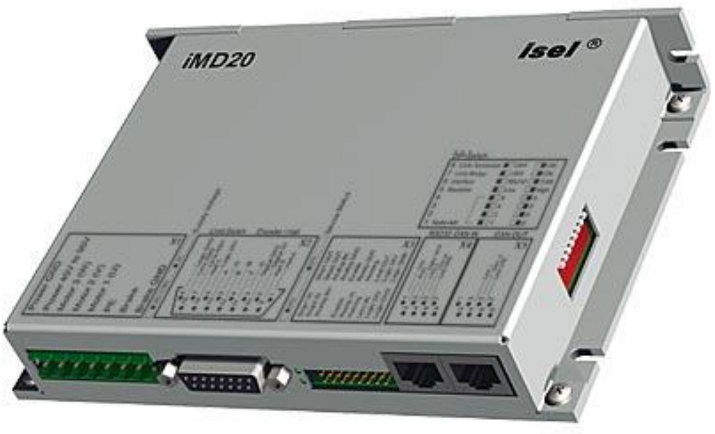
From Components to Systems

AC servo positioning module with CanOpen interface

IMD20



IMD40

www.isel.com

Remarks concerning this manual:

Despite the utmost care, print and other errors cannot be excluded.
If you have any suggestions for improvements and hints as regards errors, don't hesitate to contact us.

Prior to the initial use of the servo drive controller you must read the chapters **"Safety notes for electrical drives and controllers"** starting on page 8 and **"Notes on safe and EMC-compliant installation"** starting on page 15.
Do not try to install or commission the servo drive controller before carefully reading all safety notes for electrical drives and controllers contained in this document. These safety instructions and all other user notes must be read prior to any work with the servo drive controller.

Manufacturer: ***isel*** Germany AG
Bürgermeister-Ebert-Straße 40
D-36124 Eichenzell
Germany

Tel.: +49 - 6659 - 981-0
Fax: +49 – 6659 - 981-776
email: automation@isel.com
http://www.isel.com

Last Updated: 12/04/2018

Table of contents

Overview

Introduction

Our tried-and-tested IMD20 and IMD40 drive modules are used more than 25000 times worldwide. The present manual contains all descriptions and documentations required for the wiring, commissioning and control of this both drive modules.

It is intended for qualified personnel with basic knowledge of the control and automation technology as well as of the CAN field bus.

Prior to the initial use of the servo drive controller you must read the chapters “Safety notes for electrical drives and controllers” starting on page 9 and “Notes on safe and EMC-compliant installation” starting on page 17.

Do not try to install or commission the servo drive controller before carefully reading all safety notes for electrical drives and controllers contained in this document. These safety instructions and all other user notes must be read prior to any work with the servo drive controller.

The basics of the CanOpen protocol are described in the chapter “CanOpen protocol” starting on page 89 to the extent that they are important to the drive process. This chapter also presents an overview of all CanOpen objects of this module as well as a description of the drive functions. The chapter “ACSetup programme” starting on page 46 gives a description of the ACSetup programme and serves as a guide to the commissioning of the drive.

When you try to pin the program ACSetup to the task bar or to the start menu, you will observe that both selection possibilities upon clicking with the right mouse key on the ACSetup-program symbol are not shown. This is due to the word “Setup” in the program name. The operation system Windows impedes the attachment of each program whose name contains the name “Setup”. Therefore, when you want to pin the program ACSetup, you must rename it.

The manual is also provided as online help with the setting program ACSetup. If, upon starting ACSetup, the error message

“Error upon opening the help file! “

is shown, you must take one of the following measures.

- You must realise the program ACSetup temporarily as administrator (“Right mouse key on the ACSetup-program-symbol\Run as administrator“). When you want to run ACSetup always as administrator, you must activate this option. With the right mouse key on ACSetup-program-symbol, by selecting “Properties” you can show the dialogue “ACSetup Properties“. In the tap “Compatibility” you can activate the option “Run the program as an administrator” permanently.
- You must install the program ACSetup on a local hard disc instead of on a network hard disk.
- Both online help files ACSetup_eng.chm and ACSetup_ger.chm are in the folder where ACSetup is also situated. With the right mouse key on the help file, by selecting “Properties” you can show the dialogue “Properties of …“ In the Tab “Sercurity” you must release by means of the button “Edit…” the right “Read & Execute”. Follow the same procedure with the other help file.

