag	0	0	0	0	0	0	0

Toggle-Flag	If the data that have been written to the mailbox are to be accepted, the status of this bit is changed. Data is also accepted when an opcode is specified that is different from the previous one. Therefore: First write the data and then the opcode!
0	Reserved

Status_MBX							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Toggle-Flag	Return-Code						

Return-Code	The return code indicates whether the last command has been executed without any errors. If so, a value of 0 is returned. When the return code provides a value other than 0, you must check the corresponding opcode. The return messages are then individual.
Toggle Flag	The mailbox is evaluated when the status of the toggle flag is different in the Control_MBX. The status of the bit is then also changed.

2.1.2.8 Mailbox Mode

The mailbox expands the application range considerably.

The mailbox is activated when bit 5 of the control byte C0 is set to 1. Activation of the mailbox is acknowledged by bit 5 of the status byte S0.

Note
The Mailbox mode is not selected automatically when the mailbox is activated in the coupler/controller! Bit 7 of control byte C1 must be set to 1 for this.

2.1.2.9 Table Manager

Access to the tables is handled using the Table Manager. Possible types of tables:

- Move programs,
- Positioning of camshaft,
- Target positions,
- Configuration,
- Data recorder.

Several tables may exist at different storage locations for one type of table. The storage location is addresses using an index.

Index	Storage location
0:	not available / no table active / Factory Default for configuration / EEPROM
1:	RAM 1
2:	RAM 2

Exceptions to this are: Configuration and the data recorder use RAM1 exclusively. The data from the data recorder can not be copied to the EEPROM.

One table of each type of table can be activated and this is then evaluated in the Move mode.

One table of each type can be saved to the EEPROM. Tables in the EEPROM can not be activated, but only copied to a RAM table. If a table is present in the EEPROM, it will be copied automatically to RAM1 and RAM2 following a reset and is activated such that it can be directly executed in RAM1.
The tables are loaded from the control system by download. The following rules apply to downloading:

- A download is always conducted only into one table in the RAM.
- A download is only permitted when the target table is not active.

The download is checked for consistency using a checksum. If the checksum is not correct, the table that has been loaded is marked as invalid.

Tables may be copied. The following rules apply to copying of tables:

- The target table must be blank. Tables in the RAM can not be deleted using the command TBL_ERA.
- Copying is only permitted when the target table is not active.

The Table Manager detects whether a table is blank, valid or invalid (for example during a download or after a faulted download). This information is retained for each table in a status byte.

Access to a table (except for Configuration), including by other program modules, can be performed using the Table Manager. Depending on the table type, the following access options are available:

- Downloading of a table,
- Copying of a table,
- Deleting of a / all tables,
- Activating a table,
- Writing / Teaching of an element (position table only).

The position table stores target positions that can be queried using special commands. This position table enables target positions to be edited and taught, without having to change the Move program.

The table for the camshaft stores a bit sample that is output as a function of the position. Activation of a different table for the camshaft is performed immediately.

The table for configuration stores a data field that contains configuration data.

2.1.2.9.1 Download

Table download is performed to implement a transport layer for transferring relatively large data volumes via the I/O bus.
The data blocks that are to be transferred are fragmented into 4-byte blocks, which are then transferred to the module at each I/O module cycle. These data bytes are embedded in the mailbox and can be transferred simultaneously with the process data, ensuring control over this process while also in this mode.

A download is basically broken down into 3 phases:

1. Preparation for download using the command DLD_START
2. Transfer of data using the command DLD_CONT
3. Conclusion of download using the command DLD_END.

These commands are elucidated in the appendix.

2.1.2.9.2 Control

After downloading of tables types:

- Move program,
- Camshaft
- Position table

the tables, and the associated functions, must be enabled.

- The camshaft is always active after this.
- The Move program is active after this and can also be halted again, contrary to the camshaft.

These commands are elucidated in the appendix.

2.1.2.10 Configuration

The response of the stepper module is essentially determined by the settings in the configuration table. The configuration table is broken down into two sectors: Addresses 0 ... 127 directly describe the corresponding parameters, whereas addresses above 128 are interpreted as indicators. These indicators point to sources in the bit I/O table and are assigned to fixed targets in the same table.
The bit I/O table is broken down into two sectors: Addresses 0 ... 127 describe the data sources; addresses 128 ... 255 described the targets to which the indicator can point.

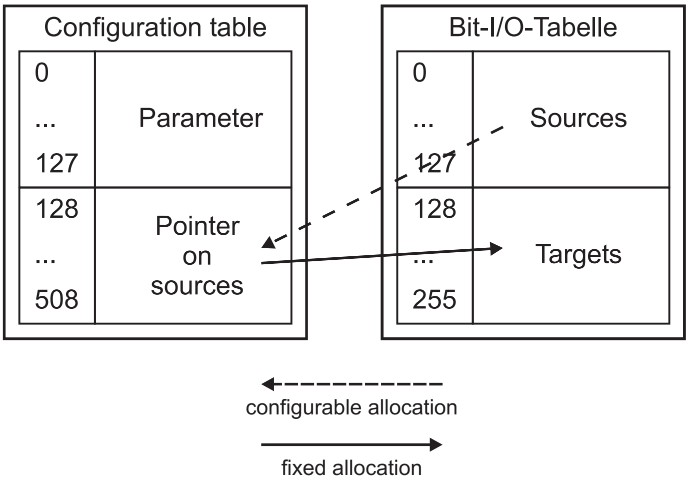


Fig. 2.1.2-9: Configuration and bit I/O tables g067x30e

2.1.2.10.1 Configuration Table

The table below shows an excerpt from the configuration table.

Configuration variable	Offset (Dec.)	Bit Offs.	Data type	Default	Range	Description	
User_Conf_Id	0		UINT16	0	0 ... 50000	Data set numbers can be freely assigned by the user. Numbers above 50000 are reserved.	
ConfVersion	2		UINT8	4	0 ... 254	Configuration version number	
Application_Selector	3		UINT8	1	0 ... 2	Switching of applications. The appropriate process image is activated when a new application is selected.	
						0:	Reserved
						1:	Positioning controller
						2:	Velocity control
Freq_Div	4		UINT16	200	4 ... 65535	Sets the prescaler for the maximum velocity	
Acc_Fact	6		UINT16	80	1 ... 65535	Sets factor for maximum acceleration	

2.1.2.10.1.1 Configuration of Basic Parameters

The integrated output stage runs with 64 times microstepping. This means that the full step of a motor is divided into 64 single steps.
The following equations are related to microstepping.

2.1.2.10.1.1.1 Application Selection

Application_Selector, Offset 0, Range [0 ... 2]

The Application_Selector determines the basic function:

Value	Application
0	Reserved
1	Positioning controller
2	Velocity control/Frequency output/PWM

2.1.2.10.1.1.2 Prescaler for Maximum Velocity

Freq_Div, Offset 4, Range [4 ... 65335]

The maximum internal output frequency is derived from an internal 2 MHz cycle by a prescaler. When the smallest possible prescaler (4) is selected, a maximum internal frequency of 500,000 Hz is yielded.

2.1.2.10.1.1.3 Factor for Maximum Acceleration

Acc_Fact, Offset 6, Range [1 ... 65535]

Acceleration is given in steps/s². The specified value is multiplied by the acceleration factor (Acc_multiplier) and then divided by the frequency prescaler (Freq_Prescaler).

a=acceleration value * Acc_multiplier/Freq_Prescaler

Acceleration value: Setting via the process image, or parameter in an opcode.

2.1.2.10.1.1.4 Reference Run

Reference_Offset, Offset 108, Range [±8388607]

Position of reference switch.

Reference_Mode, Offset 112

Mode for referencing on start of a reference run using the control bit M_Reference. At the start of a reference run via the mailbox using the Move command START_REFERENCING, the call parameters are used (and NOT the following configuration bits).

Bit 1:	
0:	Reference run to reference switch
1:	Reference run to limit switch
Bit 2:	
0:	Reference run to negative end of a reference switch
1:	Reference run to positive end of a reference switch
Bit 3 ... 7: Reserved	

2.1.2.10.1.1.5 Jog Mode

Acc_Fact, Offset 44, Range [1 ... 25000]

Default setup speed.
The current moving speed is used when this parameter is 0.

Acc_Fact, Offset 62, Range [0 ... 32767]

Acceleration for Jog mode and Referencing.

2.1.2.10.1.1.6 Ramps

Acceleration_Stop_Fast, Offset 46, Range [0 ... 32767]

Default acceleration for STOP mode; the current acceleration is used when this parameter is 0.

Acceleration_RampUp, Offset 48, Range [0 ... 32767]

Default acceleration for acceleration phase.

Acceleration_RampDown, Offset 50, Range [0 ... 32767]

Default acceleration for delay phase.

Acceleration_RampUp_Param, Offset 52, Range [0 ... 16777216]

Default acceleration time or acceleration path

Acceleration_RampDown_Param, Offset 56, Range [0 ... 16777216]

Default deceleration time or deceleration path

Acceleration_Modes, Offset 60

Bit 0 ... 1: AccType (Acceleration type)	
0:	constant acceleration
1:	linear rise in acceleration; the period for acceleration increase is Acceleration_RampUp_Param
2:	$\sin^2$ rise in acceleration; the period for acceleration increase is Acceleration_RampUp_Param
3:	Reserved
Bit 2 ... 3: AccParam (Acceleration parameter)	
0:	no modification
1:	Acceleration_RampUp_Param interpreted as the acceleration period
2:	Acceleration_RampUp_Param interpreted as the acceleration path
3:	Reserved
Bit 4 ... 5: DecType (Deceleration type)	
0:	constant acceleration
1:	linear rise in acceleration; the period for acceleration increase is Acceleration_RampUp_Param
2:	$\sin^2$ acceleration; the period for acceleration increase is Acceleration_RampUp_Param
3:	Reserved
Bit 6 ... 7: DecParam (Deceleration parameter)	
0:	no modification
1:	Acceleration_RampUp_Param interpreted as the acceleration period
2:	Acceleration_RampUp_Param interpreted as the acceleration path
3:	Reserved

2.1.2.10.1.1.7 User Scaling Factors

The internal unit of measure for the position is a micro step.
Velocity and acceleration are derived form this.

The scaling factors enable adaptation and conversion to application-specific units (path in m, mm, degrees, and velocities in m/s, degrees/s).

Conversion is performed by multiplying by a configurable factor *_Mult and then dividing by a configurable factor *_Div.

This way, fractions can be set with high accuracy within a wide range.

Pos_Mult, Offset 20, Range [1 ... 65535]
Pos_Div, Offset 22, Range [1 ... 65535]

Scaling factors for position.

Speed_Mult, Offset 28, Range [1 ... 65535]
Speed_Div, Offset 30, Range [1 ... 65535]

Scaling factors for speed.

Acc_Mult, Offset 32, Range [1 ... 65535]
Acc_Div, Offset 34, Range [1 ... 65535]

Scaling factors for acceleration.

If the conversion violates the permissible value range, the error message 1511 or 1512 is issued.

2.1.2.10.1.1.8 Brake

Braketime_Turn_On, Offset 100, Range [1 ... 8388607]

Activation time for brake in [ms].

Braketime_Turn_Off, Offset 104, Range [0 ... 8388607]

Switch-off time for brake in [ms].

2.1.2.10.1.1.9 Hardware/Software Configuration

HwSwConfig, Offset 19

Bit 0 … 1: Reserved	
Bit 2: Drive_Direction (Direction of rotation inversion)	
0:	Output signal processed directly
1:	Output signal: Direction of rotation inverted
Bit 3 … 6: Reserved	
Bit 7: Program_Autostart (Move program Autostart – Normal mode)	
0:	Move program activated only via Move program or Mailbox mode.
1:	Move program activated immediately after startup, see description.

2.1.2.10.2 Configuration using Control Byte C2

The following values can be configured using the control byte C2 in the standard configuration.

2.1.2.10.2.1 Frequency Prescaler

The values for the frequency prescaler (Freq_Prescaler) are determined by the bits Freq_Range_Sel in control byte 2 (C2.0 and C2.1). If both of these bits are zero (0), the value for the parameter Freq_Div in the configuration table is used.

Freq_Range_Sel	'00'	'01'	'10'	'11'
Freq_Prescaler (frequency prescaler)	Freq_Div <> 0: Parameter Freq_Div from configuration table Freq_Div = 0: 200	80	20	4
$f_{p,max}$	Freq_Div <> 0: 2 MHz / Freq_Div Freq_Div = 0: 10 kHz	25 kHz	100 kHz	500 kHz

Note
If Freq_Range_Sel = 0 and configuration parameter Freq_Div = 0, the variable Freq_Prescaler is set to 200.

The moving speed is determined by the pulse frequency (fp), which is determined by the output data Velocity (D0 and D1) and by the prescaler Freq_Prescaler.

fp = Velocity * 80 / Freq_Prescaler [Hz]

The acceptable velocity range is 1 … 25000. The setting for the pulse frequency in [Hz] is given by selecting Freq_Prescaler = 80.

Note
The bits Freq_Range_Sel may only be modified when the control system is deactivated! These bits are therefore only accepted when Enable is not set.

2.1.2.10.2.2 Acceleration Factor

The value for the acceleration factor (Acc_Multiplier) is determined by the bits Acc_Range_Sel in control byte 2 (C2.2 and C2.3). If both of these bits are zero (0), the value for the parameter Acc_Fact in the configuration table is used.

Acc_Range_Sel	'00'	'01'	'10'	'11'
Acc_Multiplier (acceleration factor)	Acc_Fact <> 0: Parameter Acc_Fact from configuration table Acc_Fact = 0: 8	80	800	8000
Acceleration period T to f_{max} at max. acceleration 32767	Acc_Fact <> 0: Setting from configuration Acc_Fact = 0: 7600 ms	760 ms	76 ms	7.6 ms

Note
If Acc_Range_Sel = 0 and configuration parameter Acc_Fact = 0, the variable Acc_Multiplier is set to 8.

Acceleration is determined by the output data Acceleration (D2 and D3) and the prescaler Freq_Prescaler and by the acceleration factor Acc_Multiplier.

a = Acceleration * Acc_Multiplier / Freq_Prescaler [Hz/s]

The permissible acceleration range is 1 … 32767.
Acceleration is set in [Hz/s] when the acceleration factor Acc_Multiplier is selected equal to the prescaler Freq_Prescaler.

Note
The bits Acc_Range_Sel may only be modified when the control system is deactivated! These bits are therefore only accepted when Enable is not set.

2.1.2.10.3 Configuration via Mailbox Mode

The configuration data set elements are determined by an address. This address is yielded from the table assignment and may be greater than one byte; in this case the byte with the lowest value will be present at the specified address. The size of the element must also be given for unique access to an element.

The table containing the configuration values is given in the appendix in section 3.5, „Configuration Variables".

The complete data set is loaded to the RAM on a download and is then saved to the EEPROM; a module warm start is then carried out. The download is conducted using the commands DLD_START, DLD_CONT and DLD_END. These commands are explained in Chapter 2.1.2.9, "Table Manager".

The writing procedure to the EEPROM is signaled by LED H.

The download is conducted by the table manager. The configuration data set is not saved to the EEPROM until successful completion of the download. A warm start is carried out subsequently, regardless of whether the download was successful.

The Reset status bit is set after the warm start; this must be canceled using the Reset_Quit control bit. Only then is the module operational again.

As an alternative to this, individual parameters can be specifically modified while the module is in operation.

This requires the diagnostics opcodes (see Chapter 2.1.2.14, "Advanced Diagnostics").

If a valid configuration data set is available in the EEPROM after power-on, reset or a warm start of the module, this data set is loaded to the RAM; if not, the factory default data set is loaded, i.e. the module is restored to the WAGO as-delivered status.

2.1.2.10.4 Digital Signals and Signal Linking

The vital binary signals are addressed by central access functions. This permits easy, external access to all bits and allows linking of the bits to one another to be parameterized. Access to individual bits is performed using an index 0 … 255.

A bit function is defined by the function unit that sets the bit (source) and the function unit that reacts to this bit (target). Only the source is defined for output bits, and only the target for control bits. Only status bits have set links between two function units; they can, however, also be queries as output bits for further processing. Bit for which the source is not fixed are designated as linkable bits.

The universal filter functions FILT1 ... FILT8 possess a special status. The inputs for these filter functions are linkable bits that can be linked to any other bit. A query of these bits, on the other hand, provides information about the status of the filter. As a result, these bits represent a function between inputs and outputs.

The table containing the available bits is given in the appendix in section 3.4, „Bit field for I/O driver".

The following conventions apply:

- Source bits are assigned numbers 0 to 127 and may not be used as target bits. A source bit may reference several target bits.
- Target bits are assigned numbers 128 to 255 and may also be used as source bits. Target bits have exactly one source.
- References are stored in the configuration table. The names of the table entries correspond to those in the bit table. The prefix Ptr is placed in front of the identifier.
- The standard link between the source and target is entered in the column "Target/Source". This corresponds to the WAGO default settings (FACTORY_DEFAULT).

2.1.2.10.5 Linking of Bits

The expanded parameters are set using pointers (indices). The address for these pointers indicates a corresponding address in the bit I/O field. Allocation using names is also provided. "Ptr_nnn" indicates the variable "nnn".

Actual allocation is conducted using the content of the pointer.

The figure below illustrates allocation with an example of motor shutdown using Stop1_N and Stop2_N.

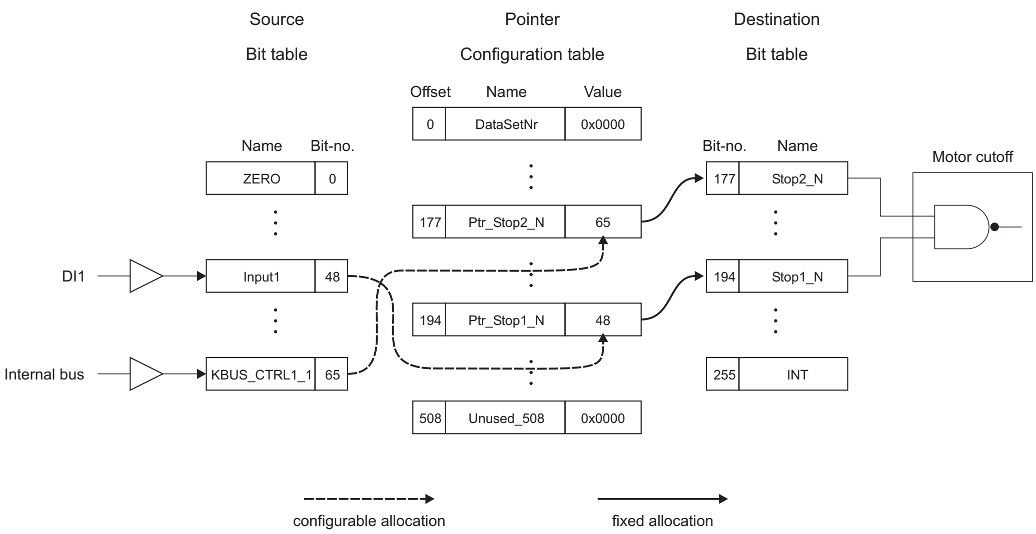


Fig. 2.1.2-10: Linking of bits p067x18e

Index Ptr_Stop1_N has a value of 48 (0x30), thus assigning Input1 to the Stop1_N variable.
Index Ptr_Stop2_N has a value of 65 (0x41), thus assigning control bit C1.1 to the Stop2_N variable.

The table excerpts given below show the corresponding entries for the configuration and bit I/O table.

Configuration variable	Address		Data type	Default	Range	Description
	Dec.	Hex.				
…						
Ptr_Stop2_N	177	0xB1	UINT8	0x41	0 ... 255	Source for linkable bit 0xB1
…						
Ptr_Stop1_N	194	0xC2	UINT8	0x30	0 ... 255	Source for linkable bit 0xC2
…						

Name	Bit number		Type	Default allocation		Description
	Dec.	Hex.		Target/Source	Bit no.	
…						
Input1	48	0x30	SRC	KBUS_ST3_0 Stop1_N	0x90 0xC2	Input 1
…						
KBUS_CTRL1_1	65	0x41	SRC	Stop2_N	0xB1	Internal bus control byte 1 bit 1
…						
Stop2_N	177	0xB1	DST/ SRC	KBUS_ CTRL1_1	0x41	Drive Stop 2 inverted
…						
Stop1_N	194	0xC2	DST/ SRC	Input1	0x30	Drive Stop 1 inverted
…						

Linkable bits can be programmed for any source. This "linkability" enables flexible configuration and flexible arrangement of module terminal assignments. For example, the Start linkable bit can be set to the Input 1 fixed bit.

Note
A linkable bit can also be linked to another linkable bit, but the maximum number of nesting levels is four (4). Too many nesting levels will yield an ambiguous result and the error ERR_LINK_NESTING will be issued.

Note
The nesting levels are not checked until the system run time.

Linking of bits is performed in the device configuration and can only be changed by reconfiguration. An exception to this rule are the bits that are linked to MZERO and MONE. These bits can be set or reset as often as required during operation using mailbox commands of the Move program.

The example below provides an illustration of this:

Input 2 is normally set as the reference input. However, in a certain application is may be more advantageous to use Input 2 for specifying the direction of movement for the Jog mode. Moreover, a "1" at the input should signify that the motor is moving in a positive direction.

The address and length of the configuration variables are given in the appendix in section 3.5, „Configuration Variables".

Configuration variable	Address		Data type	Default	Range	Description
	Dec.	Hex.				
…						
Ptr_FILT1	168	0xA8	UINT8	0x00	0 ... 255	Source for linkable bit 0xA8
...						
Ptr_Direction_ Neg	187	0xBB	UINT8	0x53	0 ... 255	Source for linkable bit 0xBB
…						
Filter1_Function	224	0xE0	UINT8	0	0 ... 11	Function of filter: 1 = inversion
…						

The bits required for this are given in the appendix in section 3.4, „Bit field for I/O driver“.

Name	Bit number		Type	Default allocation		Description
	Dec.	Hex.		Target/Source	Bit no.	
…						
Input2	49	0x31	SRC	KBUS_ST3_1 Set_Reference	0x91 0xBC	Input 2
…						
FILT1	168	0xA8	FILT	ZERO	0x00	Timer / Filter 1
…						
Direction_Neg	187	0xBB	DST/SRC	KBUS_CTRL3_3	0x53	Move in negative direction.
…						

A prerequisite for this is that the mailbox has already been activated. This is explained in Chapter 2.1.2.8, “Mailbox Mode”.

Applying the procedure described above, the bits for the standard configuration control and status bytes can also be modified.

Attention
Any change to the standard configuration will nullify the description given for the changed items.

Unassigned linkable bits can be linked to the constant bits ZERO or ONE.

2.1.2.10.5.1 Special Bits: ZERO, ONE, MZERO and MONE

The ZERO and ONE bits are fixed. The ZERO bit is always deleted and has a value of 0, the ONE bit is always set and has a value of 1.
Linkable bits can be set to a fixed value using ZERO and ONE bits.

The MZERO and MONE bits have the same function, but also possess an additional function when they are the source for linkable bits.
The status of linkable bits is given by the source bit. As a result, linkable bits can not be changed by commands or access functions. An exception to this rule are the linkable bits that are linked to MZERO and MONE.
A bit that is linked to MZERO is first deleted after a reset, but can be manipulated as required using the mailbox command or the Move program. A bit that is linked to MONE is initially set after a reset and is otherwise treated the same as a bit linked to MZERO.

2.1.2.10.5.2 Filters, Low Pass, Timers and Counters

The filter is configured using the table with the configuration values (see Chapter 3.5, „Configuration Variables“).

Eight (8) special bits are defined that can implement the following functions:

1. Inverting,
2. Starting edge filter,
3. Low pass,
4. Pulse extension,
5. Monoflop, not retriggerable,
6. Pulse delay,
7. Math,
8. Counter, up,
9. Counter, up, stop at overrun,
10. Counter, down,
11. Counter, down, stop at overrun.

The functions Inverting and Starting edge filter react to the linkable input bit and trigger the output immediately.

The functions Low pass, Pulse extension, Monoflop and Pulse delay react to the linkable input bit and trigger the output in accordance with the selected function at a time constant for which parameters can be assigned. This time can be set between 0 … 16777215 ms.

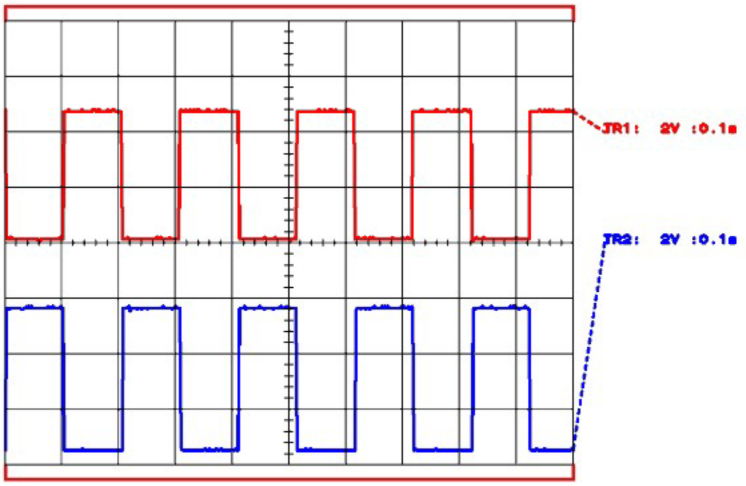


Fig. 2.1.2-11: Inverting p067020x

There are two statuses for the starting edge filter: After a reset this filter is in the Wait status, as long as the input is active. The output provides the value zero. As soon as the input signal is zero for the first time the filter switches to its operating status and the input signal is passed on to the output without being changed. An operational change back to the Wait status is not provided.

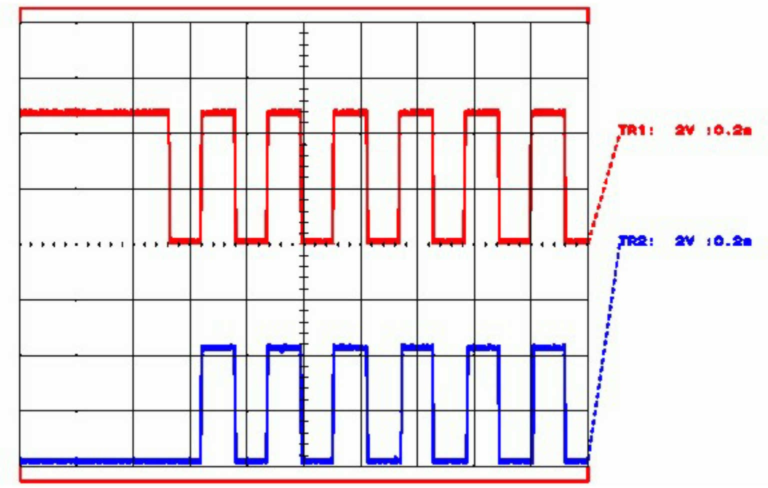


Fig. 2.1.2-12: Starting edge filter p067021x

The Low pass does not accept any change of the input signal until the new status is constant during the runtime.

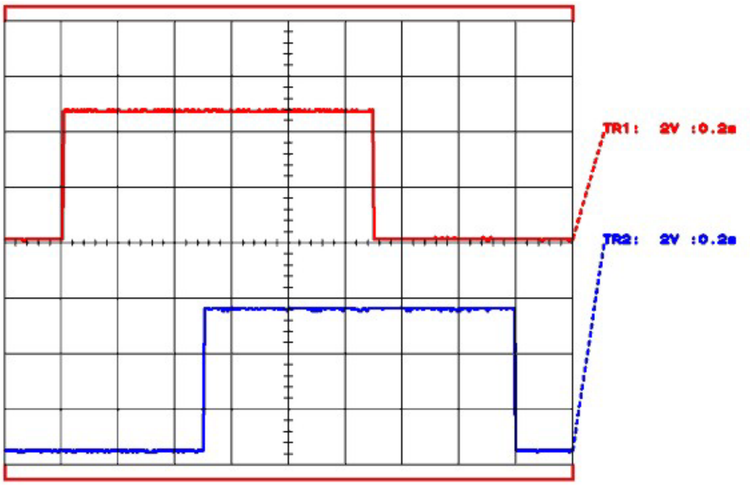


Fig. 2.1.2-13: Low pass 500ms p067022x

Pulse extension sets the output on an input 0→1 edge. The output is reset when the set time expires after the 1→0 edge. Retriggering during the runtime is possible.

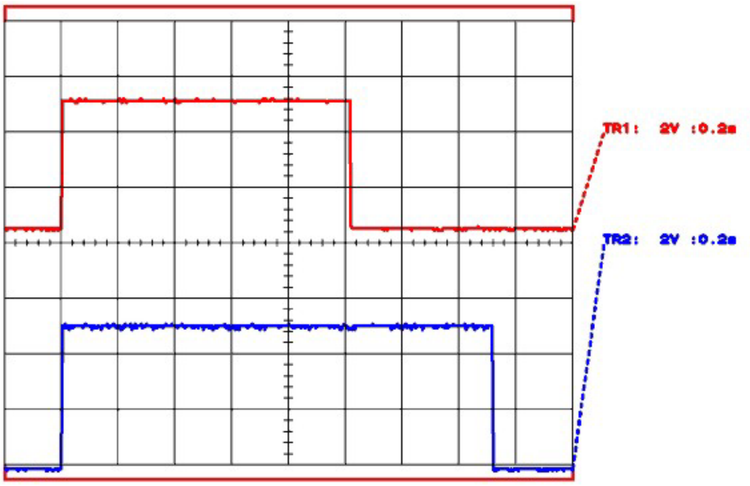


Fig. 2.1.2-14: Pulse extension 500ms p067023x

The monoflop function sets the output on an input 0→1 edge. The output is reset after the set time expires. Retriggering during the runtime is not possible.

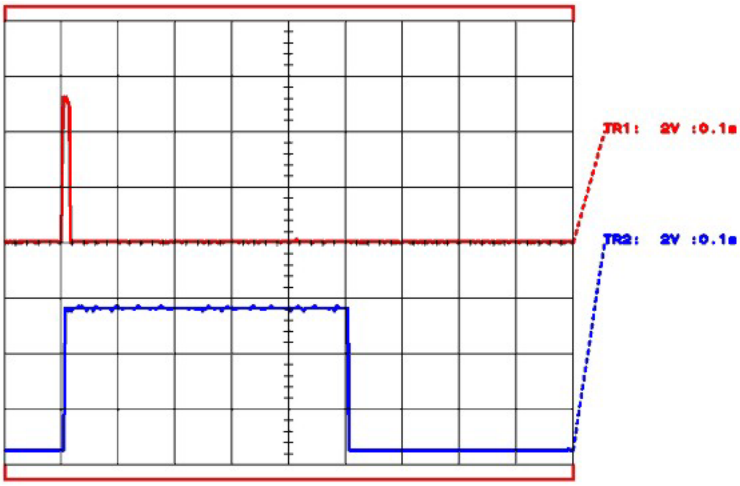


Fig. 2.1.2-15: Monoflop 500ms not retriggerable, with short input pulse p067024x

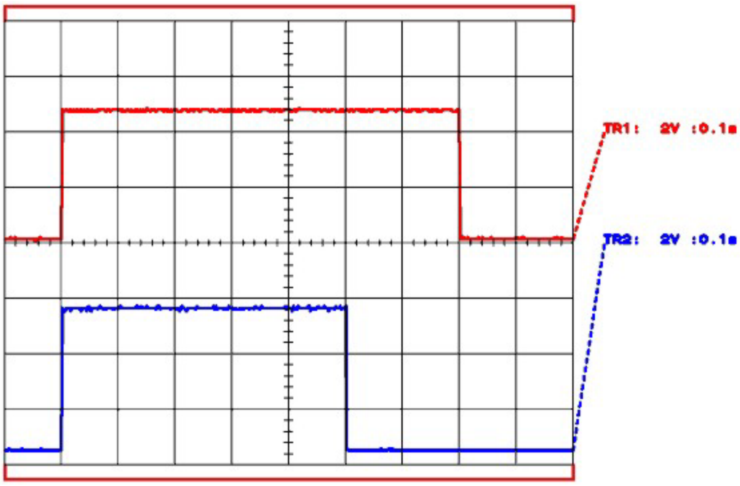


Fig. 2.1.2-16: Monoflop 500ms not retriggerable, with long input pulse p067025x

Pulse extension sets the output when the set time has expired after an input 0→1 edge. The output is reset as soon as the input is deactivated. The output is not reset if the input is deactivated before the set time expires.

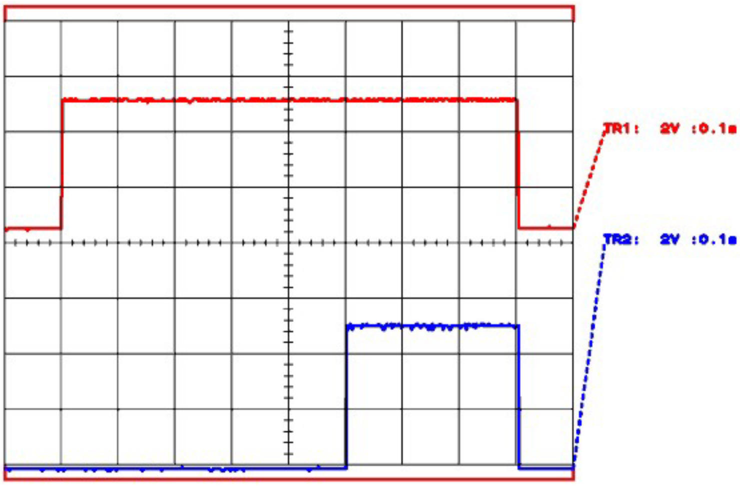


Fig. 2.1.2-17: Pulse delay 500ms p067026x

Application example: Filters connected in series: Monoflop 500ms, inverting and a second monoflop 100ms.

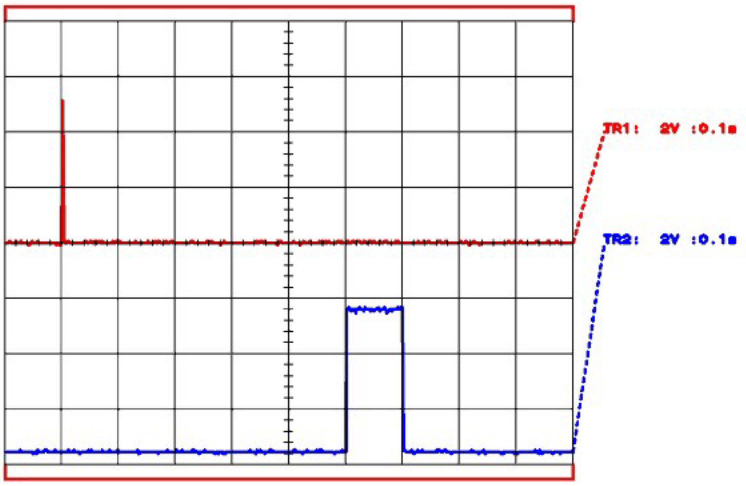


Fig. 2.1.2-18: Filters connected in series: Monoflop 500ms, Inverting and second monoflop 100ms p067027x

The math function reacts independently of the input bit. The value for this value can only be set to a value, incremented or decremented using commands. These commands can be transferred in the mode "Mailbox Move command", or processed as a component of the downloaded program in the mode "Program mode". The output is set when the counter has a value other than zero and is reset when the filter reaches zero. The commands for modifying the filter can only be used in the function "Math" or "Counters".

In the mode "Counter, up" / "Counter, down" the filter value is raised or lowered by one for each 0→1 edge of the filter input. The maximum counting frequency for external signals is 1 kHz; internal signals are normally evaluated once per program cycle.

The mode "Counter, up, stop" / "Counter, down, stop" behaves in the same manner as "Counter, up" / "Counter, down", except that counting is halted when the filter value reaches zero. What this also means is that the filter value must have a starting value other than zero, with subsequent counting then performed to zero.

2.1.2.10.6 Move Commands

These commands are classified as follows:

- Table commands
- Move Commands
- Auxiliary commands

In addition to pure Move commands, auxiliary commands and table commands are also accepted. The Move commands are passed along to the command interpreter. The table commands and auxiliary commands are required exclusively for the Move program mode.

A process is started using Move commands; there is no waiting for the end of the process (such as reaching a target).

Most commands are processed directly, except for the following cases:

Move commands are not started until the previous Move command has reached its target. Table processing is interrupted in this case until the target is reached.

An exception here are the direct Move commands _IMM (immediate), which discontinue the Move command currently in progress and are directly processed.

Commands that wait on an event are repeated until the event occurs.

The Move command manager decides from which source the Move commands are to be accepted. Potential sources are:

- Status control,
- Limit switches,
- Move commands via mailbox commands,
- Positioning,
- Referencing,
- Jog mode,
- Program mode.

Status control and limit switches are handled with priority; they can only execute commands for braking the drive. Switching between the other setpoint sources is performed by selecting the corresponding mode.

A Move program is used to execute individual movements one after the other. Some values can also be set using the Move program.

There are two ways of processing a Move program:

- Program mode:
  The individual commands are compiled and loaded to the module by download. The Move program can then be executed there through the Program mode (see Chapter 2.1.2.11.1.7 , “Move program mode”)

- Move task via mailbox:
  The module can be operated via the mailbox using the move commands. Movement can be made directly to different positions. This command is accepted only when the mode "Move task via mailbox"(Chapter 2.1.2.11.2, ”Move Mode via Mailbox”) has been activated.

Each individual step in the Move program has the following format:

Byte	Meaning
1	Command
2	Data 1
3	Data 2
4	Data 3

The individual commands are elucidated in the appendix in section 3.2, „Commands for Move Mode “.

2.1.2.10.7 Scaling, Number Ranges and Units

Stepper drives rotate at a defined angle on each pulse. The software is oriented toward this pulse output and the internal unit of measure for the position is, accordingly, a "Microstep". Velocity and acceleration are derived form this. The interface enables adaptation and conversion to application-specific units (path in m, mm, degrees, and velocities in m/s, degrees/s).

2.1.2.10.7.1 Internal Units of Measure

2.1.2.10.7.1.1 Time

On account of the periodical processing employed, time is measured in TICKS, with a TICK being the duration of one scan interval. A TICK is equal to one millisecond. Physical units based on time are converted accordingly.

2.1.2.10.7.1.2 Travel

The travel, or path, is measured in "steps". The number range is 24 bits, including sign.

Position range: -8388608 ... +8388607,
Presentation in two's complement.

2.1.2.10.7.1.3 Velocity

Velocity is measured in "steps per unit of time", with the range being ±15 bits.

Velocity range:
Velocity $_{min}$ ... Velocity $_{max}$ = -25000 ... +25000.

The maximum pulse rate is 500000 Hz. This frequency is derived from an internal 2 MHz cycle by a prescaler.
Parameters can be assigned to the prescaler, with the permissible range being Freq_Prescaler = 4 ... 65535.

The maximum pulse rate f_{max} is calculated by:

$$f_{max} = 25000 / (12.5\text{ ms} * \text{Freq\_Prescaler}) = 2\text{ MHz} / \text{Freq\_Prescaler} = 500000\text{ Hz} \dots 30.5\text{ Hz}$$

$$\text{Resolution} = 1 / (12.5\text{ ms} * \text{Freq\_Prescaler}) = 80\text{ Hz} / \text{Freq\_Prescaler} = 20\text{ Hz} \dots 1.2\text{ mHz}$$

Freq_Prescaler	**f_{max}: maximum velocity/frequency**	**Frequency resolution**
4	500000 Hz	20 Hz
80	25000 Hz	1 Hz
65535	30,5 Hz	1.2 mHz

To ensure the highest degree of accuracy, the value Freq_Prescaler should be set for the smallest whole value at the greatest speed required by the motor.

For example, if a stepper motor is to operate at 200 steps and 64 microsteps at speeds between 0 … 20 s^{-1} , a Freq_Prescaler range of 4 ... 7 is possible. A setting of Freq_Prescaler = 7 provides the highest degree of accuracy and frequency resolution.

Freq_Prescaler = 2000000 Hz / f_{max}

with f_{max} = Speed * Number of steps * Microsteps = n * Z * M yields:

Freq_Prescaler = 2000000 Hz / (n * Z * M)
= 2000000 / (20 * 200 * 64) = 7.8125

At this selected Freq_Prescaler, the module can output a maximum pulse frequency of 285714.29 Hz at a setpoint of 0 … 25000, which corresponds to a motor speed of 22.32 Hz. For a specified speed of exactly 20 Hz, a setpoint of

Velocity = (f / f_{max}) * Velocity_max = (n * Z * M / fmax) *25000

= (n * Z * M / (2000000 / Freq_Prescaler)) *25000

= 20 Hz * 200 * 64 * 7 * 25000 / 2000000 = 22400

must be specified in the process image.

Alternatively, an attempt can also be made to achieve a decadic relationship between the setting and the motor speed by selecting a different Freq_Prescaler. For example, Freq_Prescaler could also be selected as 6, in which case the module could output a maximum pulse frequency of 333333,3 Hz at a setpoint of 0 … 25000, which corresponds to a motor speed of 26,04 Hz. For a speed of 20 Hz a setpoint of

20 Hz * 200 * 64 * 6 * 25000 / 2000000 = 19200 <~> 20000 – 4 %

must be specified. Here, the setting is nearly 100 times the motor speed.

At a setting of Freq_Prescaler = 80, the numerical speed value is equal to the output frequency in [Hz].

These values can also be specified using user-specific units that can be defined. For this, the specified data for speed is multiplied by a factor Speed_Mult prior to internal processing and then divided by a factor Speed_Div. If the internal → external or external → internal conversion violates the permissible value range, the error message
1513 (UNITS_SPEED_INT_RESULT) or
1514 (UNITS_SPEED_USER_RESULT) is issued.

2.1.2.10.7.1.4 Acceleration

The prescaler for velocity also figures into the definition for acceleration. Acceleration is given as

(Steps / s^2) * Acc_Multiplier / Freq_Prescaler.

The initial factor Acc_Multiplier is determined such that the highest and lowest required acceleration can be easily specified.

Acceleration is scaled to [Hz/s] for Acc_Multiplier = 1 and Freq_Prescaler = 1.

These values can also be specified using user-specific units that can be defined. For this, the specified data for acceleration is multiplied by a factor Acc_Mult prior to internal processing and then divided by a factor Acc_Div. If the internal → external or external → internal conversion violates the permissible value range, the error message
1515 (UNITS_ACC_INT_RESULT) or
1516 (UNITS_ACC_USER_RESULT) is issued.

2.1.2.10.7.2 External Units of Measure

Internal representation can also be converted to application-specific units using conversion factors. Conversion is performed by multiplying by a configurable factor (*_MULT) and then dividing by a configurable factor (*_DIV). This way, fractions can be set with high accuracy within a wide range. These factors are explained in greater detail in the following sections.

2.1.2.10.7.2.1 Path

The setting data for the positions are multiplied by the configuration factor Pos_Mult prior to internal processing and then divided by the configuration factor Pos_Div.

For example, is a step (or microstep) corresponds to travel of 0.12 mm, the setting can be given in µm by selecting Pos_Mult = 1 and Pos_Div = 120, or in mm by selecting Pos_Mult = 25 and Pos_Div = 3.

If the internal → external or external → internal conversion violates the permissible value range, the error message
1511 (UNITS_POS_INT_RESULT) or 1516 (UNITS_POS_USER_RESULT) is issued.

2.1.2.10.7.2.2 Velocity

The setting data for velocity are multiplied by the configuration factor Speed_Mult prior to internal processing and then divided by the configuration factor SPEED_DIV and the prescaler Freq_Prescaler.

If the internal → external or external → internal conversion violates the permissible value range, the error message
1513 (UNITS_SPEED_INT_RESULT) or
1514 (UNITS_SPEED_USER_RESULT) is issued.

2.1.2.10.7.2.3 Acceleration

The setting data for acceleration are multiplied by the configuration factor Acc_Mult prior to internal processing and then divided by the configuration factor Acc_Div.

If the internal → external or external → internal conversion violates the permissible value range, the error message
1515 (UNITS_ACC_INT_RESULT) or
1516 (UNITS_ACC_USER_RESULT) is issued.

2.1.2.11 Positioning

2.1.2.11.1 Operation via Cyclic Process Image

Different operating modes are available with the stepper modules. These are described in the following sections.

The application Stepper positioning control is selected using the configuration parameter Application_Selector = 1.

2.1.2.11.1.1 Selecting a Mode

A mode can only be selected when the module is ready for operation and no other mode is active. This is the case when the status bits Ready and Stop_N_ACK are active and Start_ACK and Start are not active.
If these condition is not fulfilled, but a mode request bit is nevertheless set, mode selection is delayed until this condition is fulfilled.
This ensures that any tasks already in progress are ended properly.

A mode is selected by setting the mode request bit. If a mode is already active, setting of a further mode request bit has no function; the old mode (in progress) is continued.

Selection of a mode is confirmed by the associated mode status bit *_ACK.

2.1.2.11.1.2 Ending a Mode

A mode is ended when the associated mode request bit is canceled. If the drive is still in motion, it is then braked at the acceleration Acceleration_Stop_Fast. The assigned mode status bit will not be canceled until the drive comes to a standstill.

The Standstill status bit is set when the drive is motionless.

A mode is also ended when Stop_N_ACK is reset. If the drive is still in motion, it is then braked at the acceleration Acceleration_Stop_Fast. The assigned mode status bit will not be canceled until the drive comes to a standstill.

A mode is also ended when Ready is reset. Velocity is set immediately to zero, regardless of the status of the drive. The assigned mode status bit is canceled immediately. This procedure poses the risk of losing control over the drive and allowing the motor to run down in an uncontrolled manner, in particular at loads with high moments of inertia. This is also normally associated with step losses, meaning a reference run must subsequently be started.

2.1.2.11.1.3 Sequence Diagram for Selection and Ending of Modes

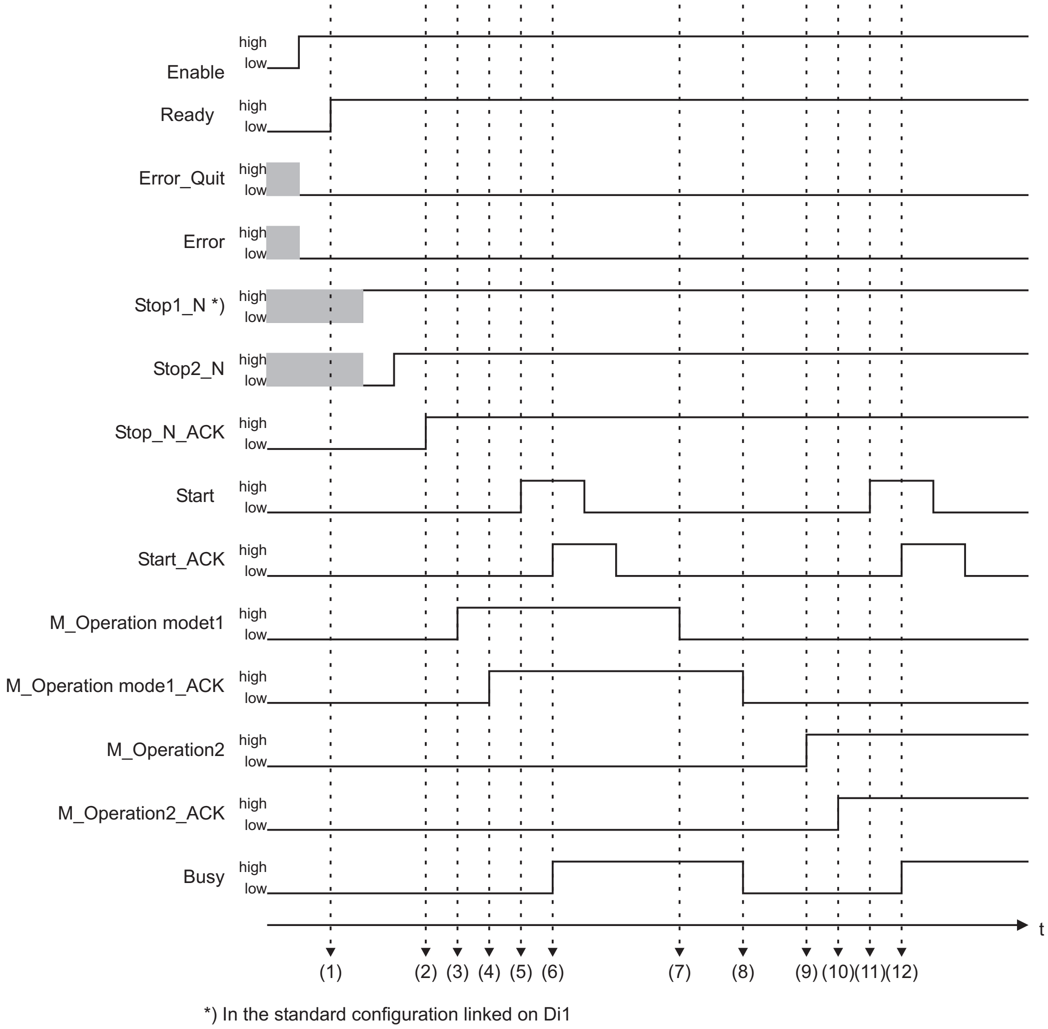


Fig. 2.1.2-19: Sequence diagram for Positioning g067x20e

(1)	The module is ready for operation when no error is present after setting of Enable.
(2)	An operating mode can be selected when Stop_N_ACK has been set.
(3)	Mode 1 is selected.
(4)	Mode 1 has been accepted by the module.
(5)	The drive is started by the Start rising edge in Mode 1.
(6)	Start can be canceled if the Start_ACK bit has been set.
(7)	Mode 1 is ended. The drive is run down to standstill.
(8)	Mode 1 is ended. The drive is motionless.
(9)	Mode 2 is selected.
(10)	Mode 2 has been accepted by the module.
(11)	The drive is started by the Start rising edge in Mode 1.
(12)	Start can be canceled if the Start_ACK bit has been set.

2.1.2.11.1.4 Positioning Mode

The Positioning mode is possibly only when the Mailbox is deactivated.

First, the mode must be activated using M_Positioning. The Positioning mode is active when the M_Positioning_ACK bit is set. Then, the following setpoints can be specified:

- Velocity,
- Acceleration,
- Absolute position.

The permissible velocity range is 1 … 25000. A velocity equal to 0, velocities greater than 25000 and negative velocities are not permitted and will result in an error message.

The permissible acceleration range is 1 … 32767. An acceleration equal to 0 and negative acceleration are not permitted and will result in an error message.

The setpoints are not accepted until a rising edge of Start. The drive starts up, or frequency is output, directly after the setpoints are accepted. The bit On_Target is canceled immediately and is not reset until the final position has been reached.

The Busy bit is set from the time when the setpoints are accepted until the target is reached.

New setpoints can be activated during the run by a new rising edge of Start. Movement is made toward the new position immediately at the new velocity and acceleration. Movement toward the old position is not continued (on-the-fly change). This function also allows you to change only the velocity or acceleration during an ongoing process.

Attention
With linear $\sin^2$ acceleration, the "on the fly" set point value can only be taken over when the drive is running at v = const.With constant acceleration the new set up value is taken over during the ramps.

The special features associated with limit switches and the Jog and Referencing modes are described in the corresponding sections.

2.1.2.11.1.4.1 Positioning Process Image

The Positioning process image represents the standard configuration for stepper positioning control and is shown in the following tables.

Off set	Input Data		Output Data	
0	S0	Status byte S0	C0	Control byte C0
1	Reserved		Reserved	
2	D0	Actual Velocity L	D0	Velocity L
3	D1	Actual Velocity H	D1	Velocity H
4	D2	Reserved	D2	Acceleration L
5	D3	Reserved	D3	Acceleration H
6	D4	Actual position L	D4	Target position L
7	D5	Actual position M	D5	Target position M
8	D6	Actual position H	D6	Target position H
9	S3	Status byte S3	C3	Control byte C3
10	S2	Status byte S2	C2	Control byte C2
11	S1	Status byte S1	C1	Control byte C1

Control byte C1							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
M_Drive ByMBX	M_Jog	M_Refer ence	M_Progr am	M_Positi oning	Start	Stop2_N	Enable

Enable	Module enable 0: The module is blocked. When this bit is reset during ongoing operation, frequency output is immediately set to 0. This bit terminates the current operating mode. 1: The module is enabled and can be started when the corresponding return message is also available in the status.
Stop2_N	Stopping of drive. This bit can be used to stop the drive from the control system. This bit must be set to activate an operating mode. The return message is transmitted via the Stop_N_ACK bit. 0: Current is being supplied to the motor, but it is at standstill. If the motor is still turning it is put into standstill by the STOP acceleration command Acceleration_Stop_Fast. The motor current can be raised using the parameter Current_Ratio_Stop. The motor can not be started up. 1: The drive may be started.
Start	Startup of drive. The drives, or frequency output, is started in the selected mode on a positive edge. If the edge is not accepted (in the Jog or Mailbox mode), an error message is generated. 0→1: The drive is started accordingly on the rising edge. Positioning: The specified setpoints have been accepted from the process image. Movement is made directly to the new target position, even if the drive is already turning. A previously calculated movement sequence is started immediately when the PreCalc_ACK bit is set (on the fly). Move program: The current Move program is started by the first command for the Move program. If a Move program is already running, it will be restarted at the first command. Referencing: The reference run is started. Jog Mode: No effect. The drive is started in the Jog mode only when the pushbutton Direction_Pos or Direction_Neg is actuated. An error message is generated. Mailbox Mode: May not be set when the mailbox is active Otherwise, an error message will be generated. Various commands can be issued via the mailbox as soon as the Mailbox mode is activated.
M_Positioning	Positioning Mode The mailbox may not be active in this mode. 0: The Positioning mode is not active (selected). 1: The Positioning mode has been selected.
M_Program	Move program mode 0: The Move program mode is not active (selected). 1: The Move program mode has been selected.
M_Reference	Referencing mode. 0: The Referencing mode is not active (selected). 1: The Referencing mode is selected.
M_Jog	Jog mode The drive can be run manually at the setup speed when the Jog mode is active. Control is implemented using Direction_Pos and Direction_Neg. 0: The Jog mode is not active (selected). 1: The Jog mode has been selected.
M_DriveByMBX	Move commands via mailbox mode. In this mode, all movement commands are issued directly via the mailbox. 0: The Move commands via mailbox mode is not active (selected). 1: The Move commands via mailbox mode is active (selected).

Status byte S1							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
M_DriveByMBX_ACK	M_Jog_ACK	M_Reference_ACK	M_Program_ACK	M_Positioning_ACK	Start_ACK	Stop_N_ACK	Ready

Ready
Ready for operation
0: The module is not ready for operation. Either a corresponding request is present via Enable, or an error has resulted in cancellation of Ready. When the bit switches from 1 to 0 the output stage is deactivated, or the output frequency is set to 0.
1: Readiness for operation has been requested via Enable and no error is present.

Stop_N_ACK
Acknowledge request bit Stop2_N.
0: The control system has reset the request bit Stop2_N
or the Enable input has a (1) signal
or the drive is motionless. The drive can not be started up using the Start control bit in this status.
1: The control system has set the request bit Stop2_N
and the Enable input has a (1) signal or the drive is being braked.

Start_ACK
Start sequence in the operating mode.
0: This bit is also set to 0 when the Start request is canceled.
1: The rising edge function is a function of the selected operating mode.

Positioning	The specified setpoints have been accepted from the process image. Movement is made directly to the new target position, even if the drive is already turning. When PreCalc_ACK is set, the movement sequence has already been precalculated and will not be started immediately (on the fly).
Move program:	The Move program is started. If a Move program is already running, it will be restarted at the first command.
Referencing	The reference run is started. If the reference run is still in operation, the (new) setpoints are again accepted and calculated (same procedure as for positioning). The reference run is then restarted.
Jog Mode	No effect. Handshake not performed. The drive is started using the pushbutton Direction_Pos or Direction_Neg.
Mailbox Mode	No effect. Handshake not performed. Various commands can be issued via the mailbox as soon as the Mailbox mode is activated.

M_Positioning_ACK
Positioning mode active
0: The Positioning mode is not active (selected).
1: The Positioning mode has been selected. Movement is made to the active setpoint on the next rising edge for Start.

M_Program_ACK
Move program mode active
0: The Move program mode is not active (selected).
1: The Move program mode has been selected. The Move program is started with the first command on the next rising edge for Start.

M_Reference_ACK
Referencing mode active.
0: The Referencing mode is not active (selected).
1: The Referencing mode is selected. The drive is started at the setup speed on the next rising edge for Start.

M_Jog_ACK
Jog mode active
0: The Jog mode is not active (selected).
1: The Jog mode has been selected.

M_DriveByMBX_ACK
Move commands via mailbox mode active.
In this mode, all movement commands are selected directly via the mailbox.
0: The Move commands via mailbox mode is not active (selected).
1: The Move commands via mailbox mode is active (selected).

Control byte C2							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Error_Quit	PreCalc	X	X	Acc_Range_Sel		Freq_Range_Sel	

Freq_Range_Sel	Select frequency prescaler. The prescaler Frq_Prescaler can be set for frequency using these two bits when the module is to be operated without configuration via the mailbox. These values are accepted only when Enable is set to 0. '00': The Freq_Prescaler prescaler is loaded with the parameter Freq_Div from the current configuration data set. Freq_Div <> 0: Freq_Prescaler = Freq_Div, f_{max} = 2 MHz / Freq_Div Freq_Div = 0: Freq_Prescaler = 200, f_{max} = 10 kHz '01': Freq_Prescaler = 80 f_{max} = 25 kHz '10': Freq_Prescaler = 20 f_{max} = 100 kHz '11': Freq_Prescaler = 4 f_{max} = 500 kHz
Acc_Range_Sel	Select acceleration factor. These two bits are used to set the Acc_Multiplier factor for acceleration. These values are accepted only when Enable is set to 0. '00': The factor Acc_Multiplier is loaded with the parameter Acc_Fact from the current configuration data set. Acc_Fact <> 0: Acc_Multiplier = Acc_Fact Acc_Fact = 0: Acc_Multiplier 8, T = 7600ms '01': Acc_Multiplier = 80 T = 760 ms '10': Acc_Multiplier = 800 T = 76 ms '11': Acc_Multiplier = 8000 T = 7.6 ms
PreCalc	Precalculation for movement sequence The setpoints are taken from the process image and, where required, a movement sequence precalculated. 0: Each setpoint that is transmitted via cyclic telegram traffic must be accepted and processed. Any precalculated movement sequence is rejected. A movement sequence can be calculated and started using Start. 1: The setpoints from the cyclic telegram traffic are to be ignored and the setpoint saved for the 0→1 edge used instead. If the starting speed is zero, a movement sequence will be calculated in advance using this setpoint; this sequence can then be started with the normal delay using Start.
Error_Quit	Acknowledge error. All errors that are present are acknowledged at the rising edge from 0 to 1. After acknowledgement, the error switches to 0, or a new error is present:
X	Reserved

Status byte S2							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Error	PreCalc_ ACK	Referenc e_OK	Direction	On_Spee d	StandStill	Busy	On_Targ et

On_Target	Target reached. The significance of this bit depends on the selected operating mode. 0: A new mode will be selected, or a movement made to a new position. 1: Positioning: The defined position has been reached. Move program: The current Move program has been concluded successfully. Referencing: The reference point has been moved to and set successfully. Jog Mode: The bit is not used in this mode and remains at 0. Mailbox mode: Function of mailbox command.
Busy	Move command being executed. The selected mode is active and a task has been started; the drive is rotating, or frequency output is not equal to 0. 0: The mode has been ended. Operation is completed successfully only when the On_Target bit is set. 1: Positioning: Movement being made toward specified position. Move program: The current Move program is being executed. Referencing: Movement made toward reference point. Jog Mode: The drive has been started up using the pushbutton and is rotating. Mailbox mode: Function of mailbox command.
StandStill	Drive at standstill, or frequency output at 0. 0: Motor is turning. 1: Motor at standstill.
On_Speed	Running speed achieved. 0: The drive has not reached its setpoint speed. 1: The drive has reached its setpoint speed.
Direction	Direction of rotation. This bit is valid only when StandStill is 0. 0: Drive moving in the negative direction. 1: Drive moving in the positive direction.
Reference_OK	Referencing OK Set when reference run has been successfully concluded 0: A reference run has been started. 1: The reference point has been successfully located in the reference run mode.
PreCalc_ACK	Status; precalculation for movement sequence concluded This bit acknowledges the request for a precalculation using PreCalc. 0: Precalculation not yet completed, or no request received. 1: Precalculation completed.
Error	Common error for module An error can/must be acknowledged using Error_Quit. 0: No error present for the drive. 1: Error present for the drive.

Control byte C3							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Reset_Quit	SetupSpeed_Active	LimitSwitch_Neg	LimitSwitch_Pos	Direction_Neg	Direction_Pos	0	0

Direction_Pos	Move in positive direction. This bit is required for the Jog mode. When this mode has been selected, the drive is controlled using these bits. In the Reference run mode this bit defines that the reference switch be searched for in a positive direction. 0: Drive not to move in a positive direction. 1: Drive should move in a positive direction. The drive is deactivated when the bit Direction_Neg is set at the same time.
Direction_Neg	Move in negative direction. This bit is required for the Jog mode. When this mode has been selected, the drive is controlled using these bits. In the Reference run mode this bit defines that the reference switch be searched for in a negative direction. 0: Drive not to move in a negative direction. 1: Drive should move in a negative direction. The drive is deactivated when the bit Direction_Pos is set at the same time.
LimitSwitch_Pos	Limit switch moving range limit in positive direction. This bit is linked to the internal bus. 0: The positive direction limit switch is not actuated. 1: The positive direction limit switch is actuated. The drive is being run down.
LimitSwitch_Neg	Limit switch moving range limit in negative direction. This bit is linked to the internal bus. 0: The negative direction limit switch is not actuated. 1: The negative direction limit switch is actuated. The drive is being run down.
SetupSpeed_Active	Velocity limited to setup speed in all modes. When the bit SetupSpeed_Active_ACK is set the drive speed is limited to the defined setup speed. 0: Limiting not active 1: Limiting active
Reset_Quit	Reset acknowledgement. A Power-on reset or a warm start of the module can be detected by the control system with the Reset status bit; this must be acknowledged using Reset_Quit. This also occurs after saving the user configuration to the EEPROM. Volatile data, parameters and tables for the module may be inconsistent and must be reloaded to ensure proper operation. 0: Function not defined. 1: The Reset signal is reset.
0	Reserved

Status byte S3							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Reset	SetupSpeed_Active_ACK	X	X	X	X	Input2	Input1

Input1	Status for Input 1 In the default setting, input DI1 is linked with the motor shutdown circuit. Shutdown can be performed via DI1 or through the control system. 0: Current is being supplied to the motor, but it is at standstill. If the motor is still turning it is put into standstill by the STOP acceleration command. The motor can not be started up. This is signaled via bit Stop_N_ACK. 1: The drive may be started.
Input2	Status for Input 2 Input DI2 is used as the reference input in the default settings. 0: The reference switch is not actuated. 1: The reference switch is actuated.
SetupSpeed_Active_ACK	Setup speed active. Velocity limited to setup speed in all modes. Acceleration is not limited. The currently valid acceleration value is accepted. 0: Limiting not active 1: Limiting active The drive speed is limited to the parameterized setup speed.
Reset	Module has performed a reset. A module reset can be detected by the controller with this bit. The bit is set after a reset and is confirmed and deleted by Reset_Quit. 0: No reset since last confirmation. 1: A reset has been carried out but not yet confirmed with Reset_Quit. A Power-on reset or a warm start of the module can be detected by the control system with the Reset status bit; this must be acknowledged using Reset_Quit. This also occurs after saving the user configuration to the EEPROM. Volatile data, parameters and tables for the module may be inconsistent and must be reloaded to ensure proper operation.
X	Reserved

2.1.2.11.1.4.2 Sequence Diagram for Positioning

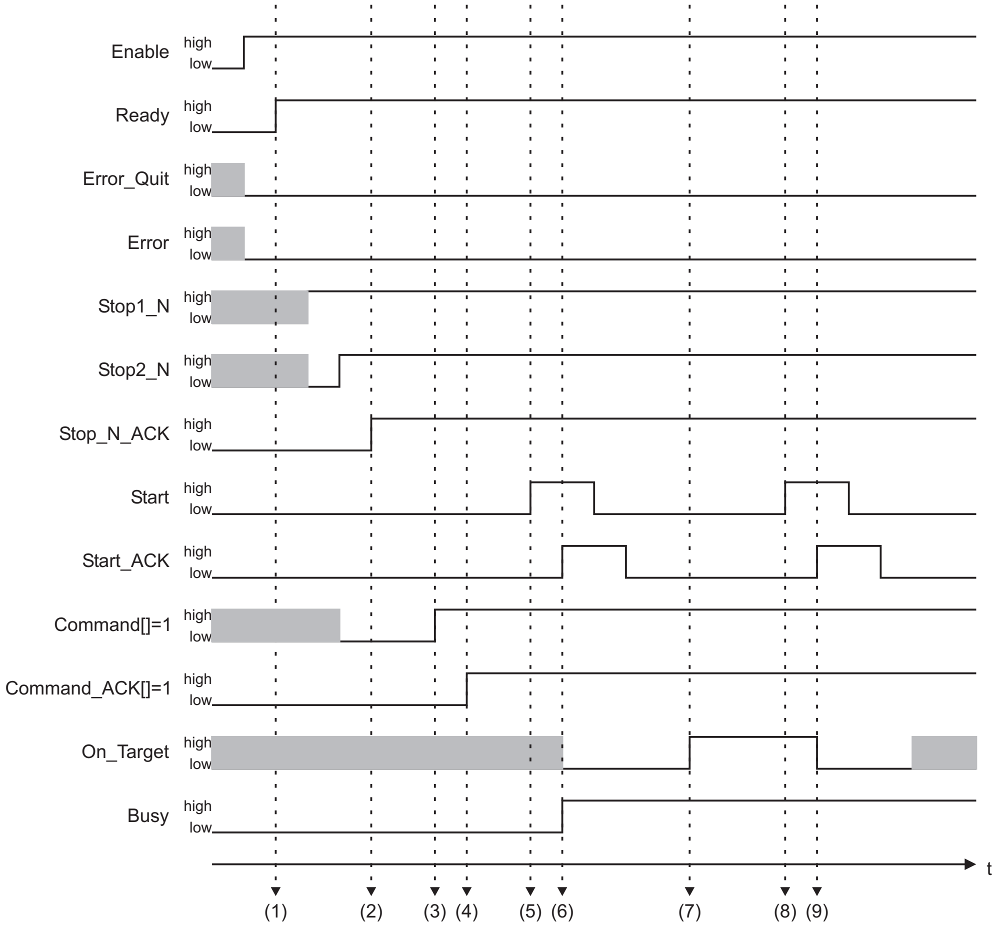


Fig. 2.1.2-20: Sequence diagram for positioning g067x21x

(1)	The module is ready for operation when no error is present after setting of Enable.
(2)	An operating mode can be selected when Stop_N_ACK has been set.
(3)	Positioning mode is selected.
(4)	The Positioning mode has been accepted by the module.
(5)	The drive is started by the Start rising edge.
(6)	The setpoint from the process image is accepted and movement made toward the target position. Start can be canceled after Start_ACK has been set.
(7)	The drive has reached its target position.
(8)	The drive is restarted by the Start rising edge.
(9)	The current setpoint from the process image is accepted, the patch recalculated and movement made toward the target position, where applicable, on the fly. Start can be canceled after Start_ACK has been set.