If the contents page remains empty when accessing help topics via the table of contents, there is a access problem to the help file. In this case, select the help file with the right mouse button. Selecting "Properties" displays the dialog box "… Properties". The button "Unblock" on the tab "General" should be clicked.

If you want to use the online help files ACSetup_eng.chm and ACSetup_ger.chm separately without the ACSetup program, you must rename the online help files to ACSetup.chm. Otherwise, the links in the online files will no longer work correctly.

Features of the drive module

- For operation of 3-phase permanent magnet synchronous motors (PMSM)
- Supply voltage IMD20 40-95 V DC, IMD40 230 V AC (+/- 10%)
- Current consumption of control board (24 V)
- IMD20: 350 mA without brake and 750mA with brake
- IMD40: 400 mA without brake and 800mA with brake
- Motor current IMD20 bis 25 A (continuous current 12 A), IMD40 bis 8 A
- CAN bus interface according to CanOpen DS301 V4.0 and DS402 V1.0
- RS232 interface
- Analogue input (±10 V) mit 11-bit resolution
- Inputs for limit and reference switches
- Digital current, velocity and position control with high cycle times
- Brake control
- Gantry operation or synchronous control of 2 modules
- Online distance control with analogue input (±10 V)
- Fast stop on inputs
- Electronical handwheel
- Movement of the homing point
- Shifting the reference point for endless rotation
- Monitoring of the motor current and encoder signals
- Monitoring of the software through the internal watchdog timer
- Galvanic isolation of processor, power section and I/Os
- Easy firmware update via RS232

Scope of application

The drive module is suited ideally for the control of small- and medium-power permanet magnet motors. What has to be pointed out is the high torque even at small velocities and the outraging synchronous characteristics even at low speeds. Thanks to five modes of operation and a high diversity of adjustable parameters, a broad spectrum of the most diverse applications in the automation technology and the mechanical engineering can be covered.

The CanOpen interface is an open interface which allows to build up a flexible, extensible plant structure or to integrate the modules into existing plants.

Operational environment

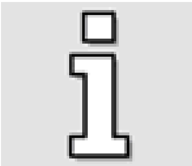
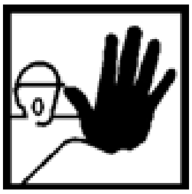
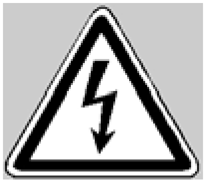
To get the full functionality of the drive module the following environmental conditions should be ensured

- Ambient temperature during operation: from +5^{O}C to 40^{O}C
- Temperature during transport and storage: from -25^{O}C to 55^{O}C and at 70^{O}C for maximum 24 hours
- Maximum height of installation: about 1000m above sea level
- Maximum humidity 50% at 40^{O}C, 90% at 20^{O}C

- Vertical mounting position with appropriate distance each other to allow adequate air circulation
- Shielded motor cable with maximum lenght of 25m and minimal thickness of 1,5mm²
- IMD20-protection in power supply and IMD40-protection 1-phase 16A time-lage fuse

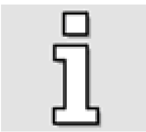
Safety notes for electrical drives and controllers

Used symbols

	**Information** Important information and notes
	**Caution !** Nonobservance may result in severe property damages.
	**DANGER !** Nonobservance may result in **property damages** and in **personal injuries**.
	**Caution! Dangerous voltages.** The safety note indicates a possible perilous voltage.

General notes

In the case of damage resulting from non-compliance of the safety notes in this manual isel Germany AG will assume no liability.

	Prior to the initial use you must read the chapters “Safety notes for electrical drives and controllers” starting on page 9 and “Notes on safe and EMC-compliant installation” starting on page 17.

If the documentation in the language at hand is not understood accurately, please contact and inform your supplier.

Sound and safe operation of the servo drive controller requires proper and professional transportation, storage, assembly and installation as well as proper operation and maintenance. Only trained and qualified personnel may handle electrical devices:

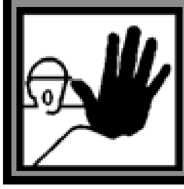
TRAINED AND QUALIFIED PERSONNEL

in the sense of this product manual or the safety notes on the product itself are persons who are sufficiently familiar with the project, the setup, assembly, commissioning and operation of the product as well as all warnings and precautions as per the instructions in this manual and who are sufficiently qualified in their field of expertise:

- Education and instruction concerning the standards and accident prevention regulations for the application, or authorisation to switch devices/systems on and off and to ground them as per the standards of safety engineering and to efficiently label them as per the job demands.
- Education and instruction as per the standards of safety engineering regarding the maintenance and use of adequate safety equipment.
- First aid training.

The following notes must be read prior to the initial operation of the system to prevent personal injuries and/or property damages:

	These safety notes must be complied with at all times.
	Do not try to install or commission the servo drive controller before carefully reading all safety notes for electrical drives and controllers contained in this document. These safety instructions and all other user notes must be read prior to any work with the servo drive controller.
	In case you do not have any user notes for the servo drive controller, please contact your sales representative. Immediately demand these documents to be sent to the person responsible for the safe operation of the servo drive controller.
	If you sell, rent and/or otherwise make this device available to others, these safety notes must also be included.
	The user must not open the servo drive controller for safety and warranty reasons.
	Professional control process design is a prerequisite for sound functioning of the servo drive controller!



DANGER!

Inappropriate handling of the servo drive controller and non-compliance of the warnings as well as inappropriate intervention in the safety features may result in property damage, personal injuries, electric shock or in extreme cases even death.

Danger resulting from misuse

	DANGER! High electrical voltages and high load currents! Danger to life or serious personal injury from electrical shock!
	DANGER! High electrical voltage caused by wrong connections! Danger to life or serious personal injury from electrical shock!
	DANGER! Surfaces of device housing may be hot! Risk of injury! Risk of burning!
	DANGER! **Dangerous movements!** Danger to life, serious personal injury or property damage due to unintentional movements of the motors!

General safety notes

	The servo drive controller corresponds to IP20 class of protection as well as pollution level 1. Make sure that the environment corresponds to this class of protection and pollution level.
	Only use replacements parts and accessories approved by the manufacturer.
	The devices must be connected to the mains supply as per EN regulations, so that they can be cut off the mains supply by means of corresponding separation devices (e.g. main switch, contactor, power switch).
	The servo drive controller may be protected using an AC/DC sensitive 300mA fault current protection switch (RCD = Residual Current protective Device).
	Gold contacts or contacts with a high contact pressure should be used to switch the control contacts.
	Preventive interference rejection measures should be taken for control panels, such as connecting contactors and relays using RC elements or diodes.

	The safety rules and regulations of the country in which the device will be operated must be complied with.
	The environment conditions defined in the product documentation must be kept. Safety-critical applications are not allowed
	For notes on installation corresponding to EMC, please refer to chapter “Notes on safe and EMC-compliant installation” (page17). The compliance with the limits required by national regulations is the responsibility of the manufacturer of the machine or system.
	The technical data and the connection and installation conditions for the servo drive controller are to be found in this product manual and must be met.

DANGER!

The general setup and safety regulations for work on power installations (e.g. DIN, VDE, EN, IEC or other national and international regulations) must be complied with.

Non-compliance may result in death, personal injury or serious property damages.

	Without claiming completeness, the following regulations and others or standards apply: VDE 0100 Regulations for the installation of high voltage (up to 1000 V) devices EN 60204-1 Electrical equipment of machines EN 50178 Electronic equipment for use in power installations EN ISO 12100 Safety of machinery - Basic terminology, general principles for design EN 1050 Safety of machinery - Principles for risk assessment EN 1037 Safety of machinery - Prevention of unexpected start-up EN 954-1 Safety-related parts of control systems

Safety notes for assembly and maintenance

The appropriate DIN, VDE, EN and IEC regulations as well as all national and local safety regulations and rules for the prevention of accidents apply for the assembly and maintenance of the system. The plant engineer or the operator is responsible for compliance with these regulations:

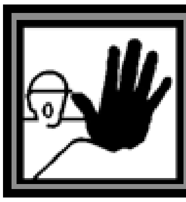
	The servo drive controller must only be operated, maintained and/or repaired by personnel trained and qualified for working on or with electrical devices.