2.1.2.11.1.5 Referencing Mode

First, the mode must be activated using M_Reference. The Referencing mode is active when the M_Reference_ACK bit is set.

The bit Direction_Neg must also be set if the reference run is to be started in a negative direction, or the Direction_Pos bit set when the reference run is to be started in the positive direction. Direction for reference switch and information detailing whether movement is to be made to the reference switch or a limit switch, specified by the Reference_Mode configuration parameter.

If a reference run is started via the mailbox with the move command START_REFERENCING, the starting direction, moving direction for the reference switch and the information specifying whether movement is to be made toward a reference switch or limit switch, are transferred as parameters.

The reference run is always performed at the setup speed SpeedSetup and at the setup acceleration SetupAcceleration.

Note
The referencing speed should be low so as to take the mechanical requirements into account (such as length of limit switch cams, residual travel after final shutdown, etc).

The reference run is started by the Start rising edge. The drive starts up, or frequency is output, directly after the setpoints are accepted. The On_Target bit is canceled immediately and is not set again.

The Busy bit is set from the time of start until the reference run is completed.

If a limit switch is recognized before the reference switch is reached, the reference run is continued in the opposite direction. If a limit switch is again recognized before a reference switch, the reference process is terminated and an error message generated.

If the drive is at a limit switch it will not move further.
In the Mailbox mode the call of the command START_REFERENCING is acknowledged by error message 23.
No error message is issued in the Referencing mode. In this case, the ERR_RANGE_NEG, or ERR_RANGE_POS bit is set and can then be evaluated.

2.1.2.11.1.5.1 Referencing Process Image

The Referencing process image corresponds to that for Positioning, see Chapter 2.1.2.11.1.4.1, „Positioning Process Image“.

2.1.2.11.1.5.2 Sequence Diagram for Referencing

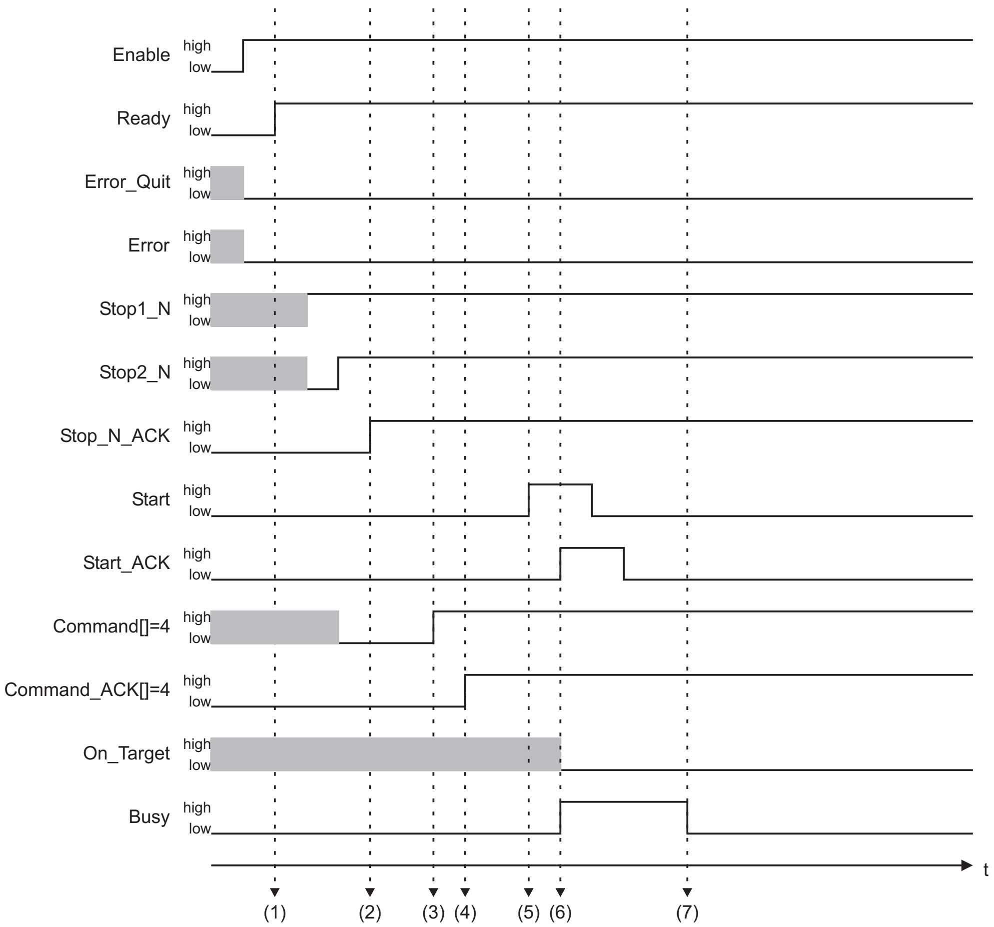


Fig. 2.1.2-21: Sequence diagram for referencing g067x23x

(1)	The module is ready for operation when no error is present after setting of Enable.
(2)	An operating mode can be selected when Stop_N_ACK has been set.
(3)	Referencing mode is selected.
(4)	The Referencing mode has been accepted by the module.
(5)	The drive is started by the Start rising edge.
(6)	The reference run is started. Start can be canceled if Start_ACK has been set.
(7)	The reference point has been moved to and set.

2.1.2.11.1.5.3 Start Parameters for Referencing Mode

Referencing to positive end of reference switch, starting in negative direction

Operating mode				
Referencing, M_Reference = 1		**Mailbox, Command START_ REFERENCING**		**Note**
Reference_Mode, Bit 0	0	Parameter 3 Bit 0	0	Referencing to reference switch
Direction_Pos	0	Parameter 3 Bit 1	0	Start in negative direction
Direction_Neg	1			
Reference_Mode, Bit 1	1	Parameter 3 Bit 2	1	Referencing to positive end

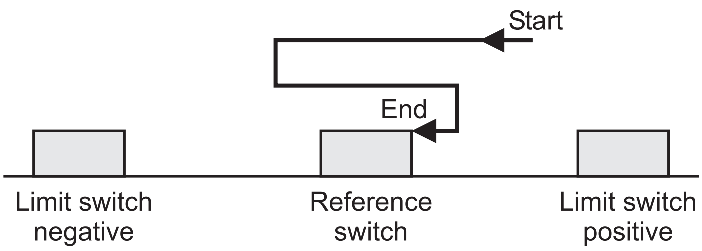


Fig. 2.1.2-22: Referencing to positive end of reference switch, with start in negative direction from positive movement range g067x10e

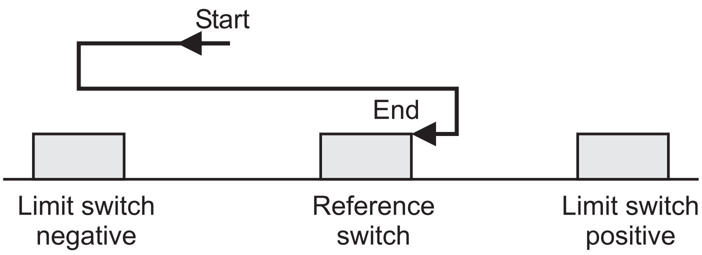


Fig. 2.1.2-23: Referencing to positive end of reference switch, with start in negative direction from negative movement range g067x14e

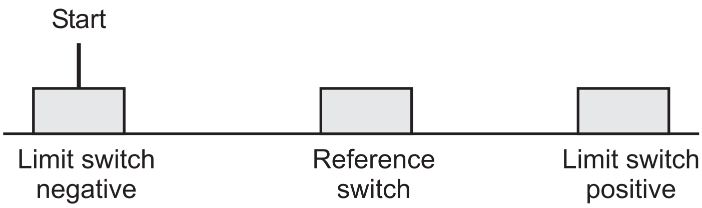


Fig. 2.1.2-24: Referencing to positive end of reference switch with start in negative direction from limit switch g067x13e

Note
The drive is not started on a start from the limit switch. In the Mailbox mode the error message 23 is generated on calling of command START_REFERENCING. No error message is generated in the Referencing mode. The Bit ERR_RANGE_POS or ERR_RANGE_NEG bit is set!

Referencing to negative end of reference switch, starting in negative direction

Operating mode				Note
Referencing, M_Reference = 1		**Mailbox, Command START_ REFERENCING**		
Reference_Mode , Bit 0	0	Parameter 3 Bit 0	0	Referencing to reference switch
Direction_Pos	0	Parameter 3 Bit 1	0	Start in negative direction
Direction_Neg	1			
Reference_Mode , Bit 1	0	Parameter 3 Bit 2	0	Referencing to negative end

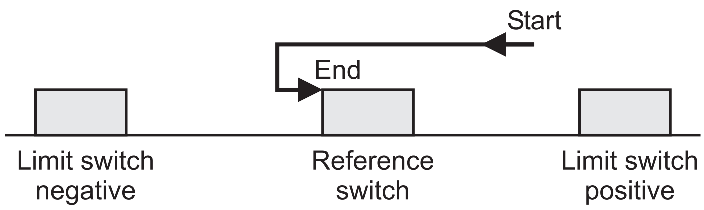


Fig. 2.1.2-25: Referencing at the negative end of reference switch with start in negative direction from positive movement range g067x11e

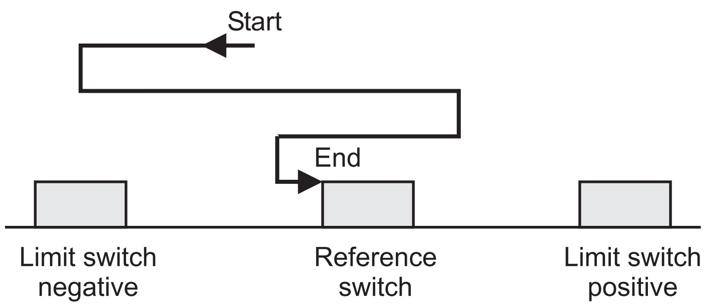


Fig. 2.1.2-26: Referencing at the negative end of reference switch with start in negative direction from negative movement range g067x12e

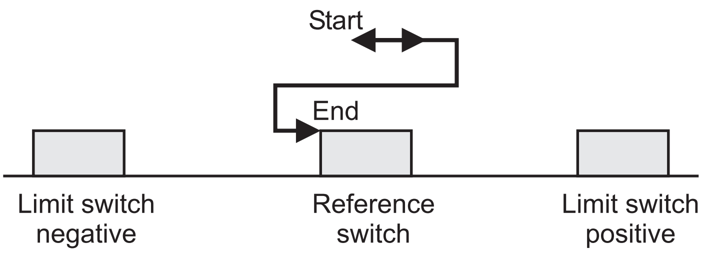


Fig. 2.1.2-27: Referencing at negative end of reference switch with start in negative direction from reference switch g067x15e

Note:
On a start from the reference switch, the starting direction is reversed by the module to first ensure that the reference switch is free.

Referencing to positive end of reference switch, starting in positive direction

Operating mode				**Note**
Referencing, M_Reference = 1		**Mailbox, Command START_ REFERENCING**		
Reference_Mode , Bit 0	0	Parameter 3 Bit 0	0	Referencing to reference switch
Direction_Pos	1	Parameter 3 Bit 1	1	Start in positive direction
Direction_Neg	0			
Reference_Mode , Bit 1	1	Parameter 3 Bit 2	1	Referencing to positive end

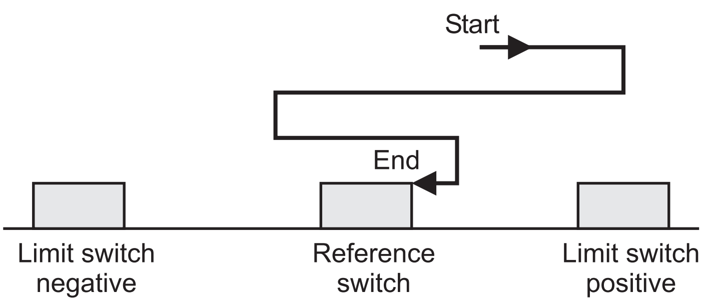


Fig. 2.1.2-28: Refderencing at positive end of reference switch with start in positive direction from positive movement range g067x16e

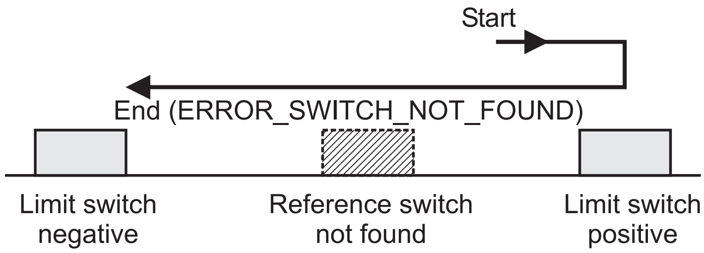


Fig. 2.1.2-29: Referencing to positive end of reference switch with start in positive direction from positive movement range, reference switch not found g067x38e

Referencing of limit switch with start in negative direction

Operating mode				Note
Referencing, M_Reference = 1		**Mailbox, Command START_ REFERENCING**		
Reference_Mode , Bit 0	1	Parameter 3 Bit 0	0	Referencing to limit switch
Direction_Pos	0	Parameter 3 Bit 1	0	Start in negative direction
Direction_Neg	1			
Reference_Mode , Bit 1	x			

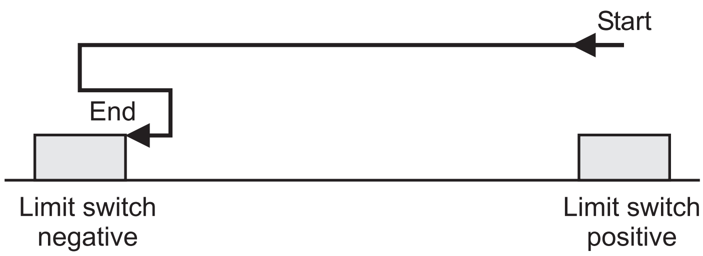


Fig. 2.1.2-30: Referencing to limit switch with start in negative direction from positive limit switch g067x39e

Fig. 2.1.2-31: Referencing to negative limit switch with start in negative direction from positive movement range g067x40e

Fig. 2.1.2-32: Referencing to limit switch with start in negative direction from negative limit switch g067x15e

2.1.2.11.1.6 Jog and Stepping Mode

The drive can be run manually at the defined setup speed when the Jog mode is active. Control is implemented via Direction_Pos or Direction_Neg. The two control bits are locked against each other. The run is ended when the set timeout period (stepping mode) expires via the process image. A timeout of zero allows unlimited movement, as long as Direction_Pos or Direction_Neg is set (Jog mode).

If the setup speed is parameterized as zero, the Jog mode is run at speed 1.

If movement is made to a limit switch during the Jog mode, the drive will stop. After that, movement can only be made away from the limit switch.

The drive stops when it leaves the movement range defined by the parameter Drive_Range_Neg and Drive_Range_Pos. The drive can then be operated outside the movement range using a repeated JOG command.

2.1.2.11.1.6.1 Jog and Stepping Mode Process Image

This process image is different from the standard configuration for stepper positioning control and is shown in the table below.

Off set	Input Data		Output Data	
0	S0	Status byte S0	C0	Control byte C0
1	Reserved		Reserved	
2	D0	Current velocity (LSB)	D0	Reserved
3	D1	Current velocity (MSB)	D1	Reserved
4	D2	Reserved	D2	Timeout (LSB)
5	D3	Reserved	D3	Timeout (MSB)
6	D4	Current position (LSB)	D4	Reserved
7	D5	Current position	D5	Reserved
8	D6	Current position (MSB)	D6	Reserved
9	S3	Status byte S3	C3	Control byte C3
10	S2	Status byte S2	C2	Control byte C2
11	S1	Status byte S1	C1	Control byte C1

The function of the control and status bytes corresponds to the standard configuration for stepper positioning control given in Chapter 2.1.2.11.1.4.1, “Positioning Process Image”.

2.1.2.11.1.6.2 Sequence Diagram for Jog and Stepping Mode

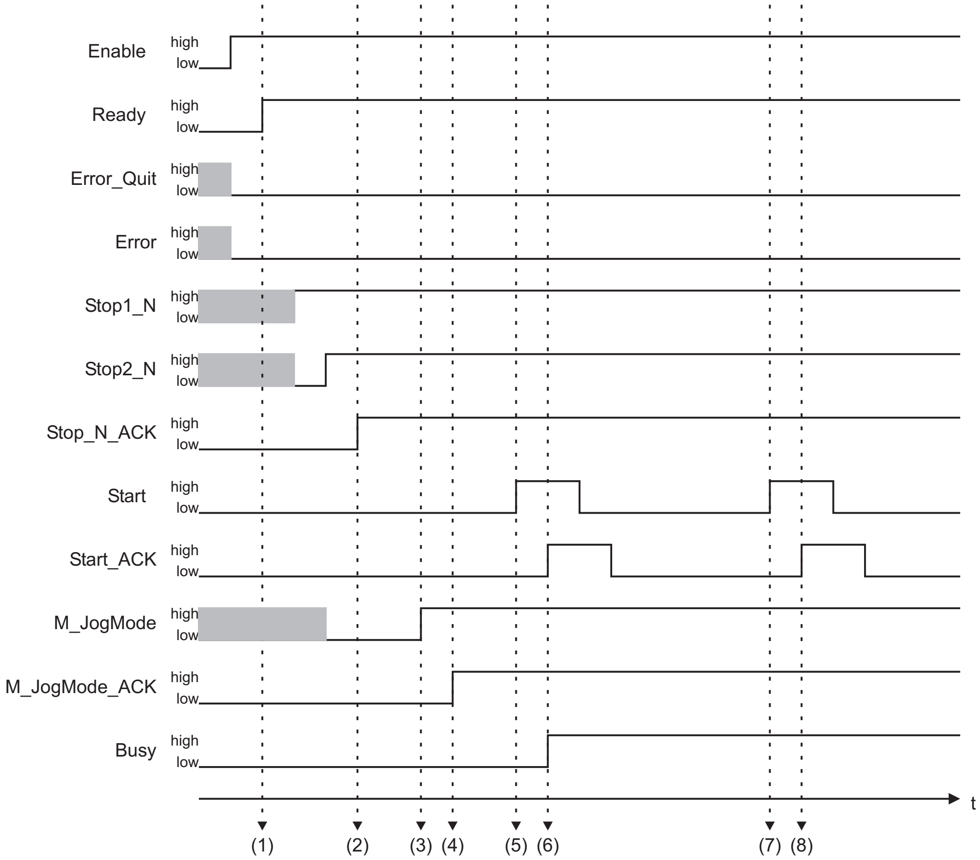


Fig. 2.1.2-33: Sequence diagram, JogMode g067x24x

(1)	The module is ready for operation when no error is present after setting of Enable.
(2)	An operating mode can be selected when Stop_N_ACK has been set.
(3)	The Jog mode is selected.
(4)	The Jog mode has been accepted by the module.
(5)	The drive is activated by the Start rising edge.
(6)	The Jog mode is activated, the drive can be started using the pushbutton Direction_Pos and Direction_Neg. Start can be canceled if Start_ACK has been set.
(7)	The drive is restarted by the Start rising edge.
(8)	The Jog mode is activated, the drive can be started using the pushbutton Direction_Pos and Direction_Neg. Start can be canceled if Start_ACK has been set.

2.1.2.11.1.7 Move program mode

A Move program can be downloaded to the I/O module via the mailbox.

The available commands for this are given in the section 3.2, „Commands for Move Mode“.

First, the mode must be activated using M_Program. The Move program mode is active when the M_Program_ACK bit is set.

The Move program is started on a rising edge.

The address for the first command to be executed is given via the process image.

The Program_Running bit is set from the time of start to the end of the program. This bit can be queried using the mailbox command GET_BIT.

The On_Target and Busy bits are controlled by the individual program commands.

When the bit SetupSpeed_Active_ACK is set at the same time for setup, speed is limited to the defined setup speed.

2.1.2.11.1.7.1 Move Program Process Image

This process image is different from the standard configuration for stepper positioning control and is shown in the table below.

Off set	Input Data		Output Data	
0	S0	Status byte S0	C0	Control byte C0
1	Reserved		Reserved	
2	D0	Current velocity (LSB)	D0	Reserved
3	D1	Current velocity (MSB)	D1	Reserved
4	D2	Current value for command counter (LSB)	D2	Starting value for command counter (LSB)
5	D3	Current value for command counter (MSB)	D3	Starting value for command counter (MSB)
6	D4	Current position (LSB)	D4	Reserved
7	D5	Current position	D5	Reserved
8	D6	Current position (MSB)	D6	Reserved
9	S3	Status byte S3	C3	Control byte C3
10	S2	Status byte S2	C2	Control byte C2
11	S1	Status byte S1	C1	Control byte C1

The function of the control and status bytes corresponds to the standard configuration for stepper positioning control given in Chapter 2.1.2.11.1.4.1, “Positioning Process Image”.

2.1.2.11.1.7.2 Sequence Diagram for Move Program

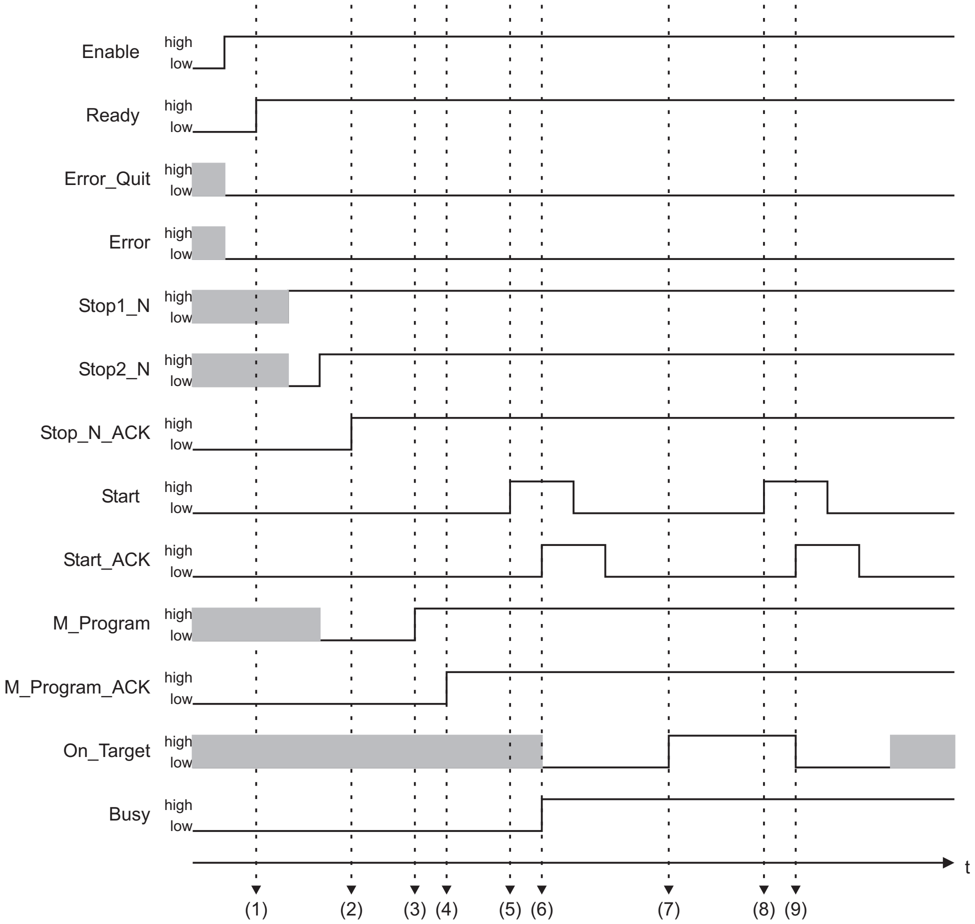


Fig. 2.1.2-34: Sequence diagram, Move program g067x22x

(1)	The module is ready for operation when no error is present after setting of Enable.
(2)	An operating mode can be selected when Stop_N_ACK has been set.
(3)	The Move program mode is selected.
(4)	The Move program mode has been accepted by the module.
(5)	The drive is started by the Start rising edge.
(6)	The Move program is started on the first command. Start can be canceled if Start_ACK has been set.
(7)	The current Move program has reached its last position.
(8)	The drive is restarted by the Start rising edge.
(9)	The Move program in progress will be terminated and the drive set to standstill. The Move program is then restarted on the first command. Start can be canceled if Start_ACK has been set.

2.1.2.11.1.7.3 Example of Move Program

In this example of a Move program, first the velocity (20000) and then the acceleration (3000) is set. After this, a "True" signal is anticipated at Input 1 on line 2. If the signal is "True", movement is made to Position 1 (specified position = 65065) and then the system waits for a "False" signal at input 1 on line 4. If the "False" signal is received, movement is made to Position 2 (specified position = 0). The system then returns to line 2 of the Move program, i.e. it waits for a "True" signal at Input 1.

Line in the Move table	Opcode	Data (LSB)	Data	Data (MSB)	Meaning
	MB2	MB3	MB4	MB5	
0	0x25	0x20	0x4E	0x00	Set velocity to 20000
1	0x22	0xB8	0x0B	0x00	Set acceleration to 3000
2	0x71	0x30	0x01	0x00	Query, Input 1 = "1"
3	0x02	0x20	0xFE	0x00	Move to position 65065
4	0x71	0x30	0x00	0x00	Query, Input 1 = "0"
5	0x02	0x00	0x00	0x00	Move to position 0
6	0xF5	0x02	0x00	0x00	Go to line 2
7	0x70	0xD0	0x07	0x00	Wait 2000 ms

2.1.2.11.1.7.4 Autostart

An existing Move program is started automatically after a module reset when the configuration bit HwSwConfig.Program_AutoStart (Bit 7) is set.

Prerequisites for successful automatic start:

- Valid Move program present in EEPROM,
- The HwSwConfig.Program_AutoStart bit is set in the configuration,
- The Rest status bit is set and has not yet been canceled,
- The Ready status bit is set,
- The Stop_N_ACK status bit is set.

A Move program started by HwSwConfig.Program_AutoStart can be ended by canceling Reset. The bit is reset by Reset_Quit.

A Move program is started only one time after each reset by HwSwConfig.Program_AutoStart.

A Move program started with HwSwConfig.Program_AutoStart is always started at an address of 0.

2.1.2.11.2 Move Mode via Mailbox

The mailbox must first be displayed. This is described in Chapter 2.1.2.8, “Mailbox Mode”.

After that, the Move commands via mailbox mode must be activated. This is accomplished by setting bit 7 in the control byte C1.

Only then can Move commands be specified.

2.1.2.11.2.1 Move Commands

The module can be operated via the mailbox using the move commands. Movement can be made directly to different positions. This command is accepted only when the mode Move mode via mailbox has been activated.

The available commands for move find you in the appendix in chapter 3.2, “Commands for Move Mode”.

2.1.2.11.3 Limiting of Moving Range

2.1.2.11.3.1 Hardware Limit Switch

The hardware limit switches are active in the Positioning and Velocity control applications. These devices limit the movement path.

Any allocation of limits switches to the direction of movement must be maintained. The LimitSwitch_Neg hardware limit switch restricts the movement range to smaller positions, i.e. in the negative direction. The LimitSwitch_Pos hardware limit switch restricts the movement range to larger positions, i.e. in the positive direction.

Positioning

If movement is made to a limit switch in the Positioning mode, the drive will brake the movement until standstill using the defined deceleration Acceleration_Stop_Fast.
The drive can only be started in the Jog and Referencing modes when it is located at a limit switch.

Jog Mode

If movement is made to a limit switch in the Jog mode, the drive will brake the movement until standstill using the defined deceleration Acceleration_Stop_Fast.
The drive can then be moved away from the limit switch by pressing the "Jog" button Direction_Neg or Direction_Pos again; from the positive limit switch in a negative direction and from the negative limit switch in a positive direction.
The drive will brake movement again until standstill using the defined deceleration Acceleration_Stop_Fast as soon as it moves away from the limit switch. The drive is then no longer located at the limit switch and can be run in any mode without any restrictions.

Referencing

Attention
During the reference run, limiting of the moving range is not evaluated by the software limit switch! This may result in damage to the system if proper functioning of the hardware limit switch is not ensured!

If the drive is located at a limit switch in the Referencing mode to a reference switch, it can only be started in the Jog or Referencing mode. Only a negative direction of movement is possible from the positive limit switch and vice versa.

If a reference run has been made to a limit switch in the Referencing mode, the drive will end up at the limit switch and a special operating mode will be activated.
In this special mode the drive can be moved away from the limit switch in any arbitrary mode, with the positive limit switch only permitting movement in a negative direction and the negative limit switch in a positive direction. The special mode is terminated 100 ms after the drive leaves the limit switch.

2.1.2.11.3.2 Software Limit Switch

The permissible movement range of the drive is limited by the hardware limit switch. Options are also available, however, for restricting the permissible movement range using limits that can be parameterized (software limit switches), for example if no hardware limit switches are available.

Note
Evaluation of the hardware limit switches has priority over evaluation of the software limit switches.

The software limit switches are defined by the limits Drive_Range_Neg and Drive_Range_Pos in the Configuration table. The limit Drive_Range_Neg restricts the range to smaller positions, i.e. in the negative direction, while Drive_Range_Pos restricts the range for larger positions, i.e. in the positive direction.

The software limit switches are only active in the Positioning and Move program via mailbox modes, as well as some subfunctions being available in the Jog mode. The switches are not evaluated in other modes.

Attention
During a reference run, limiting of the moving range is not evaluated by the software limit switch! This may result in damage to the system if proper functioning of the hardware limit switch is not ensured!

The limits Drive_Range_Neg and Drive_Range_Pos define the permissible range of movement. If one of these defined limits is violated, the associated bit is set.
The default setting is Drive_Range_Neg = 0x800001 and Drive_Range_Pos = 0x7FFFFF.

If movement is made beyond a defined movement range, the drive is brought to standstill using the defined deceleration Acceleration_Stop_Fast and, after that, only those directions of movement accepted that move the unit back into the permissible range; the exception here is the Jog mode.

In the Jog mode the drive is brought to a standstill each time it attempts to move out of the permissible range. In this mode the drive can also be operated outside the movement range defined by the software limit switches with repeated JOG commands. The software limit switches are not active again until the drive is back within the defined range.

2.1.2.12 Expanded Positioning Functions

2.1.2.12.1 Rotary Axis

The "Rotary axis" function is activated by the parameter Rotary_Axis_Period being written with a value other than zero. If the Rotary_Axis_Period parameter is zero, a linear and limited movement range is assumed.

The position is repeated with a rotary axis every 2π or 360°. The Rotary_Axis_Period parameter indicates how many motor steps correspond to one rotation around the axis by 2π or 360.

The actual value for rotary axis is always within the range 0 … Rotary_Axis_Period. This ensures that no internal overrun occurs with relative motion repeated any number of times.

Parameter	Linear axis	Rotary axis	
Rotary_Axis_Period	0	Microsteps per revolution > 0	
Working range	Drive_Range_Neg … Drive_Range_Pos	Limited to Drive_Range_Neg … Drive_Range_Pos when Drive_Range_Neg >=0 or Drive_Range_Pos < Rotary_Axis_Period, otherwise unrestricted	
Actual value	Working range	0 … Rotary_Axis_Period, periodic	
Setpoint value	Working range	Absolute Positioning: 0 … Rotary_Axis_Period	Relative Positioning: -8388607 … 8388607. Can be repeated any number of times. No internal overrun

The setpoint setting distinguishes between absolute and relative positioning.

2.1.2.12.1.1 Relative Positioning

The target position is added to the current position for relative positioning. The "Rotary axis" mode is initially ignored during calculation of the position; as a result, the "virtual target" may lie outside the range 0 … 2π. This allows relative positioning to be performed over several revolutions. The actual value, however, is reported only within the range 0 … 2π; the number of completed revolutions can not be determined.

The direction of movement depends on the sign for the relative setpoint.

2.1.2.12.1.2 Absolute Positioning

The target position is always within the range 0 … 2π for absolute positioning. A setpoint defined outside of this range will result in an error.