Prevention of accidents, injuries and/or damages:

	Additionally secure vertical axes against falling down or lowering after the motor has been switched off, e.g. by means of: • Mechanical locking of the vertical axle, • External braking, catching or clamping devices or • Sufficient balancing of the axle.
	The motor holding brake supplied by default or an external motor holding brake driven by the drive controller alone is not suitable for personal protection!
	Render the electrical equipment voltage-free using the main switch and protect it from being switched on again until the DC bus circuit is discharged, in the case of: • Maintenance and repair work • Cleaning • Long machine shutdowns
	Prior to carrying out maintenance work make sure that the power supply has been turned off, locked and the DC bus circuit is discharged.
	The external or internal brake resistor carries dangerous DC bus voltages during operation of the servo drive controller and up to 5 minutes thereafter. Contact may result in death or serious personal injury. Actual value of the DC bus voltages see CAN object 0x2072.
	Be careful during the assembly. During the assembly and also later during operation of the drive, make sure to prevent drill chips, metal dust or assembly parts (screws, nuts, cable sections) from falling into the device.
	Also make sure that the external power supply of the controller (24V) is switched off.
	The DC bus circuit or the mains supply must always be switched off prior to switching off the 24V controller supply.
	Carry out work in the machine area only, if AC and/or DC supplies are switched off. Switched off output stages or controller enablings are no suitable means of locking. In the case of a malfunction the drive may accidentally be put into action.
	Initial operation must be carried out with idle motors, to prevent mechanical damages e.g. due to the wrong direction of rotation.
	Electronic devices are never fail-safe. It is the user’s responsibility, in the case an electrical device fails, to make sure the system is transferred into a secure state.
	The servo drive controller and in particular the brake resistor, externally or internally, can assume high temperatures, which may cause serious burns.

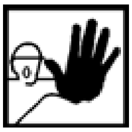
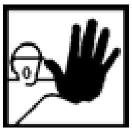
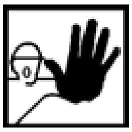
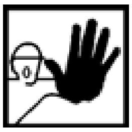
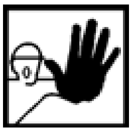
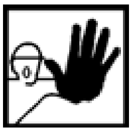
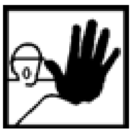
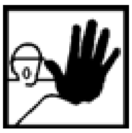
Protection against contact with electrical parts

This section only concerns devices and drive components carrying voltages exceeding 50 V. Contact with parts carrying voltages of more than 50 V can be dangerous for people and may cause electrical shock. During operation of electrical devices some parts of these devices will inevitably carry dangerous voltages.



DANGER!
High electrical voltage!
Danger to life, danger due to electrical shock or serious personal injury!

The appropriate DIN, VDE, EN and IEC regulations as well as all national and local safety regulations and rules for the prevention of accidents apply for the assembly and maintenance of the system. The plant engineer or the operator is responsible for compliance with these regulations:

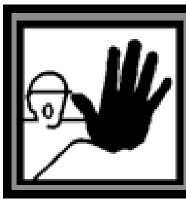
	Before switching on the device, install the appropriate covers and protections against accidental contact. Rack-mounted devices must be protected against accidental contact by means of a housing, e.g. a switch cabinet. The regulations VGB4 must be complied with!
	Always connect the ground conductor of the electrical equipment and devices securely to the mains supply. Due to the integrated line filter the leakage current exceeds 3.5 mA!
	Comply with the minimum copper cross-section for the ground conductor over its entire length as per EN60617!
	Prior to the initial operation, even for short measuring or testing purposes, always connect the ground conductor of all electrical devices as per the terminal diagram or connect it to the ground wire. Otherwise the housing may carry high voltages which can cause electrical shock.
	Do not touch electrical connections of the components when switched on.
	Prior to accessing electrical parts carrying voltages exceeding 50 Volts, disconnect the device from the mains or power supply. Protect it from being switched on again.
	For the installation the amount of DC bus voltage must be considered, particularly regarding insulation and protective measures. Ensure proper grounding, wire dimensioning and corresponding short-circuit protection.
	The servo drive controllers can carry voltage until up to 5 minutes after being switched off (residual capacitor charge). Actual value of DC bus voltage see object 0x2072

Protection against electrical shock by means of protective extra-low voltage (PELV)

All connections and terminals with voltages between 5 and 50 Volts at the servo drive controller are protective extra-low voltage, which are designed safe from contact in correspondence with the following standards:

International: IEC 60364-4-41

European countries within the EU: EN 50178/1998, section 5.2.8.1.



DANGER!

High electrical voltages due to wrong connections!

Danger to life, risk of injury due to electrical shock!

Only devices and electrical components and wires with a protective extra low voltage (PELV) may be connected to connectors and terminals with voltages between 0 to 50 Volts.

Only connect voltages and circuits with protection against dangerous voltages. Such protection may be achieved by means of isolation transformers, safe optocouplers or battery operation.

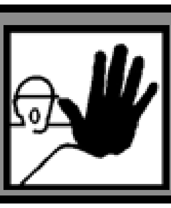
Protection against dangerous movements

Dangerous movements can be caused by faulty control of connected motors, for different reasons:

- Improper or faulty wiring or cabling
- Error in handling of components
- Error in sensor or transducer
- Defective or non-EMC-compliant components
- Error in software in superordinated control system

These errors can occur directly after switching on the device or after an indeterminate time of operation.

The monitors in the drive components for the most part rule out malfunctions in the connected drives. In view of personal protection, particularly the danger of personal injury and/or property damage, this may not be relied on exclusively. Until the built-in monitors come into effect, faulty drive movements must be taken into account; their magnitude depends on the type of control and on the operating state.



DANGER!

Dangerous movements!

Danger to life, risk of injury, serious personal injuries or property damage!

For the reasons mentioned above, personal protection must be ensured by means of monitoring or superordinated measures on the device. These are installed in accordance with the specific data of the system and a danger and error analysis by the manufacturer. The safety regulations applying to the system are also taken into consideration. Random movements or other malfunctions may be caused by switching the safety installations off, by bypassing them or by not activating them.

Protection against contact with hot parts

DANGER!
Housing surfaces may be hot!
Risk of injury! Risk of burning!

	Do not touch housing surfaces in the vicinity of heat sources! Danger of burning!
	Before accessing devices let them cool down for 10 minutes after switching them off.
	Touching hot parts of the equipment such as the housing, which contain heat sinks and resistors, may cause burns!

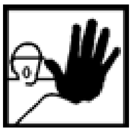
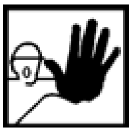
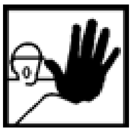
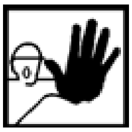
Protection during handling and assembly

Handling and assembly of certain parts and components in an unsuitable manner may under adverse conditions cause injuries.

DANGER!
Risk of injury due to improper handling!
Personal injury due to pinching, shearing, cutting, crushing!

The following general safety notes apply:

	Comply with the general setup and safety regulations on handling and assembly.
	Use suitable assembly and transportation devices.
	Prevent incarcerations and contusions by means of suitable protective measures.
	Use suitable tools only. If specified, use special tools.

	Use lifting devices and tools appropriately.
	If necessary, use suitable protective equipment (e.g. goggles, protective footwear, protective gloves).
	Do not stand underneath hanging loads.
	Remove leaking liquids on the floor immediately to prevent slipping.

Notes on safe and EMC-compliant installation

Definition and terms

Electromagnetic compatibility (EMC) or electromagnetic interference (EMI) includes the following requirements:

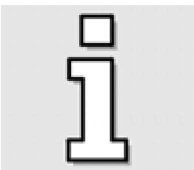
- Sufficient **immunity** of an electrical installation or an electrical device against outside electrical, magnetic or electromagnetic interferences via cables or the ambient.
- Sufficiently small **unwanted emission** of electrical, magnetic or electromagnetic interference from an electrical installation or an electrical device to other devices in the vicinity via cables or the ambient.

General information on EMC

The interference emission and interference immunity of a device always depend on the entire drive concept consisting of the following components:

- Voltage supply
- Servo positioning controller
- Motor
- Electromechanics
- Executiion and type of wiring
- Superimposed control

In order to increase interference immunity and to decrease interference emissions the servo positioning controller already comprises mains filters, so that it can be operated without additional shielding and filtering devices.

	In most cases no external filtering is required (see below).

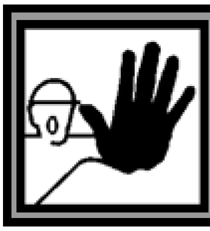
Warning!

This product can cause high-frequency interference in residential areas, which could require measures for radio interference suppression.