On a movement task from standstill, the system determines in what direction the target can be reached in the shortest time.

For a positioning movement at a starting velocity, or target velocity, the direction of movement that requires no, or the fewest, changes in direction will be selected.

Absolute positioning permits braking at a certain velocity setting to a precisely defined spot (e.g. coil end that is to be stopped exactly at an attitude angle of 0 from full speed, only possible via mailbox!).

2.1.2.12.2 Camshaft

The camshaft provides pulses as a function of position for nine (9) channels CAM1 … CAM9. Up to 50 switching positions can be freely defined through channels 1 … 8. Channel CAM9, on the other hand, supplies a periodic signal as a function of position.

Parameterization of channels 1 ... 8 is performed using a table containing 50 entries. Each entry consists of a position xp (24-bit) and a bit sample (8-bit). The bit samples each describe the position of the eight output channels CAM1 … CAM8, which is valid starting from the assigned position xp up to the next larger position entry xp+1.

The table entries are arranged according to ascending positions.

The bit sample for the first entry is output for lesser entries.

The configuration assigns Channel 9 the starting position, the cam width and the repeat cycle.

The activated camshaft is always active, independent of the Move mode, with the exception of the reset status.

In contrast to other setpoints, the switching positions for the camshaft are always given in "microsteps". Conversion from or to other user-specific units is not provided for.

The Camshaft table can not be edited in the module, but must be downloaded completely.

Example: Eight (8) entries are to be loaded to the camshaft table.

Step No.	CAM								Position			Outputs
	8	7	6	5	4	3	2	1	Byte 4 MB5	Byte 3 MB4	Byte 2 MB3	Byte 1 MB 3
1	1	0	0	0	0	1	0	0	0x00	0x2e	0xe0	0x84
2	0	1	0	0	0	1	0	0	0x00	0x5d	0xc0	0x44
3	0	0	1	0	0	0	1	0	0x00	0xbb	0x80	0x22
4	0	0	0	1	0	0	1	0	0x01	0x77	0x00	0x12
5	0	0	0	0	1	0	0	0	0x01	0xd4	0xc0	0x08
6	0	1	0	0	0	1	0	0	0x02	0x90	0x40	0x44
7	0	1	0	0	0	0	1	0	0x07	0xb0	0xc0	0x42
8	0	1	0	0	0	0	0	1	0x07	0xdf	0xa0	0x41

The portion of the table highlighted in gray must be downloaded to the module.

The checksum for the table must also be calculated. To do this, the sum is taken over all the bytes. Here, the checksum is 0xb0d.

Opcodes given in Chapter 3.1.4.4, „Table Management Commands“ are required for downloading.

The corresponding opcodes have been adapted in the following table for this case. For the complete scope and explanations, refer to the sections cited above.

2.1.2.12.3 Position Table

The position table enables a set Move program sequence containing variable positions to be run.

Up to 50 positions can be stored in the position table that can be queried by Move commands. The position table entries can be evaluated on the basis of absolute or relative Move commands.

The position table can be downloaded. As an alternative to this, individual entries can be written or "taught" as absolute or relative positions with the current position.

2.1.2.12.3.1 Teaching of Positions

Teaching of positions using the mailbox command POS_TABLE_TEACH allows the current actual value to the saved in the position table so that it is available as a target for a Move task.

The current actual value can be saved as the reference point for relative travel measurement.

The soft limit switches Drive_Range_Pos and Drive_Range_Neg can also be taught.

2.1.2.12.4 Control of a Motor Brake

Control of the motor brake is conducted using the Brake bit (see Chapter 3.4, „Bit field for I/O driver“).
This bit is only available as an internal bit, as this module is not equipped with digital outputs. Nevertheless, reconfiguration can be performed to redirect this bit to a bit in the input process image that is not needed.

Control of this bit is performed from two OR-linked sources.
On the one hand, the Brake bit is set automatically as soon as the drive is running and is canceled as soon as the drive is at standstill. Defining of automatic control is performed using the configuration parameters Braketime_Turn_On and Braketime_Turn_Off. The Brake bit is then activated directly after the start of a move command. If the bit has not been set, execution of the move command will, however, be delayed by the Braketime_Turn_On time. The configuration parameter Braketime_Turn_Off defines the deactivation time for the Brake bit. This bit is deactivated before the target is reached by the Braketime_Turn_Off time. The brake can be controlled directly with this bit. The brake is released when the bit is set, and is applied when the bit is canceled.

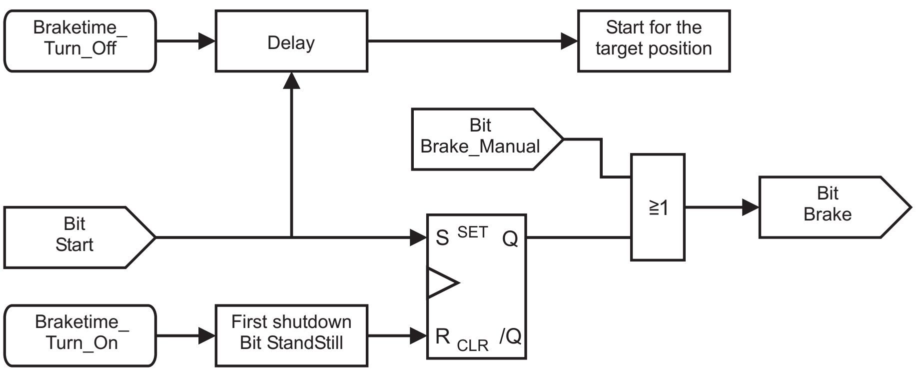


Fig. 2.1.2-35: Control of motor brake g067x17e

As an alternative, the brake can also be operated independently using the Brake_Manual bit (see Chapter 3.4, „Bit field for I/O driver“). This bit can be set and canceled externally and can also, for example, be linked to a camshaft channel so that it is switched as a function of position.

2.1.2.13 Other Applications

Other applications can be configured by modifying the stepper positioning control system.

2.1.2.13.1 Speed Control

The Frequency/Speed Control application represents a variant of Stepper control. All the functions for Stepper control can still be utilized. The basic difference is the modified process image which permits frequency and speed to be specified. Speed corresponds directly to the output frequency and the frequency ramps to acceleration.

With this module the stepper motor is operated at the specified speed.

Frequency generation is provided with one output channel and a cycle signal, along with a sign signal.

The output may also be switched to an "incremental encoder simulation", for outputting two square-wave signals offset by 90°.

The frequency/speed control system generates a definable output frequency. As with Step positioning, this frequency can be specified using the process image, the Jog mode, the Move task via mailbox and by the Program mode.

The frequency/speed control application is only feasible with the function "Rotary shaft". If the rotary shaft configuration value does not have a parameter of zero, a value of 100000 is assumed.

The frequency/speed control application is selected using the configuration parameters Application_Selector = 2 and PWM Period = 0.

All of the Stepper control functions can still be utilized. The basic difference is that the Positioning/Stepper control mode is replaced with a Speed set mode. Merely the speed and acceleration setting are evaluated for this in the process image.

Position acquisition is executed in the background in the frequency/speed control application. To prevent this from triggering a shutdown via the soft limit switches with Drive_Range_Pos or Drive_Range_Neg, the parameters Rotary_Axis_Period are used and parameters provided or simulated for a rotary axis. If parameters have been provided for the value Rotary_Axis_Period, this value is used for the rotary axis.

Step positioning/Stepper control contains a detailed description of this (selection of mode using M_Positioning and accepting of setpoints with Start).

The frequency/speed control application essentially influences speed interpretation in the process image and the evaluation of Rotary_Axis_Period. Otherwise, the complete functions of other modes, such as Jog mode, Referencing, Move task via mailbox and Program mode can be utilized.

2.1.2.13.1.1 Velocity Control Process Image

This process image is different from the standard configuration for stepper positioning control and is shown in the tables below.

Off set	Input Data		Output Data	
0	S0	Status byte S0	C0	Control byte C0
1	Reserved		Reserved	
2	D0	Actual Velocity L	D0	Velocity L
3	D1	Actual Velocity H	D1	Velocity H
4	D2	Reserved	D2	Acceleration L
5	D3	Reserved	D3	Acceleration H
6	D4	Actual position L	D4	Reserved
7	D5	Actual position M	D5	Reserved
8	D6	Actual position H	D6	Reserved
9	S3	Status byte S3	C3	Control byte C3
10	S2	Status byte S2	C2	Control byte C2
11	S1	Status byte S1	C1	Control byte C1

The function of the bits in control bytes C1 ... C3 and in status bytes S1 … S3 are determined by the Frequency/Speed control application. When switchover is made to this application, the linked locations for the old application are retained. The meaning of the bits for the standard configuration are explained below.

Control byte C1							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
M_Drive ByMBX	M_Jog	M_Refer ence	M_Progr am	M_Speed Control	Start	Stop2_N	Enable

Enable — Module enable
- 0: The module is blocked. When this bit is reset during ongoing operation, the power output stage is deactivated.. This bit terminates the current operating mode.
- 1: The module is enabled and can be started when the corresponding return message is also available in the status.

Stop2_N — Shutdown of drive.
This bit can be used to deactivate the drive from the control system.
This bit must be set to activate an operating mode.
The return message is transmitted via the Stop_N_ACK bit.
- 0: Current is being supplied to the motor, but it is at standstill. If the motor is still turning it is put into standstill by the STOP acceleration command. The motor can not be started up.
- 1: The drive may be started.

Start — The drive, or frequency output, is started in the selected mode on a positive edge. An error message is generated if the edge is not accepted.
- 0→1: The drive is started accordingly on the rising edge.
  Positioning is conducted to the current setpoint given in the process image.
  Movement is made directly to the new target velocity, even if the drive is already turning. A previously calculated movement sequence is started immediately when the PreCalc_ACK bit is set (on the fly).

M_SpeedControl — Frequency/Speed control mode
The mailbox may not be active in this mode.
- 0: The Frequency/Speed mode is not active (selected).
- 1: The Frequency/Speed mode is active (selected).

M_Program — Move program mode
- 0: The Move program mode is not active (selected).
- 1: The Move program mode has been selected.

M_Reference — Referencing mode.
- 0: The Referencing mode is not active (selected).
- 1: The Referencing mode is selected.

M_Jog — Jog mode
The drive can be run manually at the setup speed when the Jog mode is active. Control is implemented using Direction_Pos and Direction_Neg.
- 0: The Jog mode is not active (selected).
- 1: The Jog mode has been selected.

M_DriveByMBX — Move commands via mailbox mode.
In this mode, all movement commands are issued directly via the mailbox.
- 0: The Move commands via mailbox mode is not active (selected).
- 1: The Move commands via mailbox mode is active (selected).

Status byte S1							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
M_DriveByMBX_ACK	M_Jog_ACK	M_Reference_ACK	M_Program_ACK	M_SpeedControl_ACK	Start_ACK	Stop_N_ACK	Ready

Ready	Ready for operation 0: The module is not ready for operation. Either a corresponding request is present via Enable, or an error has resulted in cancellation of Ready. When the bit switches from 1 to 0 the output stage is deactivated, or the output frequency is set to 0. 1: Readiness for operation has been requested via Enable and no error is present.
Stop_N_ACK	Status of drive, or acknowledgement of request bit Stop_N 0: The control system has reset the request bit Stop2_N or the Enable input has no (1) signal or the generator is not in operation. Start can not be used for startup in this status. 1: The control system has set the request bit Stop2_N and the Enable input has a (1) signal or the drive is being braked.
Start_ACK	Start sequence in the operating mode. 0: This bit is also set to 0 when the Start request is canceled. 1: The specified setpoints have been accepted from the process image. Movement is made directly to the new target position, even if the drive is already turning. When PreCalc_ACK is set, the movement sequence has already been precalculated and will not be started immediately (on the fly).
M_SpeedControl_ACK	Acknowledge Frequency/Speed control mode 0: The Frequency/Speed mode is not active (selected). 1: The Frequency/Speed mode is active (selected). Frequency output is started on the next rising edge.
M_Program_ACK	Move program mode active 0: The Move program mode is not active (selected). 1: The Move program mode has been selected. The Move program is started with the first command on the next rising edge for Start.
M_Reference_ACK	Referencing mode active. 0: The Referencing mode is not active (selected). 1: The Referencing mode is selected. The drive is started at the setup speed on the next rising edge for Start.
M_Jog_ACK	Jog mode active 0: The Jog mode is not active (selected). 1: The Jog mode has been selected.
M_DriveByMBX_ACK	Move commands via mailbox mode active. In this mode, all movement commands are selected directly via the mailbox. 0: The Move commands via mailbox mode is not active (selected). 1: The Move commands via mailbox mode is active (selected).

Control byte C2							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Error_Quit	PreCalc	0	0	Acc_Range_Sel		Freq_Range_Sel	

Freq_Range_Sel	Select frequency prescaler. The prescaler Freq_Prescaler can be set for the velocity setting using these two bits when the module is to be operated via the mailbox without configuration. The values will only be accepted when Enable is set to 0. '00': The Freq_Prescaler prescaler is loaded with the parameter Freq_Div from the current configuration data set. '01': Freq_Prescaler = 80 Fmax = 25 kHz '10': Freq_Prescaler = 20 Fmax = 100 kHz '11': Freq_Prescaler = 4 Fmax = 500 kHz
Acc_Range_Sel	Select acceleration factor. These two bits are used to set the Acc_Multiplier factor for acceleration. These values are accepted only when Enable is set to 0. '00': The factor Acc_Multiplier is loaded with the parameter Acc_Fact from the current configuration data set. '01': Acc_Multiplier = 80 T = 760 ms '10': Acc_Multiplier = 800 T = 76 ms '11': Acc_Multiplier = 8000 T = 7.6 ms
PreCalc	Precalculation for movement sequence The setpoints are taken from the process image and, where required, a movement sequence precalculated. 0: Each setpoint that is transmitted via cyclic telegram traffic must be accepted and processed. Any precalculated movement sequence is rejected. A movement sequence can be calculated and started using Start. 1: The setpoints from the cyclic telegram traffic are to be ignored and the setpoint saved for the 0→1 edge used instead. If the starting speed is zero, a movement sequence will be calculated in advance using this setpoint; this sequence can then be started with the normal delay using Start.
Error_Quit	Acknowledge error. All errors that are present are acknowledged at the rising edge from 0 to 1. After acknowledgement, the error switches to 0, or a new error is present:
0	Reserved

Status byte S2							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Error	PreCalc_ACK	X	Direction	On_Speed	StandStill	Busy	X

Busy	Specified speed not yet reached. 0: Specified speed reached. 1: Specified speed not yet reached.
StandStill	Drive at standstill, or frequency output at 0. 0: Motor is turning. 1: Motor at standstill.
On_Speed	Running speed achieved. 0: The drive has not reached its setpoint speed. 1: The drive has reached its setpoint speed.
Direction	Direction of rotation. This bit is valid only when StandStill is 0. 0: Drive moving in the negative direction. 1: Drive moving in the positive direction.
PreCalc_ACK	Status for precalculation for movement sequence This bit acknowledges the request for a precalculation using PreCalc. 0: Precalculation not yet completed, or no request received. 1: Precalculation completed.
Error	Drive error status. An error can be acknowledged using Error_Quit. 0: No error present for the drive. 1: Error present for the drive.
X	Reserved

Control byte C3							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Reset_Quit	SetupSpeed_Active	LimitSwitch_Neg	LimitSwitch_Pos	0	0	0	0

LimitSwitch_Pos	Limit switch input on movement in positive direction. This bit is linked to the internal bus. 0: The positive direction limit switch is not actuated. 1: The positive direction limit switch is actuated. The drive is being run down.
LimitSwitch_Neg	Limit switch input on movement in negative direction. This bit is linked to the internal bus. 0: The negative direction limit switch is not actuated. 1: The negative direction limit switch is actuated. The drive is being run down.
SetupSpeed_Active	Velocity limited to setup speed in all modes. When the bit SetupSpeed_Active_ACK is set the drive speed is limited to the defined setup speed. 0: Limiting not active 1: Limiting active
Reset_Quit	Reset acknowledgement 0: Function not defined. 1: The Reset signal is reset.
0	Reserved

Status byte S3							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Reset	SetupSpeed_Active_ACK	X	X	X	X	X	Input1

Input1	In the default setting, input DI1 is linked with the motor shutdown circuit. Shutdown can be performed via DI1 or through the control system. 0: Current is being supplied to the motor, but it is at standstill. If the motor is still turning it is put into standstill by the STOP acceleration command. The motor can not be started up. This is signaled via bit Stop_N_ACK. 1: The drive may be started.
SetupSpeed_Active_ACK	Velocity limited to setup speed in all modes. 0: Limiting not active 1: Limiting active The drive speed is limited to the parameterized setup speed.
Reset	A module reset can be detected by the controller with this bit. The bit is set after a reset and is confirmed and deleted by Reset_Quit. 0: No reset since last confirmation. 1: A reset has been carried out but not yet confirmed with Reset_Quit. Volatile data, parameters and tables for the module may be inconsistent and must be reloaded.
X	Reserved

2.1.2.14 Advanced Diagnostics

The diagnostics commands allow internal module information to be accessed. This includes:

- Error status of the device,
- Variables and status bits,
- Password,
- Configuration table and
- Position table.

The commands are elucidated in the appendix in chapter 3.1.4.5, „Diagnostics Commands“.

2.1.2.14.1 Internal Status Variables

The module is provided with internal status variable that can be read out using the mailbox command DIAG_RD_VAR. These variables can also be acquired automatically by a data recorder.

The variable number determines the source to be read from.

Variable number	Source
0… 0x1000:	defined variables are read (see also 3.6 "Internal Status Variables")
0x1000…0x1100:	defined bits 0 … 0x100 are read (see also 3.4 "Bit field for I/O driver")

The status variables are elucidated in the appendix in section 3.6, „Internal Status Variables“.

2.1.2.14.2 Data Recorder

The data recorder allows two internal variable to be recorded in a definable time frame for later analysis. 500 values are recorded each time.

The configuration values Trace_Var1 and Trace_Var2 contain the index for the variables to be recorded (see "Internal Status Variables"). The configuration value Trace_MsecCycleTime denotes the scan (cycle) time in ms.

Configuration value	Meaning
Trace_Var1	Index for first recording variable
Trace_Var2	Index for second recording variable
Trace_MsecCycleTime	Scan time in ms

The Trace_Stored bit indicates that a complete data set has been recorded.

A 0→1 edge of Trace_Trigger initiates recording when the Trace_Armed bit is set.

A traced (recorded) data set can be read out using an upload command from the table manager (see Table Manager).

Bits	deleted	set
Trace_Stored	Data set not yet available	a data set has been saved
Trace_Trigger	0→ 1 edge starts recording	
Trace_Armed	Triggering is blocked; an existing data set is not overwritten	Triggering active; recording is started by the next triggering event

The internal bits (see Chapter 3.4, „Bit field for I/O driver“) can be used as triggering sources. This is accomplished by entering the corresponding link in the configuration table. The control system can also initiate recording using the mailbox commands GET_BIT and SET_BIT. The trigger bits must be linked to the MONE internal bit for this.

2.1.2.15 Connection Example

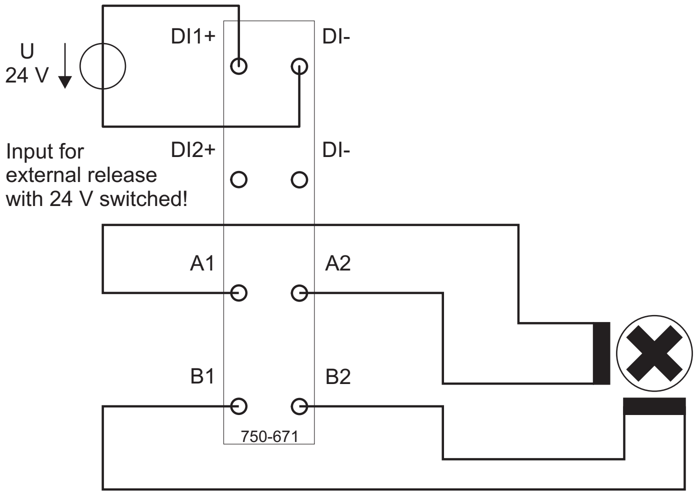


Fig. 2.1.2-36: Connection of bipolar stepper motors g067010e

3 Appendix

3.1 Mailbox Commands

3.1.1 Overview of Mailbox Commands

Function	Opcode	Meaning	Page
General commands			
IDLE	0x00	No task	124
Drive commands			
DRIVE_COMMAND	0x40	Command for Move mode	125
Download command			
DLD_START	0x41	Download Start	126
DLD_CONT	0x42	Download Continue	129
DLD_END	0x43	Completion of download	132
Table management commands			
TABLE_ERASE	0x44	Tables will be deleted.	133
TABLE_COPY	0x45	Tables will be copied.	135
TABLE_START	0x46	Table is activated	138
TABLE_STOP	0x48	Ends table processing	139
TABLE_GET_ACTIVE	0x4F	Determine active table	140
Diagnostics commands			
DIAG_RD_ERROR	0x49	Information about error retrieved from error memory	141
DIAG_QUIT_ERROR	0x4A	Terminates a device error condition	142
DIAG_RD_VAR	0x4C	Read out internal variable	143
DIAG_RD_BIT	0x4D	Read out internal bit	144
DIAG_QUERY_ STORAGE	0x4E	Read out storage process status bit	145
Configuration table commands			
CONFIG_SET_PTR	0x50	Set address for data access to the configuration	146
CONFIG_WR	0x51	Write access to configuration value	147
CONFIG_RD	0x52	Read access to configuration value	148
CONFIG_SAVE	0x53	Saves the current RAM configuration	149
CONFIG_RESTORE	0x54	Restores the configuration	150

Function	Opcode	Meaning	Page
Position table commands			
POS_TABLE_CREATE	0x5C	Generates a position table in the RAM.	152
POS_TABLE_SET_PTR	0x5D	Sets an index for the subsequent entry to be written with POS_TABLE_WR in the position table	153
POS_TABLE_WR	0x5E	Writes an entry to the active position table	154
POS_TABLE_TEACH	0x5F	Writes the current position to the active position table	155

3.1.2 Overview of Mailbox Commands, Sorted by Opcodes

Function	Opcode	Meaning	Page
IDLE	0x00	No task	124
DRIVE_COMMAND	0x40	Command for Move mode	125
DLD_START	0x41	Download Start	126
DLD_CONT	0x42	Download Continue	129
DLD_END	0x43	Completion of download ...	132
TABLE_ERASE	0x44	Tables being deleted	133
TABLE_COPY	0x45	Tables being copied ...	135
TABLE_START	0x46	Activates a table	138
TABLE_STOP	0x48	Ends table processing	139
DIAG_RD_ERROR	0x49	Information about error retrieved from error memory	141
DIAG_QUIT_ERROR	0x4A	Terminates a device error condition	142
DIAG_RD_VAR	0x4C	Read out internal variable	143
DIAG_RD_BIT	0x4D	Read out internal bit	144
DIAG_QUERY_ STORAGE	0x4E	Read out storage process status bit	145
TABLE_GET_ACTIVE	0x4F	Determine active table	140
CONFIG_SET_PTR	0x50	Set address for data access to the configuration	146
CONFIG_WR	0x51	Write access to configuration value	147
CONFIG_RD	0x52	Read access to configuration value	148
CONFIG_SAVE	0x53	Saves the current RAM configuration	149
CONFIG_RESTORE	0x54	Restores the configuration	150
POS_TABLE_CREATE	0x5C	Generates a position table in the RAM.	152
POS_TABLE_SET_PTR	0x5D	Sets an index for the subsequent entry to be written with POS_TABLE_WR in the position table	153
POS_TABLE_WR	0x5E	Writes an entry to the active position table	154
POS_TABLE_TEACH	0x5F	Writes the current position to the active position table	155

3.1.3 Overview of Mailbox Commands, Sorted by Functions

Function	Opcode	Meaning	Page
CONFIG_RD	0x52	Read access to configuration value	148
CONFIG_RESTORE	0x54	Restores the configuration	150
CONFIG_SAVE	0x53	Saves the current RAM configuration	149
CONFIG_SET_PTR	0x50	Set address for data access to the configuration	146
CONFIG_WR	0x51	Write access to configuration value	147
DIAG_QUERY_ STORAGE	0x4E	Read out storage process status bit	145
DIAG_QUIT_ERROR	0x4A	Terminates a device error condition	142
DIAG_RD_BIT	0x4D	Read out internal bit	144
DIAG_RD_ERROR	0x49	Information about error retrieved from error memory	141
DIAG_RD_VAR	0x4C	Read out internal variable	143
DLD_CONT	0x42	Download Continue	129
DLD_END	0x43	Completion of download ...	132
DLD_START	0x41	Download Start	126
DRIVE_COMMAND	0x40	Command for Move mode	125
IDLE	0x00	No task	124
POS_TABLE_CREATE	0x5C	Generates a position table in the RAM.	152
POS_TABLE_SET_PTR	0x5D	Sets an index for the subsequent entry to be written with POS_TABLE_WR in the position table	153
POS_TABLE_TEACH	0x5F	Writes the current position to the active position table	155
POS_TABLE_WR	0x5E	Writes an entry to the active position table	154
TABLE_COPY	0x45	Tables being copied ...	135
TABLE_ERASE	0x44	Tables being deleted	133
TABLE_GET_ACTIVE	0x4F	Determine active table	140
TABLE_START	0x46	Activates a table	138
TABLE_STOP	0x48	Ends table processing	139

3.1.4 Reference Commands – Mailbox Commands

3.1.4.1 General commands

3.1.4.1.1 IDLE (0x00)

No task is performed if the value for "Opcode" is 0.

Request								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x00**							
MB1	T	-						
MB2	Reserved							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Response								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x00**							
MB1	T	Return Code						
MB2	Reserved							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Return Code	0x00:	OK
	0x01:	General error

3.1.4.2 Move Commands

3.1.4.2.1 DRIVE_COMMAND (0x40)

The module can be operated via the mailbox using the move commands. Movement can be made directly to different positions. This command is accepted only when the mode "Move task via mailbox"(chapter **Fehler! Verweisquelle konnte nicht gefunden werden.**, „**Fehler! Verweisquelle konnte nicht gefunden werden.**“) has been activated.

The commands available for the Move mode are described in chapter 3.2, “3.2”.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	Command							
MB3	Data 1							
MB4	Data 2							
MB5	Data 3							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	Command							
MB3	Data 1							
MB4	Data 2							
MB5	Data 3							

Return Code	0x00:	OK
	0x01:	General error
	0x11:	The last command is still being executed
	0x12:	Command not accepted, for example, when a Move command has not yet been completed.
	0x13:	Unknown command
	0x23:	Access denied

3.1.4.3 Download Commands

3.1.4.3.1 DLD_START (0x41)

Download Start

Tables are always loaded into RAM (1 or 2) first. The cursor is first placed on the first entry. Only one table can be loaded at any one time; any previous, incomplete download is canceled and becomes invalid. Direct transfer to / from the EEPROM is not possible (see also TABLE_COPY). Download to the same RAM sector is rejected with an error message when a move program table is still active. Camshaft and position tables can also be overwritten when they are active.

Default assignment

The default assignment for moving curve tables is PROG_END (0x00). Camshaft tables have the default assignment 0x80000000 (invalid position). The default assignment for position tables is 0.

Configuration tables

The EEPROM version number is expected in byte 5 during download of a configuration table. A complete table, with 128 data values, 32 bit each, is always expected.

Download formats (see also Request Data 4)

Expanded 32-bit down-/upload

A DLD_CONT must be used for an 8-bit command / data sample and for a 32-bit data entry / position entry in the table when downloading a Move program / a camshaft table. MB 4…6 are ignored for an 8-bit command / data sample.

Compressed 24-bit down-/upload

When downloading a Move program / a camshaft table, those items are transferred with a DLD_CONT command for an 8-bit command / data sample and a 24-bit date entry / position entry.

Maximum number of data sets (see also Request, Byte 5 and 6)

Table	Typ e	Max. number of data sets
Move program:	1	400
Camshaft	2	50
Position table	3	50
Configuration	4	128
Trace	5	1000

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x41**							
MB1	T	-						
MB2	Storage location							
MB3	Table type							
MB4	Number of data (LSB)							
MB5	Transfer		Number of data (MSB)					

Storage location	Table type 1, 2, 3	0:	Reserved
		1:	RAM Table 1
		2:	RAM Table 2
		3 ... 255:	Reserved
	Table type 4, 5	0:	Reserved
		1:	RAM Table 1
		2 ... 255:	Reserved

Table type	0:	Reserved
	1:	Move program:
	2:	Camshaft
	3:	Position table
	4:	Configuration data set, User configuration
	5:	Trace
Transfer	0:	24-bit data download
	1:	24-bit data upload
	2:	32-bit data download
	3:	32-bit data upload
	4 ... 255:	Reserved

Response								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x41**							
MB1	T	Return Code						
MB2	Status							
MB3	Reserved							
MB4	Number of data sets							
MB5	EEPROM version number							

Return Code	0x00:	OK
	0x30:	Table being used
	0x31:	General error
Status	0:	Download/Upload can be started
	1:	Error; Download/Upload not possible

3.1.4.3.2 DLD_CONT (0x42)

Download Continue

An entry is written to the selected table. The cursor is then moved to the next element. The request data are ignored for an upload. An error is returned when it is detected during a download that a transmitted Move table command is invalid, or if camshaft entries are transmitted NOT in ascending order. The data that has been transmitted will not be corrected. The table can not be valid with DLD_END however.

Download formats:

Expanded 32-bit down-/upload

A DLD_CONT must be used for an 8-bit command / data sample and for a 32-bit data entry / position entry in the table when downloading a Move program / a camshaft table. MB 4 … 6 are ignored for an 8-bit command / data sample.

Move program table (Type 01)

Step	MB2	MB3	MB4	MB5
1.1	Command 1	Reserved	Reserved	Reserved
1.2	Data 1 (LSB)	Data 1	Data 1	Data 1 (MSB)
2.1	Command 2	Reserved	Reserved	Reserved
2.2	Data 2 (LSB)	Data 2	Data 2	Data 2 (MSB)
…				

Camshaft Table (Type 02)

Step	MB2	MB3	MB4	MB5
1.1	Bit sample 1	Reserved	Reserved	Reserved
1.2	Position 1 (LSB)	Position 1	Position 1	Position 1 (MSB)
2.1	Bit sample 2	Reserved	Reserved	Reserved
2.2	Position 2 (LSB)	Position 2	Position 2	Position 2 (MSB)
…				

Compressed 24-bit down-/upload

When downloading a Move program / a camshaft table, those items are transferred with a DLD_CONT command for an 8-bit command / data sample and a 24-bit date entry / position entry:

Move program table (Type 01)

Step	MB2	MB3	MB4	MB5
1	Command 1	Data 1 (LSB)	Data 1	Data 1 (MSB)
2	Command 2	Data 2 (LSB)	Data 2	Data 2 (MSB)
…				

Camshaft Table (Type 02)

Step	MB2	MB3	MB4	MB5
1	Bit sample 1	Position 1 (LSB)	Position 1	Position 1 (MSB)
2	Bit sample 2	Position 2 (LSB)	Position 2	Position 2 (MSB)
…				

Only 32-bit data exists for the position tables (Type 03) and the configuration data set (Type 04). Therefore, only the 32-bit down-/upload are given for both types of tables.

Request								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x42**							
MB1	T	-						
MB2	Data							
MB3	Data							
MB4	Data							
MB5	Data							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x42**							
MB1	T	Return Code						
MB2	Data							
MB3	Data							
MB4	Data							
MB5	Data							

Return Code	0x00:	OK
	0x31:	Upload/Download not started, or all data have already been transferred
	0x38:	Transferred data set corrupt

3.1.4.3.3 DLD_END (0x43)

End of Download

The download is completed and the stepper module checks the checksum. If the checksum is not OK, the table is invalid and can not be activated. The checksum is the sum of all data transferred with DLD_CONT. Summation is performed at 8 bits, with the 4 bytes that were transferred with DLD_CONT each being taken as 8-bit values. The difference between the sum of all transferred data and the checksum must therefore be zero. The request data are ignored for an upload. If a configuration table is transferred, saving to EEPROM is performed automatically (but only when saving has been completed successfully), with a subsequent warm start (even if the transfer was faulted) that re-initializes all software modules.
The Reset status bit is set after the warm start; this must be canceled using Reset_Quit.
Only then is the module operational again.
After a successful download of a Move program to RAM Table 1, that table is automatically activated. (only when no other table is active however, see also TABLE_START)

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x43**							
MB1	T	-						
MB2	Checksum for transferred data (LSB)							
MB3	Checksum for transferred data							
MB4	Checksum for transferred data							
MB5	Checksum for transferred data (MSB)							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x43**							
MB1	T	Return Code						
MB2	Checksum for stored data (LSB)							
MB3	Checksum for stored data							
MB4	Checksum for stored data							
MB5	Checksum for stored data (MSB)							

Return Code	0x00:	OK
	0x31:	General error

3.1.4.4 Table Management Commands

3.1.4.4.1 TABLE_ERASE (0x44)

Tables are deleted by setting their status to invalid. An active table can not be deleted. A table can not be deleted during ongoing transfer using DLD_START, DLD_CONT or DLD_END.