EMC-compliant cabling

The following must be considered for an EMC-compliant setup of the drive system:

- In order to keep the leakage currents and the losses in the motor connection cable as small as possible, the servo positioning controller should be located as close to the motor as possible.
- Motor cable and angle encoder cable must be shielded.
- The shield of the motor cable is connected to the housing of the servo positioning controller. The cable shield also has to be connected to the associated servo positioning controller so that the leakage currents can flow back into the controller causing the leakage.
- The mains-end PE connection and the inner PE conductor of the motor is connected to the PE connection point of the supply connection
- The signal lines must be as far away from the power cables as possible. They should not be placed parallel. If intersections cannot be avoided, they should be perpendicular (i.e. at a 90° angle), if possible.
- Unshielded signal and control lines should not be used. If their use is inevitable they should at least be twisted.
- Even shielded cables will inevitably have short unshielded ends (unless shielded connector housings are used). In general, the following applies:
    - Connect the inner shields to the corresponding pins of the connectors; Maximum length 40mm.
    - Length of the unshielded cores 35 mm maximum.
    - Connect the total shield on the controller side plane to the PE terminal; Maximum length 40 mm.
    - Connect the total shield on the motor side plane to the connector housing or motor housing; Maximum length 40 mm.



DANGER!

For safety reasons, all PE ground conductors must be connected prior to initial operation.

The EN 50178 regulations for protective earthing must be complied with during installation!

Description of the drive module

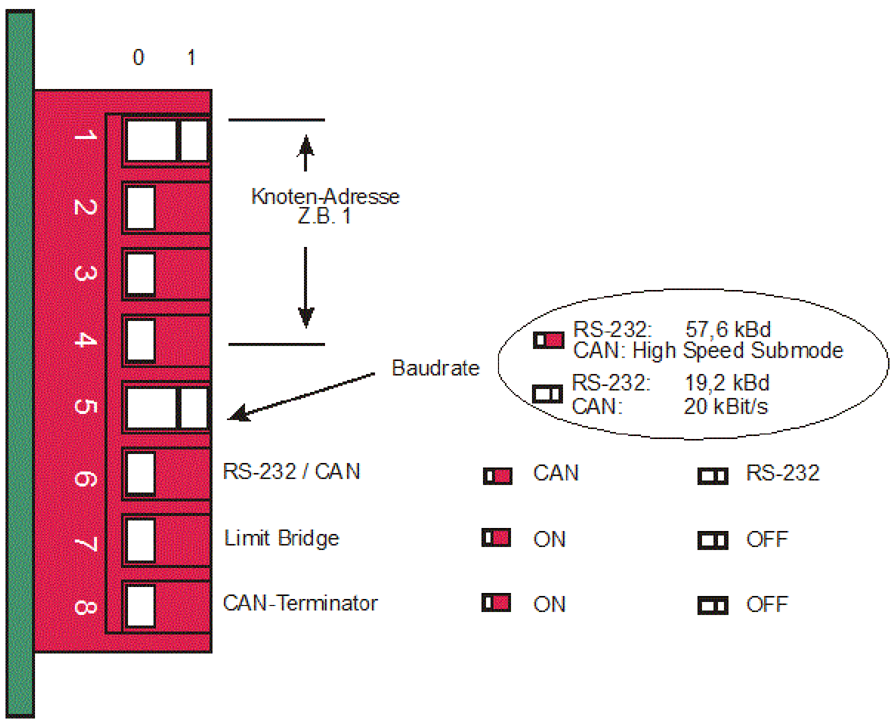
Settings and status display

On the front (IMD40) or on the side (IMD20) of the module, a DIP switch is provided for settings concerning the node address, the baud rate, limit bridge and CAN terminator. Furthermore, LEDs show the current mode of operation.

DIP switch

The DIP switch is only queried, when the module is switched on or after it was reset. During the operation, a commutation of the switch does not have any effects whatsoever.

- The switches S1 to S4 are used to set up the CanOpen node address. Possible node addresses are: **1** to **15**. If the Node number 0 is set, the drive module is started in the operation mode “velocity controller analog input”. Comunication is only possible via serial interface.
- The switch S5 defines the baud rate for the CAN connection **and** the RS232 interface. In High Speed Submode the CAN baud rate of the module is determined by the object Can Baud Rate (2001) (see object directory, manufacturer specific objects, object Can Baud Rate). Under „Setting->Object directory->Manufacturer specific objects->2001 Can Baud Rate->03 New High Speed Submode“ of ACSetup you can change the CAN baud rate of the module (see chapter “Can transmission speed” on page 133). After switch off and switch on of the modul the new baud rate is used. When you change the baud rate of the module do not forget to change also the baud rate in ACSetup and CANSet. For the interpolation with gantry axis at least a baud rate of 250 kBit/s is needed, for normal interpolation you need at least 125 kBit/s. Default is 1 Mbit/s..
- In the state ON the switch S7 connects the both connectors Limit switch chain In and Limit switch chain Out.
- In the state ON the switch S8 terminates the CAN bus with a resistor of 120 Ohm.



Modes of operation

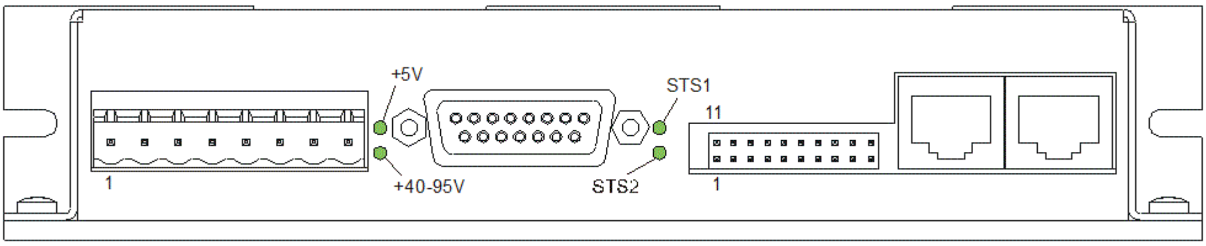
Here, two groups of modes of operation are available differing from each other mainly in the type of the controller release. In case of the CanOpen modes of operation, the internal state (state machine) is controlled via the Can bus or the serial interface, while there are for the mode “velocity controller analogue input” only two states of operation (Enabled, Disabled) which are linked directly with the enabling signal (Input 4).

The active mode of operation of the drive module is defined through the settings of the DIP switch on the front side and through the “Modes of operation” parameter ("Modes of Operation", 6060_h - See “Mode of operation” on page 112).

When the module is switched on, the DIP switch is queried at first. If the node address is set to 0 at the DIP switch the operation mode “velocity controller analogue input” is set. If the node address is unequal 0 the parameter „modes of operation“ is evaluated.

Node address		Mode of operation
Node address unequal 0. Mode of operation is set by the parameter „Modes of operation“	**Modes of operation**	
	1	Position control (Profile position mode)
	3	Velocity control (Profile velocity mode)
	6	Reference run (homing mode)
	7	Interpolation (interpolated position mode)
	-3	Leaving limit switch (Moving the axis away from a active hardware limit switch)
Node address equal 0	-2	Velocity control (via ±10 volt analogue input)

LEDs IMD20



Via the LEDs on the front, it is possible to indicate the existence of the two main supply voltages. Furthermore, the current state of the module can be viewed here (STS1 and STS2)

(See “Status diagram” on page 98 and “Device control” on page 109).

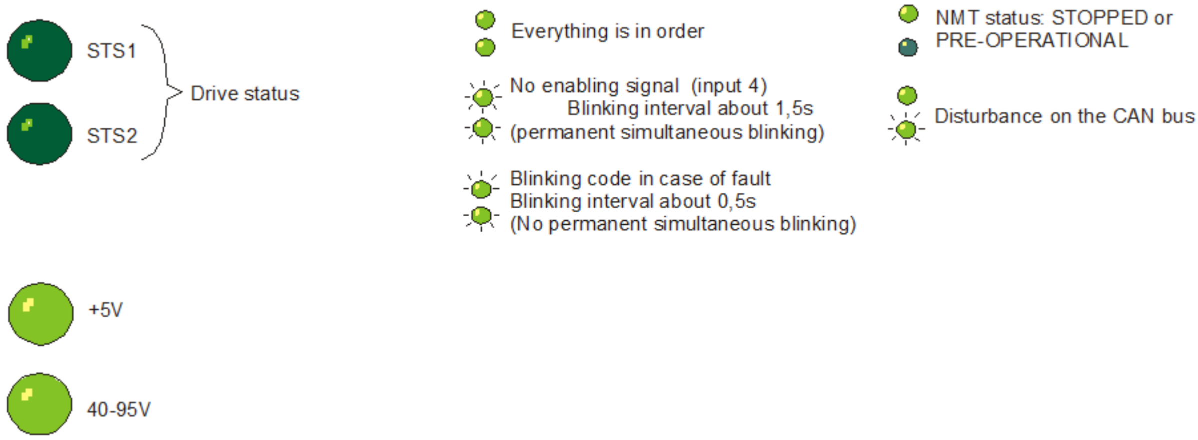
In case of a fault (Fault state), the two-digit error code of the current fault is given via a blinking code.