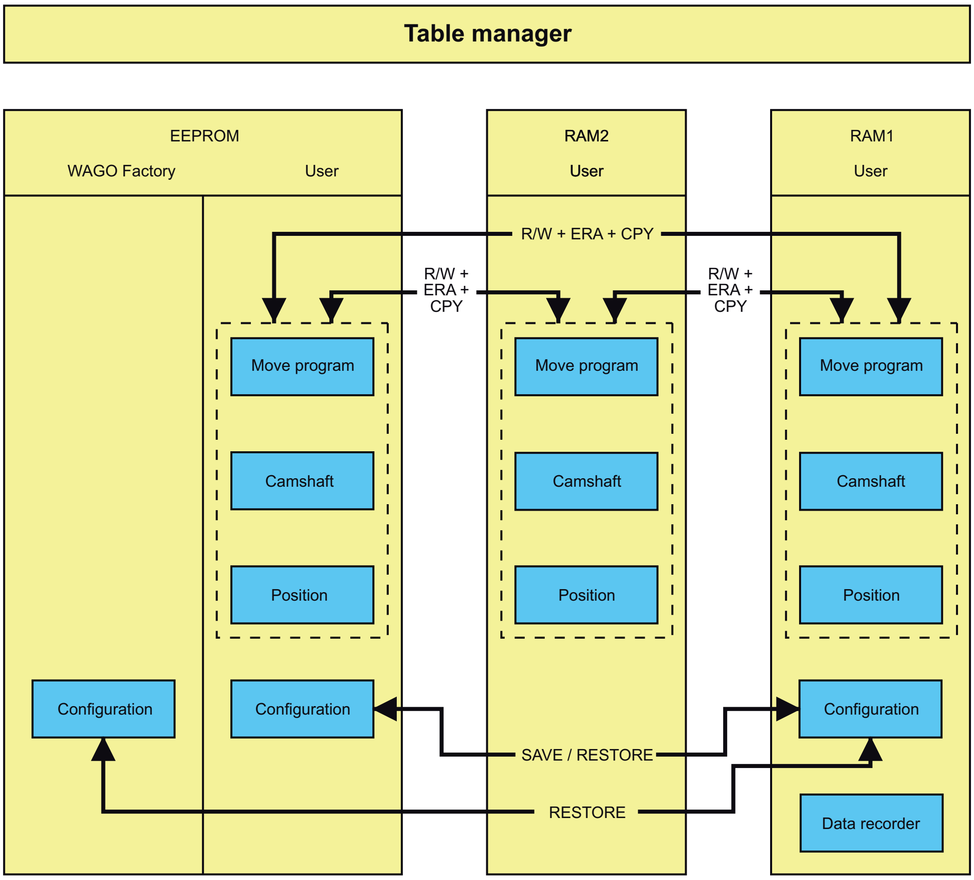
Deleting of an EEPROM table is performed in the background, independently of processing of the table command (see also DIAG_QUERY_STORAGE). The "FACTORY_DEFAULT" configuration contained in the EEPROM can not be deleted (not even when using 255 as byte 2). "FACTORY_DEFAULT" may only be overwritten.

Request								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x44**							
MB1	T	-						
MB2	Storage location							
MB3	Table type							
MB4	Reserved							
MB5	Reserved							

Storage location	0:	EEPROM table
	1:	RAM Table 1
	2:	RAM Table 2
	3 ... 255:	Reserved
Table type	0:	Reserved
	1:	Move program:
	2:	Camshaft
	3:	Position table
	4:	Configuration data set, User configuration
	5 ... 255:	Reserved

Response								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x44**							
MB1	T	Return Code						
MB2	Status							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Return Code	0x00:	OK
	0x30:	Table active
	0x31:	General error
Status	0:	Successfully deleted
	1:	Deleting aborted



R/W Read/Write
ERA Erase
CPY Copy

Fig. 3.1.4-1: Table manager g067120e

3.1.4.4.2 TABLE_COPY (0x45)

Tables will be copied.

The target may not be identical to the source.

A table can not be specified either as the target nor as the source of the copying command when transfer using DLD_START, DLD_CONT or DLD_END has not been completed. Writing of the EEPROM is performed in the background, independent of processing of the table command (see also DIAG_QUERY_STORAGE).

The tables located in the EEPROM are always copied to RAM 1 when the system is started up. This command can not be used for copying out of the EEPROM.

A configuration table can not be copied with this command (see also CFG_SAVE, CONFIG_RESTORE).

Copying options:

1. RAM → RAM

2. RAM → EERPOM

Request								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x44**							
MB1	T	-						
MB2	Table type							
MB3	Data source							
MB4	Storage target							
MB5	Reserved							

Table type	0:	Reserved
	1:	Move program:
	2:	Camshaft
	3:	Position table
	4:	Reserved
	5:	Status query for a previous copying process (see also DIAG_QUERY_STORAGE)
	6 ... 255:	Reserved
Data source	0:	Reserved
	1:	RAM Table 1
	2:	RAM Table 2
	3 ... 255:	Reserved
Storage target	0:	EEPROM table
	1:	RAM Table 1
	2:	RAM Table 2
	3 ... 255:	Reserved

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x44**							
MB1	T	Return Code						
MB2	Status							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Return Code	0x00:	OK
	0x31:	General error
	0x33:	Copying process still active
	0x34:	EEPROM copying process aborted
	0x35:	Target table not empty
Status	0:	Successfully copied
	1:	Copying aborted

3.1.4.4.3 TABLE_START (0x46)

Activates a table Only a valid table can be activated (transfer using DLD_START, DLD_CONT and DLD_END completed successfully and checksum valid). This command can only be used after the Move program has been stopped.

Request								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x46**							
MB1	T	-						
MB2	Storage location							
MB3	Table type							
MB4	Reserved							
MB5	Reserved							

Storage location	0:	No table
	1:	RAM Table 1
	2:	RAM Table 2
	3 ... 255:	Reserved
Table type	0:	Reserved
	1:	Move program:
	2:	Camshaft
	3:	Position table
	4 ... 255:	Reserved

Response								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x46**							
MB1	T	Return Code						
MB2	Status							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Return Code	0x00:	OK
	0x31:	General error
Status	0:	Successfully activated
	1:	Activation aborted

3.1.4.4.4 TABLE_STOP (0x48)

Ends table processing; after this, the STOP_FAST command is executed internally in the system.

Request								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x48**							
MB1	T	-						
MB2	Reserved							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Response								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x48**							
MB1	T	Return Code						
MB2	Status							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Return Code (error code for previous command SPEED_ STOP_ IMM)	0x00:	OK
	0x01:	General error
	0x11:	The last command is still being executed
	0x12:	Command not accepted, for example, when a Move command has not yet been completed.
	0x23:	Access denied
Status	exact error code, when return code <> 0	

3.1.4.4.5 TABLE_GET_ACTIVE (0x4F)

Determine the active table.

Request								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x4F**							
MB1	T	-						
MB2	Table type							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Table type	0:	Reserved
	1:	Move program:
	2:	Camshaft
	3:	Position table
	4:	Configuration data set, User configuration
	5 ... 255:	Reserved

Response								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x4F**							
MB1	T	Return Code						
MB2	Status							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Return Code	0x00:	OK	
	0x31:	Invalid table type	
Status	Table type 1, 2, 3	0:	No table active (even when 0x31 returned)
		1:	RAM Table 1 active
		2:	RAM Table 2 active
		3 … 255:	Reserved
	Table type 4	0:	Reserved (even when 0x31 returned)
		1:	User data set active
		2:	Factory default active
		3 … 255:	Reserved

3.1.4.5 Diagnostics Commands

3.1.4.5.1 DIAG_RD_ERROR (0x49)

Information about error is retrieved from the error memory.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x49**							
MB1	T	-						
MB2	Reserved							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x49**							
MB1	T	Return Code						
MB2	Error code (LSB)							
MB3	Error code (MSB)							
MB4	Extra information (LSB)							
MB5	Extra information (MSB)							

Return Code	0x00:	OK

3.1.4.5.2 DIAG_QUIT_ERROR (0x4A)

Terminates a device error condition.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x4A**							
MB1	T	-						
MB2	Reserved							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x4A**							
MB1	T	Return Code						
MB2	Reserved							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Return Code	0x00:	OK

3.1.4.5.3 DIAG_RD_VAR (0x4C)

Read out status variable. The variable number determines the source to be read from.

Request								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x4C**							
MB1	T	-						
MB2	Variable number							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Variable number	0 … 0x1000:	predefined variables are read (see chapter 3.6, "Internal Status Variables")
	0x1000 … 0x1100:	predefined bits 0 ….. 0x100 are read (see chapter 3.4, "Bit field for I/O driver ")
	0x40000000 … 0x40004000:	Direct reading out of RAM
	0xE0000000 … 0xE0200000:	Direct reading out of controller periphery
	0xFFE00000 … 0xFFFFFFFF:	Direct reading out of controller periphery

Response								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x4C**							
MB1	T	Return Code						
MB2	Variable (LSB)							
MB3	Variable							
MB4	Variable							
MB5	Variable (MSB)							

Return Code	0x00:	OK

3.1.4.5.4 DIAG_RD_BIT (0x4D)

Read out status bit (see also chapter 3.4, “Bit field for I/O driver

”).

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x4D**							
MB1	T	-						
MB2	Bit number							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Bit number 0 ... 255: Specifies which predefined bit is being requested.

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x4D**							
MB1	T	Return Code						
MB2	Status							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Return Code	0x00:	OK
Status	0:	Bit deleted
	1:	Bit set

3.1.4.5.5 DIAG_QUERY_STORAGE (0x4E)

Read out storage process status bit

Request								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x4E**							
MB1	T	-						
MB2	Reserved							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Response								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x4E**							
MB1	T	Return Code						
MB2	Status							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Return Code	0x00:	OK
Status	0:	Storing completed
	1 ... 255:	Storing in progress

3.1.4.6 Configuration Table Commands

3.1.4.6.1 CONFIG_SET_PTR (0x50)

Set address for data access to the configuration, see chapter **Fehler! Verweisquelle konnte nicht gefunden werden.**, "**Fehler! Verweisquelle konnte nicht gefunden werden.**". The specified address is the same as the byte address.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x50**							
MB1	T	-						
MB2	Address (LSB)							
MB3	Address (MSB)							
MB4	Number of bytes							
MB5	Reserved							

Number of bytes	0:	Reserved
	1 ... 4:	Number of bytes that are written for access with Config_WR.
	5 ... 255:	Reserved

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x50**							
MB1	T	Return Code						
MB2	Reserved							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Return Code	0x00:	OK
	0x23:	Access denied; invalid number of bytes or invalid index

3.1.4.6.2 CONFIG_WR (0x51)

Write access to configuration value.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x51**							
MB1	T	-						
MB2	Data (LSB)							
MB3	Data							
MB4	Data							
MB5	Data (MSB)							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x51**							
MB1	T	Return Code						
MB2	Reserved							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Return Code	0x00:	OK
	0x23:	Access denied

3.1.4.6.3 CONFIG_RD (0x52)

Read access to configuration value. The value 0 is returned when invalid access size specified.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x52**							
MB1	T	-						
MB2	Address (LSB)							
MB3	Address (MSB)							
MB4	Number of bytes							
MB5	Reserved							

Number of bytes	0:	Reserved
	1 ... 4:	Number of bytes that are written for access with Config_RD.
	5 ... 255:	Reserved

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x52**							
MB1	T	Return Code						
MB2	Data (LSB)							
MB3	Data							
MB4	Data							
MB5	Data (MSB)							

Return Code	0x00:	OK

3.1.4.6.4 CONFIG_SAVE (0x53)

Saves the current RAM configuration in the EEPROM. The configuration is saved as a user data set in the EEPROM with password 0x0001. The configuration is saved as RACTORY_DEFAULT in the EEPROM with password 0xE17E. At the same time, EEPROM sectors of the module registry set are also saved. A FACTORY_DEFAULT data set that has been saved can never be deleted again, only overwritten. This function does not wait for the saving process to be completed. This can be determined with DIAG_QUERY_STORAGE. Complete activation of the saved data set is conducted only after a (manual) restart of the module.
The Reset status bit is set after the warm start; this must be canceled using Reset_Quit.
Only then is the module operational again.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x53**							
MB1	T	-						
MB2	Password (LSB)							
MB3	Password							
MB4	Password							
MB5	Password (MSB)							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x53**							
MB1	T	Return Code						
MB2	Reserved							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Return Code	0x00:	OK
	0x31:	Fault

3.1.4.6.5 CONFIG_RESTORE (0x54)

The configuration is restored and the user data set overwritten. A warm start is carried out after the command has been successfully executed to ensure that all data is accepted.

Warning
During warm start the mailbox data are undefined. They may be evaluated only again if the status bit Reset signals the end of warm start.

Request								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x54**							
MB1	T	-						
MB2	Restore							
MB3	Warm start							
MB4	Reserved							
MB5	Reserved							

Restore	0:	Reserved
	1:	Last saved user data set loaded from EEPROM
	2:	User data set overwritten with FACTORY_DEFAULT
	3 ... 255:	Reserved
Warm start	0:	User data set overwritten and warm start carried out
	1:	The required data set is only loaded to the RAM, without a warm start being performed. Error 2821 = CFG_FACTORY_LOAD is reported. A warm start is carried out only when this error is acknowledged, with the original configuration being restored.
	2 ... 255:	Reserved

Response								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x54**							
MB1	T	Return Code						
MB2	Reserved							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Return Code	0x00:	OK
	0x31:	Fault

3.1.4.7 Position table commands

3.1.4.7.1 POS_TABLE_CREATE (0x5C)

Generates a position table in the RAM. The table status is set to "valid".

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x5C**							
MB1	T	-						
MB2	Storage location							
MB3	Number of elements							
MB4	Initialization							
MB5	Reserved							

Storage location	0:	Reserved
	1:	RAM Table 1
	2:	RAM Table 2
	3 ... 255:	Reserved
Number of elements	1 … 50:	Number of elements
	51 … 255:	Reserved
Initialization	0:	Install table completely with 0x80000000
	1:	Expand existing table (all existing entries are retained). Is executed only when the new size is larger than the existing table!

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x5C**							
MB1	T	Return Code						
MB2	Status							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Return Code	0x00:	OK
	0x32:	Invalid table specified
	0x3A:	Invalid number of elements
Status	0:	Initialization successful
	1:	Initialization aborted

3.1.4.7.2 POS_TABLE_SET_PTR (0x5D)

Sets an index for the subsequent entry to be written with POS_TABLE_WR in the active position table.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x5D**							
MB1	T	-						
MB2	Index							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Index	0 ... 49:	Index
	50 ... 255:	Reserved

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x5D**							
MB1	T	Return Code						
MB2	Status							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Return Code	0x00:	OK
	0x37:	Table does not exist, or index not assigned
Status	0:	Successfully indexed
	1:	Indexing aborted

3.1.4.7.3 POS_TABLE_WR (0x5E)

Writes an entry to the active position table. The table index that was last set using POS_TABLE_SET_PTR is always overwritten.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x5E**							
MB1	T	-						
MB2	Save value (LSB)							
MB3	Save value							
MB4	Save value							
MB5	Save value (MSB)							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x5E**							
MB1	T	Return Code						
MB2	Status							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Return Code	0x00:	OK
	0x37:	Table does not exist, or index not set
Status	0:	Writing completed successfully
	1:	Writing aborted

3.1.4.7.4 POS_TABLE_TEACH (0x5F)

Writes the current position to the active position table.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x5F**							
MB1	T	-						
MB2	Target for measured value							
MB3	Measurement							
MB4	Reserved							
MB5	Reserved							

Target for measured value	0 ... 49:	Index at which the current position in the currently active position table is to be filed (see also TABLE_START)
	-1 (0xFF):	Save current position as negative limit Drive_Range_Neg (see configuration table)
	-2 (0xFE):	Save current position as positive limit Drive_Range_Pos (see configuration table)
	-3 (0xFD):	The current position is the zero point for a relevant measurement
	50 ... 252:	Reserved
Measureme nt	0:	Absolute measurement: Save current position
	1:	Relative measurement: Zero point for relative measurement – save current position
	2 ... 255:	Reserved

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x5F**							
MB1	T	Return Code						
MB2	Status							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Return Code	0x00:	OK
	0x37:	Table or specified index does not exist
Status	0:	Writing completed successfully
	1:	Writing aborted

3.2 Commands for Move Mode

3.2.1 Overview of Commands for Move Mode

Function	Opcode	Meaning	Page
Setpoint commands			
MOVE	0x02	Each MOVE command starts one positioning process.	164
MOVE_IMMEDIATE	0x03	Each MOVE command starts one positioning process.	165
MOVE_TABLE	0x04	Each MOVE command starts one positioning process.	166
MOVE_TABLE_ IMMEDIATE	0x05	Each MOVE command starts one positioning process.	167
MOVE_REL	0x06	Each MOVE command starts one positioning process.	168
MOVE_TABLE_REL	0x08	Each MOVE command starts one positioning process.	169
SPEED	0x10	The SPEED commands run the drive up to a defined speed.	170
SPEED_IMMEDIATE	0x11	The SPEED commands run the drive up to a defined speed.	171
STOP_FAST	0x18	The SPEED commands run the drive to a defined speed.	172
STOP_NO_RAMP	0x19	The SPEED commands run the drive to a defined speed.	173
START_REFERENCING	0x20	Starts a reference run.	174
SET_ACC_MODE	0x21	Sets the acceleration and deceleration mode.	175
SET_ACC	0x22	Sets acceleration and/or delay; valid as of the next positioning process.	177
SET_ACC_PARAM_UP	0x23	Set the Acc_ParamUp parameter for acceleration; valid as of the next positioning process.	178
SET_ACC_PARAM_ DOWN	0x24	Set the Acc_ParamDown parameter for delay (deceleration); valid as of the next positioning process.	179
SET_VELOCITY	0x25	Sets the positioning speed; valid as of the next positioning process.	180
SET_VELOCITY_ TARGET	0x2B	Velocity to target position. Valid only for the next positioning process and is then reset automatically to zero after the next positioning process.	181
SET_ACTUALPOSITON	0x2E	The current position is applied to the transferred value.	182

Function	Opcode	Meaning	Page
SET_ACTUALPOSITION_ZERO	0x2F	The current position is set to zero	183
SET_CURRENT	0x39	Sets the motor current for drive movement.	184
Math commands			
VAR_SET	0x50	Sets one of the variables FILT1 ... FILT8 to the specified value	185
VAR_INC	0x51	Adds the specified value to one of the variables FILT1 … FILT8.	186
VAR_DEC	0x52	Subtracts the specified value from one of the variables FILT1 … FILT8.	187
VAR_ADD	0x53	Adds two variables and writes the results to a third variable.	188
VAR_SUB	0x54	Subtracts two variables and writes the results to a third variable.	189
VAR_MUL	0x55	Multiplies one variable by another one and writes the results to a third variable.	190
VAR_COPY	0x56	Copes one variable to another variable.	191
VAR_DIV	0x57	Divides one variable by another one and writes the results to a third variable.	192
WAIT_TIME	0x70	Waits a defined time period before processing the next command.	193
WAIT_TEST_BIT	0x71	Before processing the next command waits until the specified bit has the status 0 or 1.	194
Auxiliary commands			
WR_BIT	0x78	Sets a bit to 0 or 1.	195
NOP	0xF0	No function	196
PROG_STOP	0xF1	Ends table processing.	197
PROG_END	0x00 or 0xFF	End of table.	198
GOTO	0xF5	Continues table process at the addressed entry.	199
GOTO_IF	0xF6	If a bit has been set, table processing is continued at the addressed entry; otherwise from the next entry.	200
GOTO_IF_NOT	0xF7	If a bit has been deleted, table processing is continued at the addressed entry; otherwise from the next entry.	201
GOTO_LABEL	0xF8	Continues table process from a defined label.	202
GOTO_LABEL_IF	0xF9	Continues table processing for a defined label if a bit has been set.	203

Function	Opcode	Meaning	Page
GOTO_LABEL_IF_NOT	0xFA	Continues table processing for a defined label if a bit has been deleted.	204
LABEL	0xFB	Defines a label for a step target	205

3.2.2 Overview of Move Mode Commands, Sorted by Opcodes

Function	Opcode	Meaning	Page
PROG_END	0x00 or 0xFF	End of table.	198
MOVE	0x02	Each MOVE command starts one positioning process.	164
MOVE_IMMEDIATE	0x03	Each MOVE command starts one positioning process.	165
MOVE_TABLE	0x04	Each MOVE command starts one positioning process.	166
MOVE_TABLE_ IMMEDIATE	0x05	Each MOVE command starts one positioning process.	167
MOVE_REL	0x06	Each MOVE command starts one positioning process.	168
MOVE_TABLE_REL	0x08	Each MOVE command starts one positioning process.	169
SPEED	0x10	The SPEED commands run the drive up to a defined speed.	170
SPEED_IMMEDIATE	0x11	The SPEED commands run the drive up to a defined speed.	171
STOP_FAST	0x18	The SPEED commands run the drive to a defined speed.	172
STOP_NO_RAMP	0x19	The SPEED commands run the drive to a defined speed.	173
START_REFERENCING	0x20	Starts a reference run.	174
SET_ACC_MODE	0x21	Sets the acceleration and deceleration mode.	175
SET_ACC	0x22	Sets acceleration and/or delay; valid as of the next positioning process.	177
SET_ACC_PARAM_UP	0x23	Set the Acc_ParamUp parameter for acceleration; valid as of the next positioning process.	178
SET_ACC_PARAM_ DOWN	0x24	Set the Acc_ParamDown parameter for delay (deceleration); valid as of the next positioning process.	179
SET_VELOCITY	0x25	Sets the positioning speed; valid as of the next positioning process.	180
SET_VELOCITY_ TARGET	0x2B	Velocity to target position. Valid only for the next positioning process and is then reset automatically to zero after the next positioning process.	181
SET_ACTUALPOSITON	0x2E	The current position is applied to the transferred value.	182
SET_ACTUALPOSITION_ ZERO	0x2F	The current position is set to zero	183

Function	Opcode	Meaning	Page
SET_CURRENT	0x39	Sets the motor current for drive movement.	184
VAR_SET	0x50	Sets one of the variables FILT1 ... FILT8 to the specified value	185
VAR_INC	0x51	Adds the specified value to one of the variables FILT1 … FILT8.	186
VAR_DEC	0x52	Subtracts the specified value from one of the variables FILT1 … FILT8.	187
VAR_ADD	0x53	Adds two variables and writes the results to a third variable.	188
VAR_SUB	0x54	Subtracts two variables and writes the results to a third variable.	189
VAR_MUL	0x55	Multiplies one variable by another one and writes the results to a third variable.	190
VAR_COPY	0x56	Copes one variable to another variable.	191
VAR_DIV	0x57	Divides one variable by another one and writes the results to a third variable.	192
WAIT_TIME	0x70	Waits a defined time period before processing the next command.	193
WAIT_TEST_BIT	0x71	Before processing the next command waits until the specified bit has the status 0 or 1.	194
WR_BIT	0x78	Sets a bit to 0 or 1.	195
NOP	0xF0	No function	196
PROG_STOP	0xF1	Ends table processing.	197
GOTO	0xF5	Continues table process at the addressed entry.	199
GOTO_IF	0xF6	If a bit has been set, table processing is continued at the addressed entry; otherwise from the next entry.	200
GOTO_IF_NOT	0xF7	If a bit has been deleted, table processing is continued at the addressed entry; otherwise from the next entry.	201
GOTO_LABEL	0xF8	Continues table process from a defined label.	202
GOTO_LABEL_IF	0xF9	Continues table processing for a defined label if a bit has been set.	203
GOTO_LABEL_IF_NOT	0xFA	Continues table processing for a defined label if a bit has been deleted.	204
LABEL	0xFB	Defines a label for a step target	205

3.2.3 Overview of Move Mode Commands, Sorted by Function

Function	Opcode	Meaning	Page
GOTO	0xF5	Continues table process at the addressed entry.	199
GOTO_IF	0xF6	If a bit has been set, table processing is continued at the addressed entry; otherwise from the next entry.	200
GOTO_IF_NOT	0xF7	If a bit has been deleted, table processing is continued at the addressed entry; otherwise from the next entry.	201
GOTO_LABEL	0xF8	Continues table processing from a defined label.	202
GOTO_LABEL_IF	0xF9	Continues table processing for a defined label if a bit has been set.	203
GOTO_LABEL_IF_NOT	0xFA	Continues table processing for a defined label if a bit has been deleted.	204
LABEL	0xFB	Defines a label for a step target	205
MOVE	0x02	Each MOVE command starts one positioning process.	164
MOVE_IMMEDIATE	0x03	Each MOVE command starts one positioning process.	165
MOVE_REL	0x06	Each MOVE command starts one positioning process.	168
MOVE_TABLE	0x04	Each MOVE command starts one positioning process.	166
MOVE_TABLE_ IMMEDIATE	0x05	Each MOVE command starts one positioning process.	167
MOVE_TABLE_REL	0x08	Each MOVE command starts one positioning process.	169
NOP	0xF0	No function	196
PROG_END	0x00 or 0xFF	End of table.	198
PROG_STOP	0xF1	Ends table processing.	197
SET_ACC	0x22	Sets acceleration and/or delay; valid as of the next positioning process.	177
SET_ACC_ PARAM_DOWN	0x24	Set the Acc_ParamDown parameter for delay (deceleration); valid as of the next positioning process.	179
SET_ACC_PARAM_UP	0x23	Set the Acc_ParamUp parameter for acceleration; valid as of the next positioning process.	178
SET_ACC_MODE	0x21	Sets the acceleration and deceleration mode.	175
SET_ACTUALPOSITION_ ZERO	0x2F	The current position is set to zero	183

Function	Opcode	Meaning	Page
SET_ACTUALPOSITON	0x2E	The current position is applied to the transferred value.	182
SET_CURRENT	0x39	Sets the motor current for drive movement.	184
SET_VELOCITY	0x25	Sets the positioning speed; valid as of the next positioning process.	180
SET_VELOCITY_ TARGET	0x2B	Velocity to target position. Valid only for the next positioning process and is then reset automatically to zero after the next positioning process.	181
SPEED	0x10	The SPEED commands run the drive to a defined speed.	170
SPEED_IMMEDIATE	0x11	The SPEED commands run the drive to a defined speed.	171
START_REFERENCING	0x20	Starts a reference run.	174
STOP_FAST	0x18	The SPEED commands run the drive to a defined speed.	172
STOP_NO_RAMP	0x19	The SPEED commands run the drive to a defined speed.	173
VAR_ADD	0x53	Adds two variables and writes the results to a third variable.	188
VAR_COPY	0x56	Copes one variable to another variable.	191
VAR_DEC	0x52	Subtracts the specified value from one of the variables FILT1 … FILT8.	187
VAR_DIV	0x57	Divides one variable by another one and writes the results to a third variable.	192
VAR_INC	0x51	Adds the specified value to one of the variables FILT1 … FILT8.	186
VAR_MUL	0x55	Multiplies one variable by another one and writes the results to a third variable.	190
VAR_SET	0x50	Sets one of the variables FILT1 ... FILT8 to the specified value	185
VAR_SUB	0x54	Subtracts two variables and writes the results to a third variable.	189
WAIT_TEST_BIT	0x71	Before processing the next command waits until the specified bit has the status 0 or 1.	194
WAIT_TIME	0x70	Waits a defined time period before processing the next command.	193
WR_BIT	0x78	Sets a bit to 0 or 1.	195

3.2.4 Reference Commands for Move Mode

3.2.4.1 Setpoint commands

3.2.4.1.1 MOVE (0x02)

Starts a positioning process. This command immediately deletes the "On_Target" bit and sets that bit when the target position is reached. This command is accepted only when the last positioning process has been completed and the "On_Target" bit has been set. The acceleration and velocity specified by SET_ACC (0x22) and SET_VELOCITY (0x25) are used for positioning commands.

Request								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x40**							
MB1	T	-						
MB2	**0x02**							
MB3	Position (LSB)							
MB4	Position							
MB5	Position (MSB)							

Response								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x02**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.1.2 MOVE_IMMEDIATE (0x03)

Starts a positioning process. This command immediately deletes the "On_Target" bit and sets that bit when the target position is reached. This command will interrupt any positioning process that may already be in progress and immediately starts a new positioning process. The acceleration and velocity specified by SET_ACC (0x22) and SET_VELOCITY (0x25) are used for positioning commands.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x03**							
MB3	Position (LSB)							
MB4	Position							
MB5	Position (MSB)							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x03**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.1.3 MOVE_TABLE (0x04)

Starts a positioning process. This command immediately deletes the "On_Target" bit and sets that bit when the target position is reached. This command is accepted only when the last positioning process has been completed and the "On_Target" bit has been set. This command reads the target position from the specified location in a separate position table. The acceleration and velocity specified by SET_ACC (0x22) and SET_VELOCITY (0x25) are used for positioning commands.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x04**							
MB3	No. of table entry with target position							
MB4	Reserved							SRC
MB5	Reserved							

SRC 0: Read out from position table.

1: Read out from variables FILT1 ... FILT8.

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x04**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.1.4 MOVE_TABLE_IMMEDIATE (0x05)

Starts a positioning process. This command immediately deletes the "On_Target" bit and sets that bit when the target position is reached. This command will interrupt any positioning process that may already be in progress and immediately starts a new positioning process. This command reads the target position from the specified location in a separate position table. The acceleration and velocity specified by SET_ACC (0x22) and SET_VELOCITY (0x25) are used for positioning commands.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x05**							
MB3	No. of table entry with target position							
MB4	Reserved							SRC
MB5	Reserved							

SRC 0: Read out from position table.

1: Read out from variables FILT1 ... FILT8.

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x05**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.1.5 MOVE_REL (0x06)

Starts a positioning process. This command immediately deletes the "On_Target" bit and sets that bit when the target position is reached. This command is accepted only when the last positioning process has been completed and the "On_Target" bit has been set. This command calculates the target position relative to the last accepted target (if available), or to the current position. The acceleration and velocity specified by SET_ACC (0x22) and SET_VELOCITY (0x25) are used for positioning commands.

Request								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x40**							
MB1	T	-						
MB2	**0x06**							
MB3	Position (LSB)							
MB4	Position							
MB5	Position (MSB)							

Antwort								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x06**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.1.6 MOVE_TABLE_REL (0x08)

Starts a positioning process. This command immediately deletes the "On_Target" bit and sets that bit when the target position is reached. This command is accepted only when the last positioning process has been completed and the "On_Target" bit has been set. This command reads the target position from the specified location in a separate position table. This command calculates the target position relative to the last accepted target (if available), or to the current position. The acceleration and velocity specified by SET_ACC (0x22) and SET_VELOCITY (0x25) are used for positioning commands.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x08**							
MB3	No. of table entry with target position							
MB4	Reserved							SRC
MB5	Reserved							

SRC 0: Read out from position table.

1: Read out from variables FILT1 ... FILT8.

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x08**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.1.7 SPEED (0x10)

Runs the drive up to a defined speed. On execution of this command, the "On_Target" bit is deleted immediately and then set when the target speed is reached. This command does NOT set the positioning speed! SET_VELOCITY (0x25) must be used for that. The SPEED command is accepted only when the last process has been completed and the "On_Target" bit has been set.

Speed range: -25000 … 25000.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x10**							
MB3	Speed (LSB)							
MB4	Speed (MSB)							
MB5	Reserved							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x10**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.1.8 SPEED_IMMEDIATE (0x11)

Runs the drive up to a defined speed. On execution of this command, the "On_Target" bit is deleted immediately and then set when the target speed is reached. This command does NOT set the positioning speed! SET_VELOCITY (0x25) must be used for that. This command will interrupt any process that may already be in operation and immediately starts speed control.

Speed range: -25000 … 25000.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x11**							
MB3	Speed (LSB)							
MB4	Speed (MSB)							
MB5	Reserved							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x11**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.1.9 STOP_FAST (0x18)

Brakes the drive directly with the acceleration command SET_ACC_STOP down to standstill. Internal processing is structured such that this command is given priority. All other commands are discontinued immediately, in particular in the Mailbox mode. This command is also initiated internally when a stop condition is present, such as limit switch or stop input.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x18**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x18**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.1.10 STOP_NO_RAMP (0x19)

Sets the output frequency immediately to zero. Internal processing is structured such that this command is given priority. All other commands are discontinued immediately, in particular in the Mailbox mode. This command has priority over STOP_FAST. This command is also initiated internally when Enable has not been set.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x19**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x19**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.1.11 START_REFERENCING (0x20)

Starts a reference run. This command immediately deletes the bits On_Target and Reference_OK and sets them when the reference value is reached. The Busy bit is set during a reference run.

The reference run is always performed at the setup speed SpeedSetup and at the setup acceleration SetupAcceleration.

Request								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x40**							
MB1	T	-						
MB2	**0x20**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved					DIR	STD	SWT

SWT 0: Reference run to reference switch:

1: Reference run to limit switch:

STD If SWT = 0, then STD indicates the starting direction, if SWT = 1, then STD specifies the limit switch.

0: Starting direction negative / negative limit switch:

1: Starting direction positive / positive limit switch:

DIR DIR is evaluated only when SWT = 1.

0: Reference run started from negative end.

1: Reference run started from positive end.

Response								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x20**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.1.12 SET_ACC_MODE (0x21)

Set the type of acceleration and delay; valid as of the next positioning process.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x21**							
MB3	DEC_M		DEC_T		ACC_M		ACC_T	
MB4	Reserved							
MB5	Reserved							

ACC_T 0: constant acceleration

1: linear increase in acceleration
With ACC_M = 0, the period for acceleration increase is Acc_ParamUp, the increase in acceleration is calculated at ACC_M <> 0.

2: sin^2 increase in acceleration
With ACC_M = 0, the period for acceleration increase is Acc_ParamUp, the increase in acceleration is calculated at ACC_M <> 0.

3: Reserved

ACC_M 0: no modification

1: Acc_ParamUp interpreted as the acceleration period

2: Acc_ParamUp interpreted as the acceleration path

3: Reserved

DEC_T 0: constant deceleration

1: linear deceleration
With DEC_M = 0, the period for deceleration is Acc_ParamDown, the increase in deceleration is calculated at DEC_M <> 0.

2: sin^2 deceleration
With DEC_M = 0, the period for deceleration is Acc_ParamDown, the increase in deceleration is calculated at DEC_M <> 0.

3: Reserved

DEC_M 0: no modification

1: Acc_ParamDown interpreted as the deceleration period

2: Acc_ParamDown interpreted as the deceleration path

3: Reserved

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x21**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.1.13 SET_ACC (0x22)

Sets the type and rate of acceleration; valid as of the next positioning process (see also positioning commands MOVE…, 0x02, 0x03, 0x04, 0x05, 0x06, 0x08)

Acceleration range: 1 … 32767.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x22**							
MB3	Acceleration (LSB)							
MB4	Acceleration (MSB)							
MB5	Reserved						SEL	

SEL 0: Sets the value for acceleration and brake phase.
1: Sets the value for acceleration phase only.
2: Sets the value for brake phase only.
3: Sets the value for acceleration and brake phase.

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x22**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.1.14 SET_ACC_PARAM_UP (0x23)

Set the Acc_ParamUp parameter for acceleration; valid as of the next positioning process.

The function for the acceleration parameter as a function of the set acceleration modification is shown in the following table.

Acceleration modification (SET_ACC_MODE → ACC_M)	**Acceleration parameter Acc_ParamUp**
none	Time constant for acceleration increase with linear or sin^2*t acceleration
constant acceleration period	Acceleration time
constant acceleration path	Acceleration path

Acceleration parameter range: 1 … 16777215.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x23**							
MB3	Acceleration parameter (LSB)							
MB4	Acceleration parameter							
MB5	Acceleration parameter (MSB)							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x23**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.1.15 SET_ACC_PARAM_DOWN (0x24)

Set the Acc_ParamDown parameter for delay (deceleration); valid as of the next positioning process.

The function for the delay parameter as a function of the set delay modification is shown in the following table.

Deceleration modification (SET_ACC_MODE → DEC_M)	**Delay parameter Acc_ParamDown**
none	Time constant for deceleration increase with linear or $\sin^2 * t$ deceleration
constant delay period	Delay time
constant delay path	Delay path

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x24**							
MB3	Deceleration parameter (LSB)							
MB4	Deceleration parameter							
MB5	Deceleration parameter (MSB)							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x24**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.1.16 SET_VELOCITY (0x25)

Sets the positioning velocity; valid as of the next positioning command (see also positioning commands MOVE…, 0x02, 0x03, 0x04, 0x05, 0x06, 0x08)

Velocity range: 1 … 25000.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x25**							
MB3	Velocity (LSB)							
MB4	Velocity (MSB)							
MB5	Reserved							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x25**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.1.17 SET_VELOCITY_TARGET (0x2B)

Sets the target velocity for the next positioning process. The target velocity is automatically reset to zero after the next positioning process.

Velocity range: 1 … 25000.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x2B**							
MB3	Velocity (LSB)							
MB4	Velocity (MSB)							
MB5	Reserved							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x2B**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.1.18 SET_ACTUALPOSITON (0x2E)

The current position is set to the transferred value. The logical zero point is modified accordingly for this.

The position is given as a 24-bit value, including sign.

Request								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x40**							
MB1	T	-						
MB2	**0x2E**							
MB3	Position (LSB)							
MB4	Position							
MB5	Position (MSB)							

Response								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x2E**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.1.19 SET_ACTUALPOSITION_ZERO (0x2F)

Sets the position of the logical zero point to the current position.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x2F**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x2F**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.1.20 SET_CURRENT (0x39)

Sets the motor current for drive movement.

The corresponding bit in the valid range must be set to 1 for the working range for which the motor current is to be set. Several bits can be set simultaneously. If the corresponding bit is set to 0, the value for the motor current valid up to then is retained for this range.

Motor current range: 0 … 150 % module rated current

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x39**							
MB3	Motor current							
MB4	Reserved				Valid range			
MB5	Reserved							

Valid range	Bit 0:	Set motor current for standstill
	Bit 1:	Set motor current for acceleration
	Bit 2:	Set motor current for drive movement
	Bit 3:	Set motor current for deceleration

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x39**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.2 Math commands

3.2.4.2.1 VAR_SET (0x50)

Sets a variable to the defined value.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x50**							
MB3	1 … 8 (corresponds to FILT1 … FILT8)							
MB4	16 bit value (LSB)							
MB5	16 bit value (MSB)							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x50**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.2.2 VAR_INC (0x51)

Adds the given value to a variable.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x51**							
MB3	1 … 8 (corresponds to FILT1 … FILT8)							
MB4	16 bit value (LSB)							
MB5	16 bit value (MSB)							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x51**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.2.3 VAR_DEC (0x52)

Subtracts the given value from a variable.

Request								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x40**							
MB1	T	-						
MB2	**0x52**							
MB3	1 … 8 (corresponds to FILT1 … FILT8)							
MB4	16 bit value (LSB)							
MB5	16 bit value (MSB)							

Response								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x52**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.2.4 VAR_ADD (0x53)

Adds two variables and writes the results to a third variable.

Request								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x40**							
MB1	T	-						
MB2	**0x53**							
MB3	Result (1 … 8 corresponds to FILT1 … FILT8)							
MB4	Summand 2 (1 … 8 corresponds to FILT1 … FILT8)							
MB5	Summand 1 (1 … 8 corresponds to FILT1 … FILT8)							

Response								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x53**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.2.5 VAR_SUB (0x54)

Subtracts one variable from another one and writes the results to a third variable.

Request								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x40**							
MB1	T	-						
MB2	**0x54**							
MB3	Difference (1 … 8 corresponds to FILT1 … FILT8)							
MB4	Minuend (1 … 8 corresponds to FILT1 … FILT8)							
MB5	Subtrahend (1 … 8 corresponds to FILT1 … FILT8)							

Response								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x54**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.2.6 VAR_MUL (0x55)

Multiplies one variable by another one and writes the results to a third variable.

Request								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x40**							
MB1	T	-						
MB2	**0x55**							
MB3	Product (1 … 8 corresponds to FILT1 … FILT8)							
MB4	Multiplicand 2 (1 … 8 corresponds to FILT1 … FILT8)							
MB5	Multiplicand 1 (1 … 8 corresponds to FILT1 … FILT8)							

Response								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x55**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.2.7 VAR_COPY (0x56)

Copes one variable to another variable.

Request								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x40**							
MB1	T	-						
MB2	**0x56**							
MB3	Target (1 … 8 corresponds to FILT1 … FILT8)							
MB4	Source (1 … 8 corresponds to FILT1 … FILT8)							
MB5	Reserved							

Response								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x56**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.2.8 VAR_DIV (0x57)

Divides one variable by another one and writes the results to a third variable.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x57**							
MB3	Quotient (1 … 8 corresponds to FILT1 … FILT8)							
MB4	Dividend (1 … 8 corresponds to FILT1 … FILT8)							
MB5	Divisor (1 … 8 corresponds to FILT1 … FILT8)							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x57**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.3 Wait Commands

3.2.4.3.1 WAIT_TIME (0x70)

Waits a defined time period before processing the next command.

Waiting time range: 0 … 16777215 ms.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x70**							
MB3	Waiting time (LSB)							
MB4	Waiting time							
MB5	Waiting time (MSB)							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x70**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.3.2 WAIT_TEST_BIT (0x71)

Before processing the next command waits until the specified bit has the specified status 0 or 1.
Refer to Chapter **Fehler! Verweisquelle konnte nicht gefunden werden.**, "**Fehler! Verweisquelle konnte nicht gefunden werden.**" for the bit number.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x71**							
MB3	Bit No.							
MB4	Specified status of bit (0 or 1)							
MB5	Reserved							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x71**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.4 Auxiliary Commands

3.2.4.4.1 WR_BIT (0x78)

Sets a bit B to 0 or 1.
Refer to Chapter **Fehler! Verweisquelle konnte nicht gefunden werden.**, “**Fehler! Verweisquelle konnte nicht gefunden werden.**” for the bit number.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x78**							
MB3	Bit No.							
MB4	Specified status of bit (0 or 1)							
MB5	Reserved							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x78**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.4.2 NOP (0xF0)

Function not defined.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0xF0**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0xF0**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.4.3 PROG_STOP (0xF1)

Ends table processing. Sets velocity to zero, deactivates the output stage and ends table processing.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0xF1**							
MB3	Error message							
MB4	Reserved							
MB5	Reserved							

Error message	0:	No error message
	1 ... 8:	Error message ERROR_TBL_PROGRAM_STOP1 ... 8
	9 ... 255	Reserved

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0xF1**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.4.4 PROG_END (0x00 oder 0xFF)

End of table (default command for a blank / deleted table). Sets velocity to zero, deactivates the output stage, ends table processing and reports the error ERR_PROG_END.

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0x00 or 0xFF**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0x00 or 0xFF**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.4.5 GOTO (0xF5)

Continues table processing at the addressed entry.

Command number range: 1 … 500

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0xF5**							
MB3	Number of next command (LSB)							
MB4	Number of next command (MSB)							
MB5	Reserved							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0xF5**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.4.6 GOTO_IF (0xF6)

If a bit has been set, table processing is continued at the addressed entry; otherwise the next table entry is used.

Command number range: 1 … 500

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0xF6**							
MB3	Number of next command (LSB)							
MB4	Number of next command (MSB)							
MB5	Number of bit to be checked							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0xF6**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.4.7 GOTO_IF_NOT (0xF7)

If a bit has not been set, table processing is continued at the addressed entry; otherwise the next table entry is used.

Command number range: 1 … 500

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0xF7**							
MB3	Number of next command (LSB)							
MB4	Number of next command (MSB)							
MB5	Number of bit to be checked							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0xF7**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.4.8 GOTO_LABEL (0xF8)

Continues table processing at the addressed entry.

Label number range: 1 … 65536

Request								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x40**							
MB1	T	-						
MB2	**0xF8**							
MB3	Label number (LSB)							
MB4	Label number (MSB)							
MB5	Reserved							

Response								
Byte	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0xF8**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.4.9 GOTO_LABEL_IF (0xF9)

If a bit has been set, table processing is continued at the addressed entry; otherwise the next table entry is used.

Label number range: 1 … 65536

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0xF9**							
MB3	Label number (LSB)							
MB4	Label number (MSB)							
MB5	Number of bit to be checked							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0xF9**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.4.10 GOTO_LABEL_IF_NOT (0xFA)

If a bit has not been set, table processing is continued at the addressed entry; otherwise the next table entry is used.

Label number range: 1 … 65536

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0xFA**							
MB3	Label number (LSB)							
MB4	Label number (MSB)							
MB5	Number of bit to be checked							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0xFA**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.2.4.4.11 LABEL (0xFB)

Defines a label as a step target for a GOTO command; no further function. If more than one identical label numbers are defined, the one at the lowest address in the table shall be valid.

Label number range: 1 … 65536

Request								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	-						
MB2	**0xFB**							
MB3	Label number (LSB)							
MB4	Label number (MSB)							
MB5	Reserved							

Response								
Byte	**2^7**	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
MB0	**0x40**							
MB1	T	Return Code						
MB2	**0xFB**							
MB3	Reserved							
MB4	Reserved							
MB5	Reserved							

3.3 Error Blink Codes

The fault message consists of a 4-place digit.
The error display starts with the first blinking sequence (approx. 10 Hz).
After a short break, the second blinking sequence starts (approx. 2 Hz). This sequence represents the highest digit for the 4-place error code.
All of the other digits then appear at 1 second intervals, down to the lowest value digit.
The blinking sequence is then repeated.

Errors with numbers of type "1nnn" can be acknowledged, with no further reactions to follow.
Errors with numbers of type "2nnn" can be acknowledged, after which a warm start is carried out.
Errors with numbers of type "3nnn" can not be acknowledged; the fieldbus node goes into the "Stop" status. The node must then be restarted.

Errors can also be reported in status byte S0, bit 6 (ERR) and thus initiate an acyclic diagnostics message.
This bit is addressed continuously as long as the error is present.
Enabling for the indication is set using the configuration parameter Error_Notification Bit 0.
This bit has the following meaning:

0: Errors are not reported via status byte S0, bit 6 (ERR).
1: Errors are reported via status byte S0, bit 6 (ERR).

An explanation of the individual error numbers is given in the following table.

3.3.1 Overview of Error Blink Codes

Error No.	Name	Description	Possible cause/Remedy
1111	CI_UNKNOWN COMMAND	INT	
1112	CI_UNKNOWN COMMAND2	Unknown command	A Drive command was received with an unknown command. Check command (and possibly the move program)
1113	CI_ERR1	Unknown command, like Error 1112	Same as for Error 1112
1114	CI_NOT IMPLEMENTED	Command not implemented	Command has been prepared but has not (yet) been implemented, otherwise same as Error 1112
1115	CI_COMMAND_ DENIED	Could not execute command	It was not possible to execute a command, as the marginal conditions do not permit it. (incorrect mode, incorrect parameter, ... for this command)
1116	CI_SPEED1	Configuration: Invalid maximum frequency	Unacceptable value specified for maximum frequency
1117	CI_SPEED2	Invalid frequency value	Unacceptable value specified for set frequency
1118	CI_SPEED3	internal	
1121	CI_ACC1	Invalid acceleration value	Check the specified acceleration value (configuration, Move program, process data ….)
1122	CI_ACC2	Invalid acceleration value	Same as for Error 1121
1123	CI_ ROUNDARYSHAFT _ SETPOINT	Operation with rotary shaft: absolute position setpoint output of rotary shaft range	During operation with the rotary shaft, the absolute position must lie within the range 0 … RotaryShaftRange. Check the position data and parameters for the rotary shaft.
1124	CI_POS_TABLE	Invalid table selected for positioning to position table	Check Parameter 2 for a command "MOVE_L"

Error No.	Name	Description	Possible cause/Remedy
1125	CI_SET_POS	Actual value can not be set (e.g. while positioning task in progress)	The drive must be motionless for one SET_POS command.
1141	CM_UNKNOWN_ CMDSOURCE	internal	
1142	CM_AUTOSTART_ NOT_POSSIBLE	Autostart of a Move program not possible (not available)	Load the Move program to the module, or de-activate the autostart.
1161	CONFIG_WRITE_ SIZE	Invalid size specified for writing of the configuration	An attempt was made to enter an element into the configuration with a length less than 1 byte or greater than 4 bytes.
1162	CONFIG_ WRONGPASSWORD	Password not accepted	Specify a correct password
1211	CTRLOUT_FREF	Configuration: Invalid maximum frequency	The maximum frequency must be between 1…..25000
1212	CTRLOUT_F_FREF	Maximum frequency reached	An attempt was made to specify a frequency that is greater than the maximum frequency. Check the specified target frequency and the maximum frequency values.
1213	CTRLOUT_ CURRENT_SCALE	Configuration: Invalid rated current	Incorrect rated motor current specified. Check the configuration.
1214	CTRLOUT_ OPERATION_MODE	Configuration: The selected application is not present in this module.	Check the configuration value for Mode 1.
1215	CTRLOUTP_CURR_ PARAM	Configuration: Password for current regulation does not agree with the current regulation parameters.	Configuration: Check the parameters for current regulation with the password for current regulation.
1216	CTRLOUTP_F_DIV	Configuration: Frequency prescaler faulted	Configuration: Check parameters for frequency prescaler
1217	CTRLOUT_ CURRENT_FACTOR	Current setting incorrect (greater than 150%)	Invalid parameterization for current setting, check configuration, check Move program of Mailbox commands where applicable

Error No.	Name	Description	Possible cause/Remedy
1241	IO_BITINDEX TOLARGE	Access to non-available bit (internal)	Configuration, check Move program and Mailbox commands for incorrect bit addresses
1242	IO_ILLEGAL_ USERBITNR	Bit can not be modified by user	An attempt was made to change a bit not linked to MZERO or MONE using the Move program of Mailbox command
1243	IO_USERBIT_ READONLY	Bit can not be written	Bit not authorized for external write access
1245	IO_RECURSIVE_ LINK	Excessive nesting for linked bit	A linkable bit was linked to a linkable bit that was linked to a linkable bit; a linkable bit may be linked to itself.
1246	IO_TIMER_1	Configuration: Filter function not defined	Check configuration for filter functions
1247	IO_UNKNOWN_ TIMER	Filter with this number not available	Check configuration for filter functions
1248	IO_TIMERMODE	Filter function does not permit write access	The current filter function configuration does not permit writing of the filter
1249	IO_BITNOT IMPLEMENTED	Requested bit not implemented (internal)	A bit being used can not be queried (internal error)
1311	TBL_PROGRAM_ STOP1	Move program terminated with error message 1	Check the termination condition of the Move program
1312	TBL_PROGRAM_ STOP2	Move program terminated with error message 2	Check the termination condition of the Move program
1313	TBL_PROGRAM_ STOP3	Move program terminated with error message 3	Check the termination condition of the Move program
1314	TBL_PROGRAM_ STOP4	Move program terminated with error message 4	Check the termination condition of the Move program
1315	TBL_PROGRAM_ STOP5	Move program terminated with error message 5	Check the termination condition of the Move program
1316	TBL_PROGRAM_ STOP6	Move program terminated with error message 6	Check the termination condition of the Move program

Error No.	Name	Description	Possible cause/Remedy
1317	TBL_PROGRAM_ STOP7	Move program terminated with error message 7	Check the termination condition of the Move program
1319	TBL_PROGRAM_ END	Move program not ended properly	Move program ended without the regular Stop command
1321	TBL_UNKNOWN_ CMD	Unknown command for table processing (internal)	(internal)
1322	TBL_ LABELNOTFOUND	Label not available as step target in Move program	Check the definition of the label in the Move program
1322	TBL_ LABELNOTFOUND	Label not available as step target in Move program	Check the definition of the label in the Move program
1323	TBL_ENDOFTABLE	Step target outside of the Move program	Check the step targets in the Move program
1331	TBL_CAM9PARMS	Configuration: Invalid parameter for camshaft channel 9	Check configuration for camshaft channel 9, the cycle may not be 0
1332	TBL_INDEX_OUT_ OF_RANGE	Table access outside of table	Check the tables and table access
1333	TBL_INVALID	Access to invalid table	Check access to tables
1334	TBL_COPY_FAILED	Version can not be written to EEPROM	EEPROM defective
1351	OPC_START	START command not accepted	Start may only be set when a mode is active
1352	OPC_TBL_START	Move program can not be started (not available)	Move program can not be started (not available)
1353	INV_CONTR_IN_ PULSE_MODE	Mode not available in selected application	Check the activation of the modes and configuration for Application_Selector
1353	INV_CONTR_IN_ PULSE_MODE	Mode not available in selected application	Check the activation of the modes and configuration for Application_Selector
1354	OPC_ MULTIMODE_1	Multiple modes selected	Selection of mode is ambiguous
1355	OPC_ MULTIMODE_2	Multiple modes selected	Selection of mode is ambiguous
1356	OPC_WHOOPS1	Unknown mode selected (internal)	Unknown mode selected (internal)

Error No.	Name	Description	Possible cause/Remedy
1411	PARTMODL_ CURRENT	internal	internal
1412	PARTMODL_ CURRENT_SET	Could not execute current setting	Check parameter for Move command SET_CURRENT
1413	PARTMODL_ CURRENT_TIME	Time limit reached for overcurrent	The drive has been operated too long at a current >150%. Check move profile and current setting.
1414	PARTMODL_ FIFONOTREADY	internal	internal
1415	PARTMODL_ POSITION_RANGE	The movement calculator has determined partial movement that exceeds the internal 32-bit position range.	Check movement parameters. This error occurs on unrealistic settings for velocity, acceleration or positions. With extreme parameters, braking from a high speed, for example, at the lowest deceleration yields a brake path that far exceeds the internal value range.
1416	PARTMODL_ SPEED_RANGE	The movement calculator has determined partial movement that exceeds the permissible velocity range.	Check the specified velocities
1417	PARTMODL_ INTERN1	Unknown status of internal FIFO: internal	internal
1431	PROT_REF_DIR	Reference run without direction setting	The reference run via the process image must be informed of the starting direction through C3_Setup_Dir_Neg or C3_Setup_Dir_Pos.
1432	PULSE_TRAIN_NO_ CAM_ACTIVE	Pulse chain generator can not be started if the camshaft is not defined	Download the definition for the pulse chain from the camshaft table
1433	PROT_UNKNOW_ MODE	No application selected	Check configuration for Application_Selector
1434	PROT_TEST_MODE	Special function, integration test, active	The modules are switched to the test mode via register 32
1451	REF_SWITCH_NOT_ FOUND	Reference contact not found	Check reference switch

Error No.	Name	Description	Possible cause/Remedy
1452	REF_LIM_SWITCH	Reference switch not clearly identified on start of reference run	Both limit switches active simultaneously during search for reference switch
1453	REF_SPEED	Speed setting missing for reference run	Referencing speed of 0 is unacceptable
1454	REF_START_DIR_ LIMIT	Reference run to limit switch: Limit switch already actuated	Reference run to limit switch: Limit switch already actuated
1455	ERROR_REF_LIM_ SWITCH_NOT_ EXPECTED	Reference run: Unexpected limit switch	Check limit switch wiring; was the reference run started beyond the limit switch?
1511	UNITS_POS_INT_ RESULT	Conversion of position from user-specific unit to internal unit: Range exceeded	Configuration: Check units conversion
1512	UNITS_POS_USER_ RESULT	Conversion of position from internal unit to user-specific unit: Range exceeded	Configuration: Check units conversion
1513	UNITS_SPEED_INT_ RESULT	Conversion of speed from user-specific unit to internal unit: Range exceeded	Configuration: Check units conversion
1514	UNITS_SPEED_ USER_RESULT	Conversion of speed from internal unit to user-specific unit: Range exceeded	Configuration: Check units conversion
1515	UNITS_ACC_INT_ RESULT	Conversion of acceleration from user-specific unit to internal unit: Range exceeded	Configuration: Check units conversion
1516	UNITS_ACC_USER_ RESULT	Conversion of acceleration from internal unit to user-specific unit: Range exceeded	Configuration: Check units conversion
1517	UNITS_PARAM_ ZERO	Parameter for conversion is zero	Configuration: Units conversion: Divisor is zero
1521	SYS_MODE	Configuration: Application can not be executed at this module. (Module 2, 3, 4 only for stepper position control or frequency control)	Check configuration for Application_Selector

Error No.	Name	Description	Possible cause/Remedy
1551	MCALC_SPEED1	internal	internal
1552	MCALC_SPEED2	internal	internal
1553	MCALC_SPEED3	internal	internal
1554	MCALC_ACC1	During the ramp run at a defined ramp time the movement calculator has determined partial movement that exceeds the internal value range for acceleration	Adapt ramp time setting as appropriate
1555	MCALC_ACC2	same as MCALC_ACC1	same as MCALC_ACC1
1556	MCALC_ACC3	During the ramp run at a defined ramp path the movement calculator has determined partial movement that exceeds the internal value range for acceleration	Adapt ramp path setting as appropriate
1557	MCALC_PARA	The movement calculator has received invalid parameters	Check all parameters; Acceleration, moving velocity, ramp time, ramp path, ramp type, starting position, target position
1561	MCALC_TIME1	Internal range exceeded for moving time	internal *
1562	MCALC_TIME2	The movement calculator has determined partial movement that exceeds the internal time range (>500 h).	Check movement parameters. This error occurs on unrealistic settings for velocity, acceleration or positions. With extreme parameters, moving at the lowest acceleration, for example, yields a movement time that far exceeds the internal value range of 2^31ms.
1563	MCALC_TIME3	Internal range exceeded for moving time	internal *
1564	MCALC_TIME4	The movement calculator has determined a movement sequence that exceeds the internal time range (>500 h).	Check movement parameters. This error occurs on unrealistic settings for velocity, acceleration or positions. With extreme parameters, moving at the lowest acceleration, for example, yields a movement time that far exceeds the internal value range of 2^31ms.

Error No.	Name	Description	Possible cause/Remedy
1565	MCALC_DIST1	Internal range exceeded for movement path	internal *
1566	MCALC_DIST2	Internal range exceeded for movement path	internal *
1567	MCALC_DIST3	same as PARTMODL_ POSITION_RANGE	same as PARTMODL_ POSITION_RANGE
1568	MCALC_DIST4	Internal range exceeded for movement path	internal *
1569	MCALC_DIST5	same as PARTMODL_ POSITION_RANGE	same as PARTMODL_ POSITION_RANGE
1571	MCALC_MOVE1	internal: Movement calculator can find no solution	internal
1572	MCALC_MOVE2	internal: Movement calculator can find no solution	internal
1573	MCALC_MOVE3	internal: Movement calculator can find no solution	internal
1611	ERR_ILLEGAL_ ERRORCODE	Invalid error code to be reported	internal
1911	COMMAND_IS_ RUNNING	Command can not be executed, as another command is currently being processed	internal
1912	HIGH_PRIO_ COMMAND_IS_ RUNNING	Command can not be executed, as another command of higher priority is currently being processed	internal
1931	PARTMODL_ LIMITSWITCH	Command can not be executed, as a limit switch is active	Move drive away from limit switch
2811	KBUS	internal	internal *
2821	CFG_FACTORY_ LOAD	A Factory_Default data set has been copied for upload to RAM	A CONFIG_RESTORE command has been executed without a warm start. The module is not operational is this state. → Conduct a warm start or Power-on reset
2831	MEASURE_ERR1	internal: Unknown hardware	internal

Error No.	Name	Description	Possible cause/Remedy
2832	MEASURE_ERR2	same as MEASURE_ERR1	internal
2833	MCALC_INTERN1	internal: Error in path calculation	internal
2834	MCALC_INTERN2	internal: Error in path calculation	internal
2835	MCALC_INTERN3	intern: unknown acceleration profile	internal
2836	MCALC_INTERN4	MCALC_INTERN3	internal
2837	MCALC_INTERN5	MCALC_INTERN3	internal
2838	MCALC_INTERN6	MCALC_INTERN3	internal
2839	MCALC_INTERN7	MCALC_INTERN3	internal
2841	MCALC_INTERN8	MCALC_INTERN3	internal
2842	MCALC_INTERN9	MCALC_INTERN3	internal
2843	MCALC_INTERN10	MCALC_INTERN3	internal
2844	MCALC_INTERN11	MCALC_INTERN3	internal
2845	MCALC_BUFFER_ FULL	internal: buffer overflow	internal
2846	MOVECALC_ACC2	internal	internal
2863	TEST_EERPOM_ FAILURE	Self test: EEPROM faulted	Hardware defective
2864	TEST_CPLD_ FAILURE	Self test: CPLD faulted	Hardware defective
2865	TEST_INVALID_ MODULE	Self test: Unknown hardware	Hardware defective
2866	GENERIC_TEST	Self test: Wrong hardware	Hardware defective
2871	RS232_TX_ TIMEOUT	internal: Timeout at debug interface	internal
2881	SYS_IDLE_ RECURSIVE	internal:	internal
2882	SYS_SPI_ TIMEOUT	internal:	internal

Error No.	Name	Description	Possible cause/Remedy
2891	VERSION_ UNKNOWN_IDENT	Unknown hardware	Hardware defective
2892	VERSION_NOT_ COMPATIBLE_HW	Hardware not compatible with software	Hardware defective
2893	ERROR_VERSION_ NOT_COMPATIBLE _CPLD	This software checked the CPLD version and reported to inconsistency the error	Module send to manufature
3111	INT_KBUS	internal	internal
3112	INT_WATCHDOG	internal: Watchdog	internal
3113	INT_SPURIOUS	internal	internal
3114	INT_UNUSED	internal	internal
3115	INT_FIQ	internal	internal
3116	INT_SWI	internal	internal
3117	INT_UNDEF_INST	internal	internal
3118	INT_FETCH	internal	internal
3119	INT_DATA_ACESS	internal	internal
3121	INT_ROM_ISR	internal	internal
3122	INT_STACK_ OVERFLOW	internal	internal
3142	SYS_PLL_NOT_ LOCKED	internal	internal
3143	SYS_ADC_ TIMEOUT	internal	internal
3144	SYSTEMEXIT	internal	internal
3155	ERR_ILLEGAL_ ERRORCODE	Invalid error code to be reported	internal
3166	OPC_ MULTIMODE_1	internal	internal *
3167	OPC_ MULTIMODE_2	internal	internal *
3168	OPC_WHOOPS1	internal	internal *
3179	TBL_COPY_FAILED	Error while writing to EEPROM	internal *
3211	PARTMODL_ FIFONOTREADY	internal	internal *

Error No.	Name	Description	Possible cause/Remedy
3212	PARTMODL_ POSITION_RANGE	internal	internal *
3213	PARTMODL_ SPEED_RANGE	internal	internal *
3214	PARTMODL_ INTERN1	internal	internal *
3215	PARTMODL_ERR4	internal	internal *
3216	PARTMODL_ERR5	internal	internal *
3231	TBL_INVALID	internal	internal *
3232	TBL_CP2EEPROM_ FAIL	internal	internal *
3233	TBL_COPY_ INVALID	internal	internal *
3234	TBL_UNKNOWN_ TTYPE	internal	internal *
3271	TEST_FLASH	Program memory checksum corrupted	internal
3272	TEST_FLASH_ CRCGEN	Program memory checksum not available	internal
3273	TEST_EERPOM_ FAILURE	Self test: EEPROM faulted	internal
3274	TEST_CPLD_ FAILURE	Self test: CPLD faulted	internal
3275	TEST_INVALID_ MODULE	Self test: Unknown hardware	internal
3276	GENERIC_TEST	Self test: Wrong hardware	internal

3.4 Bit field for I/O driver

The bit functions described in this table refer to the stepper positioning controller standard application.

If the bits have a different function with other applications, this is noted in the description for the specific application.

The following conventions apply:

- Source bits are assigned numbers 0 to 127 and may not be used as target bits. A source bit may reference several target bits.
- Target bits are assigned numbers 128 to 255 and may also be used as source bits. Target bits have exactly one source.
- References are stored in the configuration table. The names of the table entries correspond to those in the bit table. The prefix Ptr is placed in front of the identifier.
- The standard link between the source and target is entered in the column "Target/Source". This corresponds to the WAGO default settings (FACTORY_DEFAULT_1).

<table>
<tr><th rowspan="2">Name</th><th colspan="2">Bit number</th><th rowspan="2">Type</th><th colspan="2">Default</th><th rowspan="2" colspan="2">Description</th></tr>
<tr><th>Dec.</th><th>Hex.</th><th>Target/Source</th><th>Bit no.</th></tr>
<tr><td>ZERO</td><td>0</td><td>0x00</td><td>SRC</td><td>0</td><td>-</td><td colspan="2">Bit always (false 0)</td></tr>
<tr><td>ONE</td><td>1</td><td>0x01</td><td>SRC</td><td>1</td><td>-</td><td colspan="2">Bit always (true 1)</td></tr>
<tr><td>MZERO</td><td>2</td><td>0x02</td><td>SRC</td><td>0</td><td>-</td><td colspan="2">A bit that is linked to MZERO is first false after a reset, but can be manipulated as required using the mailbox command or the Move program.</td></tr>
<tr><td>MONE</td><td>3</td><td>0x03</td><td>SRC</td><td>1</td><td>-</td><td colspan="2">A bit that is linked to MZERO is first true after a reset, but can be manipulated as required using the mailbox command or the Move program.</td></tr>
<tr><td rowspan="3">Reset</td><td rowspan="3">4</td><td rowspan="3">0x04</td><td rowspan="3">SRC</td><td rowspan="3">KBUS_ST3_7</td><td rowspan="3">0x97</td><td colspan="2">The controller with this bit can detect a module reset.
The bit is true after a reset and is confirmed and false by Reset_Quit.</td></tr>
<tr><td>0:</td><td>No reset since last confirmation.</td></tr>
<tr><td>1:</td><td>A reset has been carried out but not yet confirmed with Reset_Quit. Parameters, data or tables not stored in the EEPROM are no longer valid.</td></tr>
<tr><td></td><td>5</td><td>0x05</td><td></td><td></td><td></td><td colspan="2"></td></tr>
</table>

Name	Bit number		Type	Default		Description	
	Dec.	Hex.		Target/Source	Bit no.		
KBUS_ Active	6	0x06	SRC	-	-	I/O module communication active	
						0:	No I/O module communication for more than 100 ms
						1:	I/O module communication present
	7	0x07					
On_Target	8	0x08	SRC	KBUS_ST2_0 OUT1	0x88 0xA0	The significance of this bit depends on the selected operating mode.	
						Step positioning:	
						0:	The defined position has not been reached.
						1:	The defined position has been reached.
						Move program:	
						0:	The Move program was not terminated by the PROG_STOP drive command.
						1:	The Move program was terminated by the PROG_STOP drive command.
						Reference run:	
						0:	The bit is set to 0 when the reference run is started.
						1:	Not used.
						Jog Mode:	
						0:	The bit is set to 0 when the Jog mode is started.
						1:	Not used.
						Mailbox mode:	
						0:	Moving to a new position.
						1:	The current command has been successfully concluded.

<table>
<tr><th rowspan="2">Name</th><th colspan="2">Bit number</th><th rowspan="2">Type</th><th colspan="2">Default</th><th rowspan="2" colspan="2">Description</th></tr>
<tr><th>Dec.</th><th>Hex.</th><th>Target/Source</th><th>Bit no.</th></tr>
<tr><td rowspan="17">Busy</td><td rowspan="17">9</td><td rowspan="17">0x09</td><td rowspan="17">SRC</td><td rowspan="17">KBUS_ST2_1

LED E</td><td rowspan="17">0x89

0xD0</td><td colspan="2">Busy: The selected operating mode is active and not yet finished. This operating mode may have been discontinued.</td></tr>
<tr><td colspan="2">Step positioning:</td></tr>
<tr><td>0:</td><td>Step positioning not running.</td></tr>
<tr><td>1:</td><td>Step positioning running.</td></tr>
<tr><td colspan="2">Move program:</td></tr>
<tr><td>0:</td><td>The Move program is not running.</td></tr>
<tr><td>1:</td><td>The Move program is running.</td></tr>
<tr><td colspan="2">Reference run:</td></tr>
<tr><td>0:</td><td>Reference run not in operation.</td></tr>
<tr><td>1:</td><td>Reference run in operation.</td></tr>
<tr><td colspan="2">Jog Mode:</td></tr>
<tr><td>0:</td><td>Motor at standstill.</td></tr>
<tr><td>1:</td><td>The Jog mode is running; i.e., the motor has been started using Direction_Pos or Direction_Neg.</td></tr>
<tr><td colspan="2">Mailbox mode:</td></tr>
<tr><td>0:</td><td>No command is active.</td></tr>
<tr><td>1:</td><td>A command is active.</td></tr>
<tr><td colspan="2" style="display:none"></td></tr>
<tr><td rowspan="3">StandStill</td><td rowspan="3">10</td><td rowspan="3">0x0A</td><td rowspan="3">SRC</td><td rowspan="3">KBUS_ST2_2</td><td rowspan="3">0x8A</td><td colspan="2">Drive at standstill, or frequency output at 0.</td></tr>
<tr><td>0:</td><td>Motor is turning.</td></tr>
<tr><td>1:</td><td>Motor at standstill.</td></tr>
<tr><td rowspan="3">On_Speed</td><td rowspan="3">11</td><td rowspan="3">0x0B</td><td rowspan="3">SRC</td><td rowspan="3">KBUS_ST2_3</td><td rowspan="3">0x8B</td><td colspan="2">Running speed achieved.</td></tr>
<tr><td>0:</td><td>The drive has not reached its setpoint speed.</td></tr>
<tr><td>1:</td><td>The drive has reached its setpoint speed.</td></tr>
<tr><td rowspan="3">Direction</td><td rowspan="3">12</td><td rowspan="3">0x0C</td><td rowspan="3">SRC</td><td rowspan="3">KBUS_ST2_4</td><td rowspan="3">0x8C</td><td colspan="2">Direction of rotation is valid only when StandStill is not set to 1.</td></tr>
<tr><td>0:</td><td>Drive moving in the negative direction.</td></tr>
<tr><td>1:</td><td>Drive moving in the positive direction.</td></tr>
</table>

Name	Bit number		Type	Default		Description
	Dec.	Hex.		Target/Source	Bit no.	
Reference_ OK	13	0x0D	SRC	KBUS_ST2_5	0x8D	Set when reference run has been successfully concluded 0: This bit is at 0 when the module is activated. Additionally, it is also set to 0 when the reference run is started. 1: The reference point has been successfully located in the reference run mode.
PreCalc_Ack	14	0x0E	SRC	KBUS_ST2_6	0x8E	Setpoints from Mode 2.2 saved. This bit is set when the setpoint save mode has been requested with PreCalc and precalculation of a movement has been successfully completed. 0: Precalculation not yet performed. 1: Precalculation performed.
Error	15	0x0F	SRC	KBUS_ST2_7 OUT2	0x8F 0xA1	Drive error status. An error can be acknowledged using Error_Quit. 0: No error present for the drive. 1: Error present for the drive.
Ready	16	0x10	SRC	KBUS_ST1_0	0x80	Ready for operation 0: The module is not ready for operation. Either a corresponding request is present via Enable, or an error has resulted in cancellation of Ready. When the bit switches from 1 to 0 the output stage is deactivated, or the output frequency is set to 0. 1: Readiness for operation has been requested via Enable and no error is present.
Stop_N_AC K	17	0x11	SRC	KBUS_ST1_1 LED G	0x81 0xD2	Drive Stop inverted 0: The bit Stop1_N or Stop2_N is set to 0. The motor has stopped (StandStill is set to 1). Start cannot be used to start-up the unit. 1: The bits Stop1_N and Stop2_N are both set to 1, or the drive is braking the unit.

Name	Bit number		Type	Default		Description
	Dec.	Hex.		Target/Source	Bit no.	
Start_ACK	18	0x12	SRC	KBUS_ST1_2	0x82	Start sequence in the operating mode.
						0: This bit is also set to 0 when the Start request is canceled.
						1: The rising edge function is a function of the selected operating mode.
						M_Positioning (step positioning) The currently specified setpoint in the Mode 2.2 process image has been assumed. Movement is made directly to the new target position, even if the drive is already turning, (Change on the fly) when PreCalc_Ack is set, the movement sequence has already been precalculated and will not be started immediately.
						M_Program (MOVE program) The Move program is started. If a Move program is already running, it will be restarted at the first command.
						M_Reference (Reference run) The reference run is being started. If the reference run is still in operation, the (new) setpoints are again accepted and calculated (same procedure as for step positioning). The reference run is then restarted.
						M_Jog (JogMode) No effect. The drive is started using the pushbutton Direction_Pos or Direction_Neg.
						M_DriveByMbx (Mailbox mode) No effect. Various commands can be issued via mailbox as soon as the Mailbox mode is activated.
M_ Positioning_ ACK	19	0x13	SRC	KBUS_ST1_3	0x83	Step positioning mode
						0: The Step positioning mode is not active (selected).
						1: The Step positioning mode is active. Movement is made to the active setpoint on the next rising edge for Start.

Name	Bit number		Type	Default		Description	
	Dec.	Hex.		Target/Source	Bit no.		
M_Program_ ACK	20	0x14	SRC	KBUS_ST1_4	0x84	Move program mode	
						0:	The Move program mode is not active (selected).
						1:	The Move program mode is active. The Move program is started with the first command on the next rising edge for Start.
M_ Reference_ ACK	21	0x15	SRC	KBUS_ST1_5	0x85	Reference run mode	
						0:	The Reference run mode is not active (selected).
						1:	The Reference run mode is active and the drive is started at the setup speed on the next rising edge for Start.
M_Jog_ACK	22	0x16	SRC	KBUS_ST1_6	0x86	Jog mode	
						0:	The Jog mode is not active (selected).
						1:	The Jog mode is active.
M_DriveBy Mbx_ACK	23	0x17	SRC	KBUS_ST1_7	0x87	Mailbox mode	
						0:	The Mailbox mode is not active (selected).
						1:	The Mailbox mode is active.
Brake	24	0x18	SRC	-	-	Brake	
						0:	The brake is not released.
						1:	The brake is released.
ERR_Code	25	0x19	SRC	LED H	0xD3	This bit is normally linked with an LED. If an error is present, it is output as a blink code.	
SetupSpeed_ Active_ACK	26	0x1A	SRC	KBUS_ST3_6	0x96	Setup mode is active. When this bit is set the drive speed is limited to the defined setup speed. Acceleration is not limited. The currently valid acceleration value is applied.	
						0:	Setup mode is not active.
						1:	Setup mode is active.
Program Running	27	0x1B	SRC			A Move program is currently in progress.	
Ramp_Up	28	0x1C	SRC	-	-	Set during the acceleration phase	
Ramp_Down	29	0x1D	SRC	-	-	Set during the deceleration phase	
	30	0x1E					

<table>
<tr><th rowspan="2">Name</th><th colspan="2">Bit number</th><th rowspan="2">Type</th><th colspan="2">Default</th><th colspan="2" rowspan="2">Description</th></tr>
<tr><th>Dec.</th><th>Hex.</th><th>Target/Source</th><th>Bit no.</th></tr>
<tr><td></td><td>31</td><td>0x1F</td><td></td><td></td><td></td><td colspan="2"></td></tr>
<tr><td>Trace_Stored</td><td>32</td><td>0x20</td><td>SRC</td><td>-</td><td>-</td><td colspan="2">This bit is set when all 200 data sets TRACE_VARI/2 have been saved in the Trace table. Trace_Stored is cleared each time trace recording is started (0→1 to Trace_Trigger when Trace_Armed is also set). A trace can be read out using the table commands DLD_START, DLD_CONT and DLD_END.</td></tr>
<tr><td></td><td>33</td><td>0x21</td><td></td><td></td><td></td><td colspan="2"></td></tr>
<tr><td></td><td>34</td><td>0x22</td><td></td><td></td><td></td><td colspan="2"></td></tr>
<tr><td></td><td>35</td><td>0x23</td><td></td><td></td><td></td><td colspan="2"></td></tr>
<tr><td></td><td>36</td><td>0x24</td><td></td><td></td><td></td><td colspan="2"></td></tr>
<tr><td></td><td>37</td><td>0x25</td><td></td><td></td><td></td><td colspan="2"></td></tr>
<tr><td></td><td>38</td><td>0x26</td><td></td><td></td><td></td><td colspan="2"></td></tr>
<tr><td></td><td>39</td><td>0x27</td><td></td><td></td><td></td><td colspan="2"></td></tr>
<tr><td rowspan="3">Err_Range_ Neg</td><td rowspan="3">40</td><td rowspan="3">0x28</td><td rowspan="3">SRC</td><td rowspan="3">-</td><td rowspan="3">-</td><td colspan="2">Moving range exceeded when moving in negative direction.</td></tr>
<tr><td>0:</td><td>The bottom limit for the movement range has not been violated.</td></tr>
<tr><td>1:</td><td>The bottom limit for the movement range has been violated.</td></tr>
<tr><td rowspan="3">Err_Range_ Pos</td><td rowspan="3">41</td><td rowspan="3">0x29</td><td rowspan="3">SRC</td><td rowspan="3">-</td><td rowspan="3">-</td><td colspan="2">Moving range exceeded when moving in positive direction.</td></tr>
<tr><td>0:</td><td>The top limit for the movement range has not been violated.</td></tr>
<tr><td>1:</td><td>The top limit for the movement range has been violated.</td></tr>
<tr><td>Err_Range</td><td>42</td><td>0x2A</td><td>SRC</td><td>-</td><td>-</td><td colspan="2">Parameter is set when it has been detected by Err_Range_Neg, Err_Range_Pos, LimitSwitch_Pos or LimitSwitch_Neg that the permissible movement range has been violated.</td></tr>
<tr><td></td><td>43</td><td>0x2B</td><td></td><td></td><td></td><td colspan="2"></td></tr>
<tr><td></td><td>44</td><td>0x2C</td><td></td><td></td><td></td><td colspan="2"></td></tr>
<tr><td></td><td>45</td><td>0x2D</td><td></td><td></td><td></td><td colspan="2"></td></tr>
<tr><td></td><td>46</td><td>0x2E</td><td></td><td></td><td></td><td colspan="2"></td></tr>
<tr><td>CAM9</td><td>47</td><td>0x2F</td><td>SRC</td><td>-</td><td>-</td><td colspan="2">Camshaft 9</td></tr>
<tr><td>Input1</td><td>48</td><td>0x30</td><td>SRC</td><td>KBUS_ST3_0

Stop1_N</td><td>0x90

0xC2</td><td colspan="2">Input 1</td></tr>
<tr><td>Input2</td><td>49</td><td>0x31</td><td>SRC</td><td>KBUS_ST3_1

Set_Reference</td><td>0x91

0xBC</td><td colspan="2">Input 2</td></tr>
</table>

Name	Bit number		Type	Default		Description
	Dec.	Hex.		Target/Source	Bit no.	
	50	0x32				
	51	0x33				
	52	0x34				
	53	0x35				
	54	0x36				
	55	0x37				
CAM1	56	0x38	SRC	-	-	Camshaft 1
CAM2	57	0x39	SRC	-	-	Camshaft 2
CAM3	58	0x3A	SRC	-	-	Camshaft 3
CAM4	59	0x3B	SRC	-	-	Camshaft 4
CAM5	60	0x3C	SRC	-	-	Camshaft 5
CAM6	61	0x3D	SRC	-	-	Camshaft 6
CAM7	62	0x3E	SRC	-	-	Camshaft 7
CAM8	63	0x3F	SRC	-	-	Camshaft 8
KBUS_ CTRL1_0	64	0x40	SRC	Enable	0xB0	Internal bus control byte 1 bit 0
KBUS_ CTRL1_1	65	0x41	SRC	Stop2_N	0xB1	Internal bus control byte 1 bit 1
KBUS_ CTRL1_2	66	0x42	SRC	Start	0xB2	Internal bus control byte 1 bit 2
KBUS_ CTRL1_3	67	0x43	SRC	M_Positioning	0xB3	Internal bus control byte 1 bit 3
KBUS_ CTRL1_4	68	0x44	SRC	M_Program	0xB4	Internal bus control byte 1 bit 4
KBUS_ CTRL1_5	69	0x45	SRC	M_Reference	0xB5	Internal bus control byte 1 bit 5
KBUS_ CTRL1_6	70	0x46	SRC	M_Jog	0xB6	Internal bus control byte 1 bit 6
KBUS_ CTRL1_7	71	0x47	SRC	M_DriveBxMb x	0xB7	Internal bus control byte 1 bit 7
KBUS_ CTRL2_0	72	0x48	SRC	Freq_Range_S el_0	0xC4	Internal bus control byte 2 bit 0
KBUS_ CTRL2_1	73	0x49	SRC	Freq_Range_S el_1	0xC5	Internal bus control byte 2 bit 1
KBUS_ CTRL2_2	74	0x4A	SRC	Acc_Range_Se l_0	0xC6	Internal bus control byte 2 bit 2
KBUS_ CTRL2_3	75	0x4B	SRC	Acc_Range_Se l_1	0xC7	Internal bus control byte 2 bit 3
KBUS_ CTRL2_4	76	0x4C	SRC	-	-	Internal bus control byte 2 bit 4

Name	Bit number		Type	Default		Description
	Dec.	Hex.		Target/Source	Bit no.	
KBUS_ CTRL2_5	77	0x4D	SRC	-	-	Internal bus control byte 2 bit 5
KBUS_ CTRL2_6	78	0x4E	SRC	PreCalc	0xBD	Internal bus control byte 2 bit 6
KBUS_ CTRL2_7	79	0x4F	SRC	Error_Quit	0xBF	Internal bus control byte 2 bit 7
KBUS_ CTRL3_0	80	0x50	SRC	Set_Actual_Po s	0xC8	Internal bus control byte 3 bit 0
KBUS_ CTRL3_1	81	0x51	SRC	-	-	Internal bus control byte 3 bit 1
KBUS_ CTRL3_2	82	0x52	SRC	Direction_Pos	0xBA	Internal bus control byte 3 bit 2
KBUS_ CTRL3_3	83	0x53	SRC	Direction_Neg	0xBB	Internal bus control byte 3 bit 3
KBUS_ CTRL3_4	84	0x54	SRC	LimitSwitch_P os	0xC0	Internal bus control byte 3 bit 4
KBUS_ CTRL3_5	85	0x55	SRC	LimitSwitch_N eg	0xC1	Internal bus control byte 3 bit 5
KBUS_ CTRL3_6	86	0x56	SRC	SetupSpeed_A ctive	0xBE	Internal bus control byte 3 bit 6
KBUS_ CTRL3_7	87	0x57	SRC	Reset_Quit	0xB9	Internal bus control byte 3 bit 7
	88	0x58				
	89	0x59				
	90	0x5A				
	91	0x5B				
	92	0x5C				
	93	0x5D				
	94	0x5E				
	95	0x5F				
	96	0x60				
	97	0x61				
	98	0x62				
	99	0x63				
	100	0x64				
	101	0x65				
	102	0x66				
	103	0x67				
	104	0x68				

Name	Bit number		Type	Default		Description
	Dec.	Hex.		Target/Source	Bit no.	
	105	0x69				
	106	0x6A				
	107	0x6B				
	108	0x6C				
	109	0x6D				
	110	0x6E				
	111	0x6F				
	112	0x70				
	113	0x71				
	114	0x72				
	115	0x73				
	116	0x74				
	117	0x75				
	118	0x76				
	119	0x77				
	120	0x78				
	121	0x79				
	122	0x7A				
	123	0x7B				
	124	0x7C				
	125	0x7D				
	126	0x7E				
	127	0x7F				
KBUS_ ST1_0	128	0x80	DST/ SRC	Ready	0x10	Internal bus status byte 1 bit 0
KBUS_ ST1_1	129	0x81	DST/ SRC	Stop_N_ACK	0x11	Internal bus status byte 1 bit 1
KBUS_ ST1_2	130	0x82	DST/ SRC	Start_ACK	0x12	Internal bus status byte 1 bit 2
KBUS_ ST1_3	131	0x83	DST/ SRC	M_Positioning _ACK	0x13	Internal bus status byte 1 bit 3
KBUS_ ST1_4	132	0x84	DST/ SRC	M_Program_A CK	0x14	Internal bus status byte 1 bit 4
KBUS_ ST1_5	133	0x85	DST/ SRC	M_Reference_ ACK	0x15	Internal bus status byte 1 bit 5
KBUS_ ST1_6	134	0x86	DST/ SRC	M_Jog_ACK	0x16	Internal bus status byte 1 bit 6

Name	Bit number		Type	Default		Description
	Dec.	Hex.		Target/Source	Bit no.	
KBUS_ ST1_7	135	0x87	DST/ SRC	M_DriveByMb x_ACK	0x17	Internal bus status byte 1 bit 7
KBUS_ ST2_0	136	0x88	DST/ SRC	On_Target	0x08	Internal bus status byte 2 bit 0
KBUS_ ST2_1	137	0x89	DST/ SRC	Busy	0x09	Internal bus status byte 2 bit 1
KBUS_ ST2_2	138	0x8A	DST/ SRC	StandStill	0x0A	Internal bus status byte 2 bit 2
KBUS_ ST2_3	139	0x8B	DST/ SRC	On_Speed	0x0B	Internal bus status byte 2 bit 3
KBUS_ ST2_4	140	0x8C	DST/ SRC	Direction	0x0C	Internal bus status byte 2 bit 4
KBUS_ ST2_5	141	0x8D	DST/ SRC	Reference_OK	0x0D	Internal bus status byte 2 bit 5
KBUS_ ST2_6	142	0x8E	DST/ SRC	PreCalc_ACK	0x0E	Internal bus status byte 2 bit 6
KBUS_ ST2_7	143	0x8F	DST/ SRC	Error	0x0F	Internal bus status byte 2 bit 7
KBUS_ ST3_0	144	0x90	DST/ SRC	Input1	0x30	Internal bus status byte 3 bit 0
KBUS_ ST3_1	145	0x91	DST/ SRC	Input2	0x31	Internal bus status byte 3 bit 1
KBUS_ ST3_2	146	0x92	DST/ SRC	Input3	0x32	Internal bus status byte 3 bit 2
KBUS_ ST3_3	147	0x93	DST/ SRC	Input4	0x33	Internal bus status byte 3 bit 3
KBUS_ ST3_4	148	0x94	DST/ SRC	Input5	0x34	Internal bus status byte 3 bit 4
KBUS_ ST3_5	149	0x95	DST/ SRC	Input6	0x35	Internal bus status byte 3 bit 5
KBUS_ ST3_6	150	0x96	DST/ SRC	SetupSpeed_A ctive_ACK	0x1A	Internal bus status byte 3 bit 6
KBUS_ ST3_7	151	0x97	DST/ SRC	Reset	0x04	Internal bus status byte 3 bit 7
	152	0x98				
	153	0x99				
	154	0x9A				
	155	0x9B				
	156	0x9C				
	157	0x9D				
	158	0x9E				

Name	Bit number		Type	Default		Description		
	Dec.	Hex.		Target/Source	Bit no.			
	159	0x9F						
	160	0xA0						
	161	0xA1						
	162	0xA2						
	163	0xA3						
	164	0xA4						
	165	0xA5						
	166	0xA6						
	167	0xA7						
FILT1	168	0xA8	FILT	ZERO	0x00	Timer / Filter 1	0:	The value for the timer or filter is equal to 0
FILT2	169	0xA9	FILT	ZERO	0x00	Timer / Filter 2		
FILT3	170	0xAA	FILT	ZERO	0x00	Timer / Filter 3		
FILT4	171	0xAB	FILT	ZERO	0x00	Timer / Filter 4		
FILT5	172	0xAC	FILT	ZERO	0x00	Timer / Filter 5	1:	The value for the timer or filter is not equal to 0
FILT6	173	0xAD	FILT	ZERO	0x00	Timer / Filter 6		
FILT7	174	0xAE	FILT	ZERO	0x00	Timer / Filter 7		
FILT8	175	0xAF	FILT	ZERO	0x00	Timer / Filter 8		
Enable	176	0xB0	DST/ SRC	KBUS_CTRL1 _0	0x40	Module enable		
						0:	The module is not enabled. When this bit is reset during ongoing operation, the power output stage is deactivated.	
						1:	The module is enabled and can be started when the corresponding return message is also available in the status.	
Stop2_N	177	0xB1	DST/ SRC	KBUS_CTRL1 _1	0x41	Drive Stop 2 inverted This bit can be used to deactivate the drive from the control system. The acknowledgement is transmitted via Stop_N_ACK bit. Stop1_N and Stop2_N are always taken into account at that bit.		
						0:	Current is supplied to the motor, but it is at standstill. If the motor is still turning it is put into standstill by the STOP acceleration command. The motor cannot be started-up.	
						1:	The drive may be started.	

Name	Bit number		Type	Default		Description
	Dec.	Hex.		Target/Source	Bit no.	
Start	178	0xB2	DST/ SRC	KBUS_CTRL1 _2	0x42	The drive, or frequency output, is started in the selected mode on the positive edge. If the edge is not accepted (in the Jog or Mailbox mode), an error message is generated.
						0→1: The drive is started accordingly on the rising edge.
						M_Positioning (step positioning): Positioning is conducted to the current setpoint given in the process image, Mode 2.2. Movement is made directly to the new target position, even if the drive is already turning. (Change on the fly) A previously calculated movement sequence is started immediately when PreCalc_ACK bit is set.
						M_Program (Move program): The current Move program is started by the first command for the Move program. If a Move program is already running, it will be restarted at the first command.
						M_Reference (Reference run): The reference run is started.
						M_Jog (JogMode): No effect. The drive is started only when the pushbutton Direction_Pos or Direction_Neg is actuated.
						M_DriveByMbx (Mailbox mode): May not be set when the mailbox has not been activated; otherwise an error message is issued. Various commands can be issued via mailbox as soon as the Mailbox mode is activated.
M_ Positioning	179	0xB3	DST/ SRC	KBUS_CTRL1 _3	0x43	Step positioning mode The mailbox may not be active in this mode.
						0: The Step positioning mode is not active (selected).
						1: The Step positioning mode is selected.
M_Program	180	0xB4	DST/ SRC	KBUS_CTRL1 _4	0x44	Move program mode
						0: The Move program mode is not active (selected).
						1: The Move program mode has been selected.

Name	Bit number		Type	Default		Description
	Dec.	Hex.		Target/Source	Bit no.	
M_Reference	181	0xB5	DST/ SRC	KBUS_CTRL1 _5	0x45	Reference run mode 0: The Reference run mode is not active (selected). 1: The Reference run mode has been selected.
M_Jog	182	0xB6	DST/ SRC	KBUS_CTRL1 _6	0x46	Jog mode The drive can be run manually at the setup speed when the Jog mode is active. Control is implemented using Direction_Pos and Direction_Neg. 0: The Jog mode is not active (selected). 1: The Jog mode has been selected.
M_DriveBy Mbx	183	0xB7	DST/ SRC	KBUS_CTRL1 _7	0x47	Mailbox mode In this mode, all movement commands are issued directly via mailbox. 0: The Mailbox mode is not active (selected). 1: The Mailbox mode has been selected.
Enable_Drive	184	0xB8	DST/ SRC	ONE	0x01	Output stage enabled. The output stage can be inhibited directly by deleting this bit. Contrary to Enable, this bit does not have any further effects on internal processing. The output stage is only enabled when Enable has been set and all other enable conditions are fulfilled. This bit is linked to ONE by default. 0: Output stage inhibited. 1: Output stage can be enabled.
Reset_Quit	185	0xB9	DST/ SRC	KBUS_CTRL3 _7	0x57	Reset acknowledgement 0: Function not defined. 1: The Reset signal is reset.

Name	Bit number		Type	Default		Description
	Dec.	Hex.		Target/Source	Bit no.	
Direction_ Pos	186	0xBA	DST/ SRC	KBUS_CTRL3 _2	0x52	Move in positive direction. This bit is required for the Jog mode. When this mode has been selected, the drive is controlled using these bits. In the Reference run mode this bit defines that the reference switch be searched for in a positive direction.
						0: Drive not to move in a positive direction.
						1: Drive should move in a positive direction. The drive is deactivated when the bit Direction_Neg is set at the same time.
Direction_ Neg	187	0xBB	DST/ SRC	KBUS_CTRL3 _3	0x53	Move in negative direction. This bit is required for the Jog mode. When this mode has been selected, the drive is controlled using these bits. In the Reference run mode this bit defines that the reference switch be searched for in a negative direction.
						0: Drive not to move in a negative direction.
						1: Drive should move in a negative direction. The drive is deactivated when the bit Direction_Pos is set at the same time.
Set_ Reference	188	0xBC	DST/ SRC	Input2	0x31	Reference input Input 2 is set to this bit in the standard configuration.
						0: The reference switch is not actuated.
						1: The reference switch is actuated.

Name	Bit number		Type	Default		Description
	Dec.	Hex.		Target/Source	Bit no.	
PreCalc	189	0xBD	DST/ SRC	KBUS_CTRL2 _6	0x4E	Save setpoints from Module 2.2 and, where applicable, calculate a movement sequence in advance.
						0: Each setpoint that is transmitted via cyclic telegram traffic must be accepted and processed. Any precalculated movement sequence is rejected. A movement sequence can be calculated and started using Start.
						1: The setpoints from the cyclic telegram traffic are ignored and the setpoint saved for the 0→1 edge used instead. If the starting speed is zero, a movement sequence will be calculated in advance using this setpoint; this sequence can then be started with the normal delay using Start.
Setup_ Speed_Active	190	0xBE	DST/ SRC	KBUS_CTRL3 _6	0x56	Setup mode selected. When the bit SetupSpeed_Active_ACK is set the drive speed is limited to the defined setup speed.
						0: Setup mode not selected.
						1: Setup mode selected.
Error_Quit	191	0xBF	DST/ SRC	KBUS_CTRL2 _7	0x4F	Acknowledge error. All errors that are present are acknowledged at the rising edge from 0 to 1. After acknowledgement, the error switches to 0, or a new error is present.
LimitSwitch_ Pos	192	0xC0	DST/ SRC	KBUS_CTRL3 _4		Limit switch input on movement in positive direction. This bit is linked to the internal bus.
						0: The positive direction limit switch is not actuated.
						1: The positive direction limit switch is actuated. The drive is being run down.

Name	Bit number		Type	Default		Description	
	Dec.	Hex.		Target/Source	Bit no.		
LimitSwitch_Neg	193	0xC1	DST/SRC	KBUS_CTRL3_5		Limit switch input on movement in negative direction. This bit is linked to the internal bus.	
						0:	The negative direction limit switch is not actuated.
						1:	The negative direction limit switch is actuated. The drive is being run down.
Stop1_N	194	0xC2	DST/SRC	Input1	0x30	Drive Stop 1 inverted This bit is linked to Input 1 of the module. The return message is transmitted via Stop_N_ACK bit. Stop1_N and Stop2_N are always taken into account at that bit.	
						0:	Current is being supplied to the motor, but it is at standstill. If the motor is still turning it is put into standstill by the STOP acceleration command. The motor cannot be started-up.
						1:	The drive may be started.
Brake_Manual	195	0xC3	DST/SRC	ZERO	0x00	Manual control of brake	

<table>
<tr><th rowspan="2">Name</th><th colspan="2">Bit number</th><th rowspan="2">Type</th><th colspan="2">Default</th><th rowspan="2">Description</th></tr>
<tr><th>Dec.</th><th>Hex.</th><th>Target/Source</th><th>Bit no.</th></tr>
<tr><td>Freq_Range_ Sel_0</td><td>196</td><td>0xC4</td><td>DST/ SRC</td><td>KBUS_CTRL2 _0</td><td>0x48</td><td rowspan="2">Configuration of velocity prescaler:
The Freq-Prescaler prescaler for velocity is set using these two bits. These values are accepted only when Enable is set to 0.
<table>
<tr><td>Freq_ Range_ Sel_0</td><td>Freq_ Range_ Sel_1</td><td></td></tr>
<tr><td>0</td><td>0</td><td>The Freq_Prescaler prescaler is loaded with the parameter Freq_Div from the current configuration data set (*).</td></tr>
<tr><td>0</td><td>1</td><td>Freq_Prescaler = 80
Fmax = 25 kHz</td></tr>
<tr><td>1</td><td>0</td><td>Freq_Prescaler = 20
Fmax = 100 kHz</td></tr>
<tr><td>1</td><td>1</td><td>Freq_Prescaler = 4
Fmax = 500 kHz</td></tr>
<tr><td colspan="3">(*) If the parameter Freq_Div in the configuration data set has been assigned a value of zero, the prescaler will be set to Freq_Prescaler = 200 (Fmax = 10 kHz) for Freq_Range_Sel_0 = 0 and Freq_Range_Sel_1 = 0.</td></tr>
</table></td></tr>
<tr><td>Freq_Range_ Sel_1</td><td>197</td><td>0xC5</td><td>DST/ SRC</td><td>KBUS_CTRL2 _1</td><td>0x49</td></tr>
</table>

<table>
<tr><th rowspan="2">Name</th><th colspan="2">Bit number</th><th rowspan="2">Type</th><th colspan="2">Default</th><th rowspan="2">Description</th></tr>
<tr><th>Dec.</th><th>Hex.</th><th>Target/Source</th><th>Bit no.</th></tr>
<tr><td>Acc_Range_
Sel_0</td><td>198</td><td>0xC6</td><td>DST/
SRC</td><td>KBUS_CTRL2
_0</td><td>0x48</td><td rowspan="2">Configuration factor Acceleration.
These two bits are used to set the ACC_Multiplier factor for acceleration.
These values are accepted only when Enable is set to 0.
<table>
<tr><td>Acc_
Range_
Sel_0</td><td>Acc_
Range_
Sel_1</td><td></td></tr>
<tr><td>0</td><td>0</td><td>The factor Acc_Multiplier is loaded with the parameter Acc_Fact from the current configuration data set (*).</td></tr>
<tr><td>0</td><td>1</td><td>Acc_Multiplier = 80
T = 760 ms</td></tr>
<tr><td>1</td><td>0</td><td>Acc_Multiplier = 800
T = 76 ms</td></tr>
<tr><td>1</td><td>1</td><td>Acc_Multiplier = 8000
T = 7.6 ms</td></tr>
</table>
(*) If the parameter Acc_Fact in the configuration data set has been assigned a value of zero, the factor is set to Acc_Multiplier = 8 (T = 7.6s) for Acc_Range_Sel_0 = 0 and Acc_Range_Sel_1 = 0.</td></tr>
<tr><td>Acc_Range_
Sel_1</td><td>199</td><td>0xC7</td><td>DST/
SRC</td><td>KBUS_CTRL2
_1</td><td>0x49</td></tr>
<tr><td>Set_Actual_
POS</td><td>200</td><td>0xC8</td><td>DST/
SRC</td><td>KBUS_CTRL3
_0</td><td>0x50</td><td>The actual value is set to the reference position (configuration parameter Reference_Offset) on a rising edge from the bit Set_Actual_POS.
This function cannot be performed while a positioning run is ongoing.</td></tr>
<tr><td></td><td>201</td><td>0xC9</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>202</td><td>0xCA</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>203</td><td>0xCB</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>204</td><td>0xCC</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>205</td><td>0xCD</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>206</td><td>0xCE</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>207</td><td>0xCF</td><td></td><td></td><td></td><td></td></tr>
<tr><td>LED E</td><td>208</td><td>0xD0</td><td>DST/
SRC</td><td>Busy</td><td>0x09</td><td>LED E</td></tr>
<tr><td>LED F</td><td>209</td><td>0xD1</td><td>DST/
SRC</td><td>M_Program_A
CK</td><td>0x1B</td><td>LED F</td></tr>
</table>

Name	Bit number		Type	Default		Description
	Dec.	Hex.		Target/Source	Bit no.	
LED G	210	0xD2	DST/ SRC	On_Speed	0x0B	LED G
LED H	211	0xD3	DST/ SRC	ERR_Code	0x19	LED H
	212	0xD4				
	213	0xD5				
	214	0xD6				
	215	0xD7				
	216	0xD8				
	217	0xD9				
	218	0xDA				
	219	0xDB				
	220	0xDC				
	221	0xDD				
Trace_ Trigger	222	0xDE	DST/ SRC	Busy	0x09	A trace is started with a positive edge when Trace_Armed is set. The Trace_Var1/2 variables given in the configuration are recorded using the time frame specified in Trace_MsecCycleTime.
Trace_ Armed	223	0xDF	DST/ SRC	MONE	0x03	Activation of trace.
	224	0xE0				
	225	0xE1				
	226	0xE2				
	227	0xE3				
	228	0xE4				
	229	0xE5				
	230	0xE6				
	231	0xE7				
	232	0xE8				
	233	0xE9				
	234	0xEA				
	235	0xEB				
	236	0xEC				
	237	0xED				
	238	0xEE				
	239	0xEF				

Name	Bit number		Type	Default		Description
	Dec.	Hex.		Target/Source	Bit no.	
	240	0xF0				
	241	0xF1				
	242	0xF2				
	243	0xF3				
	244	0xF4				
	245	0xF5				
	246	0xF6				
	247	0xF7				
	248	0xF8				
	249	0xF9				
	250	0xFA				
	251	0xFB				
	252	0xFC				
	253	0xFD				
	254	0xFE				
	255	0xFF				

3.5 Configuration Variables

<table>
<tr><th rowspan="2">Configuration variable</th><th colspan="2">Address</th><th rowspan="2">Data type</th><th rowspan="2">Default</th><th rowspan="2">Range</th><th colspan="2" rowspan="2">Description</th></tr>
<tr><th>Dec.</th><th>Hex.</th></tr>
<tr><td>User_Conf_Id</td><td>0</td><td>0x00</td><td>UINT16</td><td>0</td><td>0 ... 50000</td><td colspan="2">The user can freely assign data set numbers. However, numbers greater then 50000 are reserved.</td></tr>
<tr><td>ConfVersion</td><td>2</td><td>0x02</td><td>UINT8</td><td>4</td><td>0 ... 254</td><td colspan="2">Configuration version number</td></tr>
<tr><td rowspan="4">Application_ Selector</td><td rowspan="4">3</td><td rowspan="4">0x03</td><td rowspan="4">UINT8</td><td rowspan="4">1</td><td rowspan="4">0 ... 2</td><td colspan="2">Switching of applications. The appropriate process image is activated when a new application is selected.</td></tr>
<tr><td>0:</td><td>Reserved</td></tr>
<tr><td>1:</td><td>Positioning controller</td></tr>
<tr><td>2:</td><td>Velocity control
Frequency output</td></tr>
<tr><td>Freq_Div</td><td>4</td><td>0x04</td><td>UINT16</td><td>200</td><td>4 ... 65535</td><td colspan="2">Sets the prescaler for maximum velocity</td></tr>
<tr><td>Acc_Fact</td><td>6</td><td>0x06</td><td>UINT16</td><td>80</td><td>1 ... 65535</td><td colspan="2">Sets factor for maximum acceleration</td></tr>
<tr><td>Current</td><td>14</td><td>0x0E</td><td>UINT8</td><td>10</td><td>0 ... 10</td><td colspan="2">Motor rated current in [0.1 A]</td></tr>
<tr><td>Current_Ratio_ StandStill</td><td>15</td><td>0x0F</td><td>UINT8</td><td>33</td><td>0 ... 150</td><td colspan="2">Current factor at standstill in [%], based on motor rated current "Current"</td></tr>
<tr><td>Current_Ratio_ RampUp</td><td>16</td><td>0x10</td><td>UINT8</td><td>120</td><td>0 ... 150</td><td colspan="2">Current factor for ramp-up in [%], based on motor rated current "Current"</td></tr>
<tr><td>Current_Ratio_ Drive</td><td>17</td><td>0x11</td><td>UINT8</td><td>50</td><td>0 ... 150</td><td colspan="2">Current factor for drive in [%], based on motor rated current "Current"</td></tr>
<tr><td>Current_Ratio_ RampDown</td><td>18</td><td>0x12</td><td>UINT8</td><td>90</td><td>0 ... 150</td><td colspan="2">Current factor for ramp-down in [%], based on motor rated current "Current"</td></tr>
</table>

Configuration variable	Address		Data type	Default	Range	Description
	Dec.	Hex.				
HwSwConfig	19	0x13	UINT8	0		Bit 0 … 1: Reserved
						Bit 2: Drive_Direction (Direction of rotation inversion)
						0: Output signal processed directly
						1: Output signal: Direction of rotation inverted
						Bit 3 … 6: Reserved
						Bit 7: Program_Autostart (Move program Autostart – Normal mode)
						0: Move program only activated via Move program or Mailbox mode.
						1: Move program activated immediately after startup, see description.
Pos_Mult	20	0x14	UINT16	1	1 ... 65535	Scaling factors for positions
Pos_Div	22	0x16	UINT16	1	1 ... 65535	Scaling factors for positions
	24	0x18				
	26	0x1A				
Speed_Mult	28	0x1C	UINT16	1	1 ... 65535	Scaling factors for speed
Speed_Div	30	0x1E	UINT16	1	1 ... 65535	Scaling factors for speed
Acc_Mult	32	0x20	UINT16	1	1 ... 65535	Scaling factors for acceleration
Acc_Div	34	0x22	UINT16	1	1 ... 65535	Scaling factors for acceleration
Reserved_36	36	0x24	UINT16	0		Reserved
Reserved_38	38	0x26	UINT16	0		Reserved

Configuration variable	Address		Data type	Default	Range	Description
	Dec.	Hex.				
Speed	40	0x28	INT16	10	1 ... 25000	Default speed
Speed_Limit	42	0x2A	INT16	25000	1 ... 25000	Default maximum speed; the drive is switched off if this speed is exceeded
SetupSpeed	44	0x2C	INT16	100	1 ... 25000	Default setup speed; the current moving speed is used when this parameter is 0.
Acceleration_ Stop_Fast	46	0x2E	INT16	1000	0 ... 32767	Default acceleration for STOP mode; the current acceleration is used when this parameter is 0.
Acceleration_ RampUp	48	0x30	INT16	10	0 ... 32767	Default acceleration for acceleration phase
Acceleration_ RampDown	50	0x32	INT16	10	0 ... 32767	Default acceleration for deceleration phase
Acceleration_ RampUp_Param	52	0x34	INT32	300	0 ... 16777216	Default acceleration time or acceleration path
Acceleration_ RampDown_ Param	56	0x38	INT32	300	0 ... 16777216	Default deceleration time or deceleration path

Configuration variable	Address		Data type	Default	Range	Description	
	Dec.	Hex.					
Acceleration_ Modes	60	0x3C	UINT8	0		Bit 0 ... 1: AccType (Acceleration type)	
						0:	Constant acceleration
						1:	Linear rise in acceleration; the period for acceleration increase is Acceleration_RampUp_Param
						2:	Sin^2 rise in acceleration; the period for acceleration increase is Acceleration_RampUp_Param
						3:	Reserved
						Bit 2 ... 3: AccParam (Acceleration parameter)	
						0:	No modification
						1:	Acceleration_RampUp_Param interpreted as the acceleration period
						2:	Acceleration_RampUp_Param interpreted as the acceleration path
						3:	Reserved
						Bit 4 ... 5: DecType (Deceleration type)	
						0:	Constant acceleration
						1:	Linear rise in acceleration; the period for acceleration increase is Acceleration_RampUp_Param
						2:	Sin^2 rise in acceleration; the period for acceleration increase is Acceleration_RampUp_Param
						3:	Reserved
						Bit 6..7: DecParam (Deceleration parameter)	
						0:	No modification
						1:	Acceleration_RampUp_Param interpreted as the acceleration period
						2:	Acceleration_RampUp_Param interpreted as the acceleration path
						3:	Reserved

Configuration variable	Address		Data type	Default	Range	Description
	Dec.	Hex.				
Current_Ratio_ Stop	61	0x3D	UINT8	100	0 ... 150	Current factor for STOP mode in [%], based on motor rated current "Current"
Setup Acceleration	62	0x3E	UINT16	10	0 ... 32767	Acceleration for JOG and Reference mode
Rotary_Axis_ Period	64	0x40	INT32	0	0 ... 16777216	Sets the period P for a rotary axis; zero is entered here for a linear axis.
Drive_Range_ Neg	68	0x44	INT32	-0x7fffff	0	Acceptable moving range in negative direction
Drive_Range_ Pos	72	0x48	INT32	0x7fffff	0	Acceptable moving range in positive direction
Reserved_76	76	0x4C	UINT32	0		Reserved
Reserved_80	80	0x50	UINT32	0		Reserved
	84	0x54				
Camshaft_Ch9_ Start	88	0x58	INT32	0	±8388607	Camshaft channel 9 position, starting edge
Camshaft_Ch9_ Period	92	0x5B	INT32	100	1 ... 8388607	Camshaft channel 9 period
Camshaft_Ch9_ Pulsewidth	96	0x60	INT32	50	0 ... 8388607	Camshaft channel 9 pulse width
Braketime_Turn_ On	100	0x64	UINT32	0	1 ... 8388607	Starting time for brake in [ms]
Braketime_Turn_ Off	104	0x68	UINT32	0	0 ... 8388607	Switch-off time for brake in [ms]
Reference_Offset	108	0x6C	UINT32	0	±8388607	Position of reference switch

Configuration variable	Address		Data type	Default	Range	Description
	Dec.	Hex.				
Reference_Mode	112	0x70	UINT8	0		Mode for reference run at the start of a reference run via control byte C1_M_Reference (for starting a reference run using the move command START_REFERENCING, call parameters are used, but **not** the following configuration bits)
						Bit 0:
						0: Reference run to reference switch
						1: Reference run to limit switch
						Bit 1:
						0: Reference run to negative end of a reference switch
						1: Reference run to positive end of a reference switch
						Bit 2..7: Reserved
Error_ Notification	113	0x71	UINT8	0		Bit 0: Mode (Enable group error for acyclic diagnostics)
						0: Errors are not reported via internal bus status bit S0.6
						1: Errors are reported via internal bus status bit S0.6
						Bit 1..7: Reserved
Reserved_114	114	0x72	INT16	0		Reserved
Reserved_116	116	0x74	INT32	0		Reserved
Reserved_120	120	0x78	INT32	0		Reserved
Reserved_124	124	0x7C	INT32	0		Reserved
Ptr_KBUS_ ST1_0	128	0x80	UINT8	0x10	0 ... 255	Source for linkable bit 0x80
Ptr_KBUS_ ST1_1	129	0x81	UINT8	0x11	0 ... 255	Source for linkable bit 0x81
Ptr_KBUS_ ST1_2	130	0x82	UINT8	0x12	0 ... 255	Source for linkable bit 0x82
Ptr_KBUS_ ST1_3	131	0x83	UINT8	0x13	0 ... 255	Source for linkable bit 0x83
Ptr_KBUS_ ST1_4	132	0x84	UINT8	0x14	0 ... 255	Source for linkable bit 0x84
Ptr_KBUS_ ST1_5	133	0x85	UINT8	0x15	0 ... 255	Source for linkable bit 0x85

Configuration variable	Address		Data type	Default	Range	Description
	Dec.	Hex.				
Ptr_KBUS_ ST1_6	134	0x86	UINT8	0x16	0 ... 255	Source for linkable bit 0x86
Ptr_KBUS_ ST1_7	135	0x87	UINT8	0x17	0 ... 255	Source for linkable bit 0x87
Ptr_KBUS_ ST2_0	136	0x88	UINT8	0x08	0 ... 255	Source for linkable bit 0x88
Ptr_KBUS_ ST2_1	137	0x89	UINT8	0x09	0 ... 255	Source for linkable bit 0x89
Ptr_KBUS_ ST2_2	138	0x8A	UINT8	0x0A	0 ... 255	Source for linkable bit 0x8A
Ptr_KBUS_ ST2_3	139	0x8B	UINT8	0x0B	0 ... 255	Source for linkable bit 0x8B
Ptr_KBUS_ ST2_4	140	0x8C	UINT8	0x0C	0 ... 255	Source for linkable bit 0x8C
Ptr_KBUS_ ST2_5	141	0x8D	UINT8	0x0D	0 ... 255	Source for linkable bit 0x8D
Ptr_KBUS_ ST2_6	142	0x8E	UINT8	0x0E	0 ... 255	Source for linkable bit 0x8E
Ptr_KBUS_ ST2_7	143	0x8F	UINT8	0x0F	0 ... 255	Source for linkable bit 0x8F
Ptr_KBUS_ ST3_0	144	0x90	UINT8	0x30	0 ... 255	Source for linkable bit 0x90
Ptr_KBUS_ ST3_1	145	0x91	UINT8	0x31	0 ... 255	Source for linkable bit 0x91
Ptr_KBUS_ ST3_2	146	0x92	UINT8	0x32	0 ... 255	Source for linkable bit 0x92
Ptr_KBUS_ ST3_3	147	0x93	UINT8	0x33	0 ... 255	Source for linkable bit 0x93
Ptr_KBUS_ ST3_4	148	0x94	UINT8	0x34	0 ... 255	Source for linkable bit 0x94
Ptr_KBUS_ ST3_5	149	0x95	UINT8	0x35	0 ... 255	Source for linkable bit 0x95
Ptr_KBUS_ ST3_6	150	0x96	UINT8	0x1A	0 ... 255	Source for linkable bit 0x96
Ptr_KBUS_ ST3_7	151	0x97	UINT8	0x04	0 ... 255	Source for linkable bit 0x97
Reserved_152	152	0x98	UINT8	0x00	0 ... 255	Source for linkable bit Bit 0x98
Reserved_153	153	0x99	UINT8	0x00	0 ... 255	Source for linkable bit 0x99
Reserved_154	154	0x9A	UINT8	0x00	0 ... 255	Source for linkable bit it 0x9A
Reserved_155	155	0x9B	UINT8	0x00	0 ... 255	Source for linkable bit 0x9B
Reserved_156	156	0x9C	UINT8	0x00	0 ... 255	Source for linkable bit 0x9C
Reserved_157	157	0x9D	UINT8	0x00	0 ... 255	Source for linkable bit 0x9D

Configuration variable	Address		Data type	Default	Range	Description
	Dec.	Hex.				
Reserved_158	158	0x9E	UINT8	0x00	0 ... 255	Source for linkable bit 0x9E
Reserved_159	159	0x9F	UINT8	0x00	0 ... 255	Source for linkable bit 0x9F
Ptr_OUT1	160	0xA0	UINT8	0x08	0 ... 255	Source for linkable bit 0xA0
Ptr_OUT2	161	0xA1	UINT8	0x0F	0 ... 255	Source for linkable bit 0xA1
Reserved_162	162	0xA2	UINT8	0x00	0 ... 255	Source for linkable bit 0xA2
Reserved_163	163	0xA3	UINT8	0x00	0 ... 255	Source for linkable bit 0xA3
Reserved_164	164	0xA4	UINT8	0x00	0 ... 255	Source for linkable bit 0xA4
Reserved_165	165	0xA5	UINT8	0x00	0 ... 255	Source for linkable bit 0xA5
Reserved_166	166	0xA6	UINT8	0x00	0 ... 255	Source for linkable bit 0xA6
Reserved_167	167	0xA7	UINT8	0x00	0 ... 255	Source for linkable bit 0xA7
Ptr_FILT1	168	0xA8	UINT8	0x00	0 ... 255	Source for linkable bit 0xA8
Ptr_FILT2	169	0xA9	UINT8	0x00	0 ... 255	Source for linkable bit 0xA9
Ptr_FILT3	170	0xAA	UINT8	0x00	0 ... 255	Source for linkable bit 0xAA
Ptr_FILT4	171	0xAB	UINT8	0x00	0 ... 255	Source for linkable bit 0xAB
Ptr_FILT5	172	0xAC	UINT8	0x00	0 ... 255	Source for linkable bit 0xAC
Ptr_FILT6	173	0xAD	UINT8	0x00	0 ... 255	Source for linkable bit 0xAD
Ptr_FILT7	174	0xAE	UINT8	0x00	0 ... 255	Source for linkable bit 0xAE
Ptr_FILT8	175	0xAF	UINT8	0x00	0 ... 255	Source for linkable bit 0xAF
Ptr_Enable	176	0xB0	UINT8	0x40	0 ... 255	Source for linkable bit 0xB0
Ptr_Stop2_N	177	0xB1	UINT8	0x41	0 ... 255	Source for linkable bit 0xB1
Ptr_Start	178	0xB2	UINT8	0x42	0 ... 255	Source for linkable bit 0xB2
Ptr_M_ Positioning	179	0xB3	UINT8	0x43	0 ... 255	Source for linkable bit 0xB3
Ptr_M_Program	180	0xB4	UINT8	0x44	0 ... 255	Source for linkable bit 0xB4
Ptr_M_Reference	181	0xB5	UINT8	0x45	0 ... 255	Source for linkable bit 0xB5
Ptr_M_Jog	182	0xB6	UINT8	0x46	0 ... 255	Source for linkable bit 0xB6
Ptr_M_ DriveByMbx	183	0xB7	UINT8	0x47	0 ... 255	Source for linkable bit 0xB7
Ptr_Enable_ Drive	184	0xB8	UINT8	0x01	0 ... 255	Source for linkable bit 0xB8
Ptr_Reset_ Quit	185	0xB9	UINT8	0x57	0 ... 255	Source for linkable bit 0xB9
Ptr_Direction_ Pos	186	0xBA	UINT8	0x52	0 ... 255	Source for linkable bit 0xBA
Ptr_Direction_ Neg	187	0xBB	UINT8	0x53	0 ... 255	Source for linkable bit 0xBB
Ptr_Set_ Reference	188	0xBC	UINT8	0x31	0 ... 255	Source for linkable bit 0xBC

Configuration variable	Address		Data type	Default	Range	Description
	Dec.	Hex.				
Ptr_PreCalc	189	0xBD	UINT8	0x4E	0 ... 255	Source for linkable bit 0xBD
Ptr_SetupSpeed_ Active	190	0xBE	UINT8	0x56	0 ... 255	Source for linkable bit 0xBE
Ptr_Error_Quit	191	0xBF	UINT8	0x4F	0 ... 255	Source for linkable bit 0xBF
Ptr_ LimitSwitch_ Pos	192	0xC0	UINT8	0x54	0 ... 255	Source for linkable bit 0xC0
Ptr_ LimitSwitch_ Neg	193	0xC1	UINT8	0x55	0 ... 255	Source for linkable bit 0xC1
Ptr_Stop1_N	194	0xC2	UINT8	0x30	0 ... 255	Source for linkable bit 0xC2
Reserved_195	195	0xC3	UINT8	0x00	0 ... 255	Source for linkable bit it 0xC3
Ptr_Freq_Range_ Sel_0	196	0xC4	UINT8	0x48	0 ... 255	Source for linkable bit 0xC4
Ptr_Freq_Range_ Sel_1	197	0xC5	UINT8	0x49	0 ... 255	Source for linkable bit 0xC5
Ptr_Acc_Range_ Sel_0	198	0xC6	UINT8	0x4A	0 ... 255	Source for linkable bit 0xC6
Ptr_Acc_Range_ Sel_1	199	0xC7	UINT8	0x4B	0 ... 255	Source for linkable bit 0xC7
Ptr_Set_Actual_ POS	200	0xC8	UINT8	0x50	0 ... 255	Source for linkable bit 0xC8
Reserved_201	201	0xC9	UINT8	0x00	0 ... 255	Source for linkable bit 0xC9
Reserved_202	202	0xCA	UINT8	0x00	0 ... 255	Source for linkable bit 0xCA
Reserved_203	203	0xCB	UINT8	0x00	0 ... 255	Source for linkable bit 0xCB
Reserved_204	204	0xCC	UINT8	0x00	0 ... 255	Source for linkable bit 0xCC
Reserved_205	205	0xCD	UINT8	0x00	0 ... 255	Source for linkable bit 0xCD
Reserved_206	206	0xCE	UINT8	0x00	0 ... 255	Source for linkable bit 0xCE
Reserved_207	207	0xCF	UINT8	0x00	0 ... 255	Source for linkable bit 0xCF
Ptr_LED_E	208	0xD0	UINT8	0x09	0 ... 255	Source for linkable bit 0xD0
Ptr_LED_F	209	0xD1	UINT8	0x1B	0 ... 255	Source for linkable bit 0xD1
Ptr_LED_G	210	0xD2	UINT8	0x11	0 ... 255	Source for linkable bit 0xD2
Ptr_LED_H	211	0xD3	UINT8	0x19	0 ... 255	Source for linkable bit 0xD3
Reserved_212	212	0xD4	UINT8	0x00	0 ... 255	Source for linkable bit 0xD4
Reserved_213	213	0xD5	UINT8	0x00	0 ... 255	Source for linkable bit 0xD5
Reserved_214	214	0xD6	UINT8	0x00	0 ... 255	Source for linkable bit 0xD6
Reserved_215	215	0xD7	UINT8	0x00	0 ... 255	Source for linkable bit 0xD7
Reserved_216	216	0xD8	UINT8	0x00	0 ... 255	Source for linkable bit 0xD8
Reserved_217	217	0xD9	UINT8	0x00	0 ... 255	Source for linkable bit 0xD9

Configuration variable	Address		Data type	Default	Range	Description
	Dec.	Hex.				
Reserved_218	218	0xDA	UINT8	0x00	0 ... 255	Source for linkable bit 0xDA
Reserved_219	219	0xDB	UINT8	0x00	0 ... 255	Source for linkable bit 0xDB
Reserved_220	220	0xDC	UINT8	0x00	0 ... 255	Source for linkable bit 0xDC
Reserved_221	221	0xDD	UINT8	0x00	0 ... 255	Source for linkable bit 0xDD
Ptr_Trace_ Trigger	222	0xDE	UINT8	0x09	0 ... 255	Source for linkable bit 0xDE
Ptr_Trace_ Armed	223	0xDF	UINT8	0x03	0 ... 255	Source for linkable bit 0xDF
Filter1_Function	224	0xE0	UINT8	0	0 ... 11	Filter Function:
						0: No filtering
						1: Inversion
						2: Detection of starting edge
						3: Low pass
						4: Pulse extension
						5: One-shot
						6: Delay
						7: Math
						8: Incrementing counter
						9: Incrementing counter to zero
						10: Decrementing counter
						11: Decrementing counter to zero
Filter2_Function	225	0xE1	UINT8	0	0 ... 11	Function of filter: see Filter1_Function
Filter3_Function	226	0xE2	UINT8	0	0 ... 11	Function of filter: see Filter1_Function
Filter4_Function	227	0xE3	UINT8	0	0 ... 11	Function of filter: see Filter1_Function
Filter5_Function	228	0xE4	UINT8	0	0 ... 11	Function of filter: see Filter1_Function
Filter6_Function	229	0xE5	UINT8	0	0 ... 11	Function of filter: see Filter1_Function
Filter7_Function	230	0xE6	UINT8	0	0 ... 11	Function of filter: see Filter1_Function
Filter8_Function	231	0xE7	UINT8	0	0 ... 11	Function of filter: see Filter1_Function
Filter1_Time	232	0xE8	UINT32	0	0 ... 16777215	Filter Time constant in [ms]
Filter2_Time	236	0xEB	UINT32	0	0 ... 16777215	Filter Time constant in [ms]

Configuration variable	Address		Data type	Default	Range	Description
	Dec.	Hex.				
Filter3_Time	240	0xF0	UINT32	0	0 ... 16777215	Filter Time constant in [ms]
Filter4_Time	244	0xF4	UINT32	0	0 ... 16777215	Filter Time constant in [ms]
Filter5_Time	248	0xF8	UINT32	0	0 ... 16777215	Filter Time constant in [ms]
FIlter6_Time	252	0xFC	UINT32	0	0 ... 16777215	Filter Time constant in [ms]
Filter7_Time	256	0x100	UINT32	0	0 ... 16777215	Filter Time constant in [ms]
Filter8_Time	260	0x104	UINT32	0	0 ... 16777215	Filter Time constant in [ms]
TraceVar1	264	0x108	UINT32	1		Variable number 1 for trace memory
TraceVar2	268	0x10C	UINT32	2		Variable number 2 for trace memory
TraceMsecCycle Time	272	0x110	UINT32	1	0 ... 16777215	Cycle time for recording the variables given in TraceVar1/2 in [ms]
Reserved_276	276	0x114	UINT32	0		Reserved
Reserved_280	280	0x118	UINT32	0		Reserved
Reserved_284	284	0x11C	UINT32	0		Reserved
Reserved_288	288	0x120	UINT32	0		Reserved
Reserved_292	292	0x124	UINT32	0		Reserved
Reserved_296	296	0x128	UINT32	0		Reserved
Reserved_300	300	0x12C	UINT32	0		Reserved
Reserved_304	304	0x130	UINT32	0		Reserved
Reserved_308	308	0x134	UINT32	0		Reserved
Reserved_312	312	0x138	UINT32	0		Reserved
Reserved_316	316	0x13C	UINT32	0		Reserved
Reserved_320	320	0x140	UINT32	0		Reserved
Reserved_324	324	0x144	UINT32	0		Reserved
Reserved_328	328	0x148	UINT32	0		Reserved
Reserved_332	332	0x14C	UINT32	0		Reserved
Reserved_336	336	0x150	UINT32	0		Reserved
Reserved_340	340	0x154	UINT32	0		Reserved
Reserved_344	344	0x158	UINT32	0		Reserved
Reserved_348	348	0x15C	UINT32	0		Reserved
Reserved_352	352	0x160	UINT32	0		Reserved
Reserved_356	356	0x164	UINT32	0		Reserved

Configuration variable	Address		Data type	Default	Range	Description
	Dec.	Hex.				
Reserved_360	360	0x168	UINT32	0		Reserved
Reserved_364	364	0x16C	UINT32	0		Reserved
Reserved_368	368	0x170	UINT32	0		Reserved
Reserved_372	372	0x174	UINT32	0		Reserved
Reserved_376	376	0x178	UINT32	0		Reserved
Reserved_380	380	0x17C	UINT32	0		Reserved
Reserved_384	384	0x180	UINT32	0		Reserved
Reserved_388	388	0x184	UINT32	0		Reserved
Reserved_392	392	0x188	UINT32	0		Reserved
Reserved_396	396	0x18C	UINT32	0		Reserved
Reserved_400	400	0x190	UINT32	0		Reserved
Reserved_404	404	0x194	UINT32	0		Reserved
Reserved_408	408	0x198	UINT32	0		Reserved
Reserved_412	412	0x19C	UINT32	0		Reserved
Reserved_416	416	0x1A0	UINT32	0		Reserved
Reserved_420	420	0x1A4	UINT32	0		Reserved
Reserved_424	424	0x1A8	UINT32	0		Reserved
Reserved_428	428	0x1AC	UINT32	0		Reserved
Reserved_432	432	0x1B0	UINT32	0		Reserved
Reserved_436	436	0x1B4	UINT32	0		Reserved
Reserved_440	440	0x1B8	UINT32	0		Reserved
Reserved_444	444	0x1BC	UINT32	0		Reserved
Reserved_448	448	0x1C0	UINT32	0		Reserved
Reserved_452	452	0x1C4	UINT32	0		Reserved
Reserved_456	456	0x1C8	UINT32	0		Reserved
Reserved_460	460	0x1CC	UINT32	0		Reserved
Reserved_464	464	0x1D0	UINT32	0		Reserved
Reserved_468	468	0x1D4	UINT32	0		Reserved
Reserved_472	472	0x1D8	UINT32	0		Reserved
Reserved_476	476	0x1DC	UINT32	0		Reserved
Reserved_480	480	0x1E0	UINT32	0		Reserved
Reserved_484	484	0x1E4	UINT32	0		Reserved
Reserved_488	488	0x1E8	UINT32	0		Reserved
Reserved_492	492	0x1EC	UINT32	0		Reserved
Reserved_496	496	0x1F0	UINT32	0		Reserved

Configuration variable	Address		Data type	Default	Range	Description
	Dec.	Hex.				
Reserved_500	500	0x1F4	UINT32	0		Reserved
Reserved_504	504	0x1F8	UINT32	0		Reserved
Reserved_508	508	0x1FC	UINT32	0		Reserved

3.6 Internal Status Variables

Index		Variable	Value	
Dec.	Hex.			
0	0x00	Time since start of program	[ms]	
1	0x01	Current position (measured value)	[user unit]	
2	0x02	Current output frequency	[user unit]	
3	0x03	Number of status variables		
4	0x04	0		
5	0x05	0		
6	0x06	0		
7	0x07	0		
8	0x08	Software Version	ASCII	
9	0x09	HardwareVersion		
10	0x0A	HardwareIdent	2:	Module 750-670
			4:	Module 750-671
			8:	Module 750-672
			16:	Module 750-673
11	0x0B	CompilationMonth	mmmm ASCII	
12	0x0C	CompilationDayYear	ddyy ASCII	
13	0x0D	Compilationtime	hhmm ASCII	
14	0x0E	CPLD	1:	no CPLD
			2:	CPLD not recognized
			3:	CPLD Type 3
			4:	CPLD Type 4
15	0x0F	Expecting configuration version		
16	0x10	Version of current configuration		
17	0x11	0		
18	0x12	0		
19	0x13	0		
20	0x14	0		
21	0x15	0		
22	0x16	0		
23	0x17	PARTMODEL status		
24	0x18	Specified position for position encoder		
25	0x19	Specified velocity for position encoder		
26	0x1A	Current position (internal measured value)		

Index		Variable	Value
Dec.	Hex.		
27	0x1B	Current position (internal offset)	
28	0x1C	Current position (internal OffsetSampled)	
29	0x1D	Current position (internal LastPosition)	
30	0x1E	Last target position	
31	0x1F	Acceleration factor	
32	0x20	Position error	
33	0x21	Current output frequency (internal)	
34	0x22	Current position (internal CPLD emulation)	
35	0x23	0	
36	0x24	0	
37	0x25	0	
38	0x26	0	
39	0x27	0	
40	0x28	0	
41	0x29	Move program command counter	
42	0x2A	EEPROM status	
43	0x2B	Specified velocity	
44	0x2C	Maximum velocity	
45	0x2D	Final velocity	
46	0x2E	0	
47	0x2F	Stop deceleration	
48	0x30	Priority of current command (internal)	
49	0x31	Current acceleration	
50	0x32	Current deceleration	
51	0x33	Acceleration modification (path, time)	
52	0x34	Deceleration modification (path, time)	
53	0x35	Ramp mode	
54	0x36	Maximum velocity (internal)	
55	0x37	Prescaler velocity	
56	0x38	Present current factor	
57	0x39	Original status for camshaft 1 ... 9	
58	0x3A	Reference run status	
59	0x3B	Reference run limit switch contact status	
60	0x3C	Referencing speed (internal)	
61	0x3D	Reference run: Creep speed	

Index		Variable	Value
Dec.	Hex.		
62	0x3E	0	
63	0x3F	0	
64	0x40	Fast timer min. computing time	[us]
65	0x41	Fast timer max. computing time	[us]
66	0x42	Fast timer avg. computing time	[us]
67	0x43	Main loop min. computing time	[us]
68	0x44	Main loop max. computing time	[us]
69	0x45	Main loop avg. computing time	[us]
70	0x46	Main timer min. computing time	[us]
71	0x47	Main timer max. computing time	[us]
72	0x48	Main timer avg. computing time	[us]
73	0x49	Background min. computing time	[us]
74	0x4A	Background max. computing time	[us]
75	0x4B	Background avg. computing time	[us]
76	0x4C	Aux. timer min. computing time	[us]
77	0x4D	Aux. timer max. computing time	[us]
78	0x4E	Aux. timer avg. computing time	[us]
79	0x4F	WA200 Interrupt min. computing time	[us]
80	0x50	WA200 Interrupt max. computing time	[us]
81	0x51	WA200 Interrupt avg. computing time	[us]
82	0x52	WA200 Interrupt min. period	[us]
83	0x53	WA200 Interrupt max. period	[us]
84	0x54	WA200 Interrupt avg. period	[us]
85	0x55	Internal bus interrupt min. period	[us]
86	0x56	Internal bus interrupt max. period	[us]
87	0x57	Internal bus interrupt avg. period	[us]
88	0x58	0	
89	0x59	0	
90	0x5A	0	
91	0x5B	Status variable Filter1	
92	0x5C	Status variable Filter2	
93	0x5D	Status variable Filter3	
94	0x5E	Status variable Filter4	
95	0x5F	Status variable Filter5	
96	0x60	Status variable Filter6	

Index		Variable	Value
Dec.	Hex.		
97	0x61	Status variable Filter7	
98	0x62	Status variable Filter8	
99	0x63	0	

WAGO Kontakttechnik GmbH & Co. KG
Postfach 2880 • D-32385 Minden
Hansastraße 27 • D-32423 Minden
Phone: 05 71/8 87 – 0
Fax: 05 71/8 87 – 1 69
E-Mail: info@wago.com

Internet: http://www.wago.com