1st digit: number of blinking impulses of both status LEDs (STS1 and STS2).

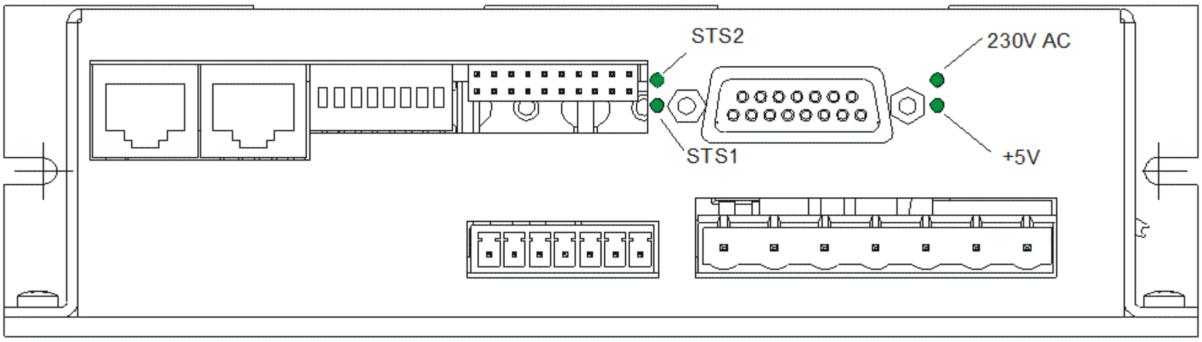
2nd digit: number of blink impulses of LED STS2.

(See "EMCY" on page 96 for the description of the error code.)

If the enbale signal is not active both LEDs (STS1 and STS2) blink at the same time and periodically, with about 1,5s interval.



LEDs IMD40



Via the LEDs on the front, it is possible to indicate the existence of the two main supply voltages. Furthermore, the current state of the module can be viewed here (STS1 and STS2)

(See “Status diagram” on page 98 and “Device control” on page 109).

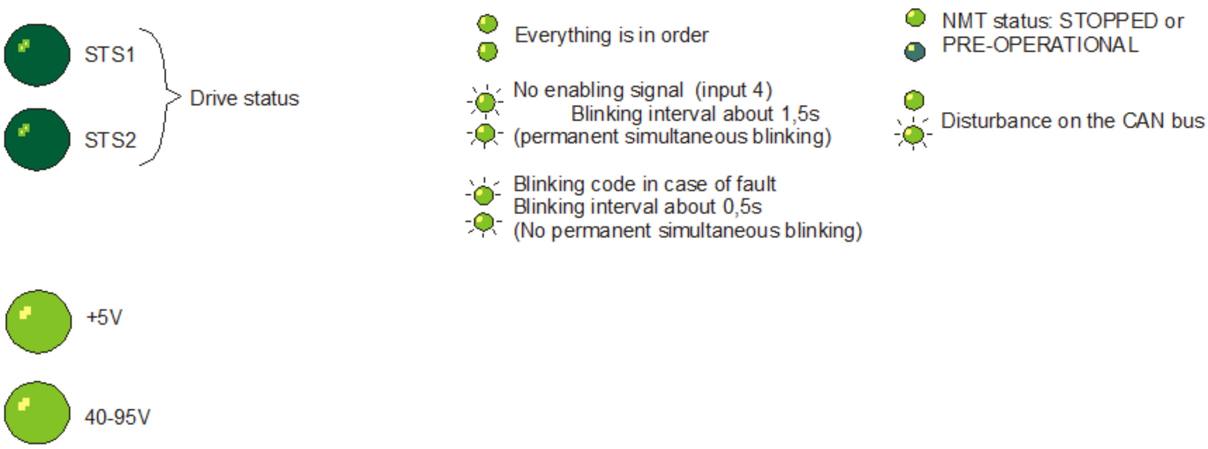
In case of a fault (Fault state), the two-digit error code of the current fault is given via a blinking code.

1st digit: number of blinking impulses of both status LEDs (STS1 und STS2).

2nd digit: number of blink impulses of LED STS2.

(See "EMCY" on page 96 for the description of the error code.)

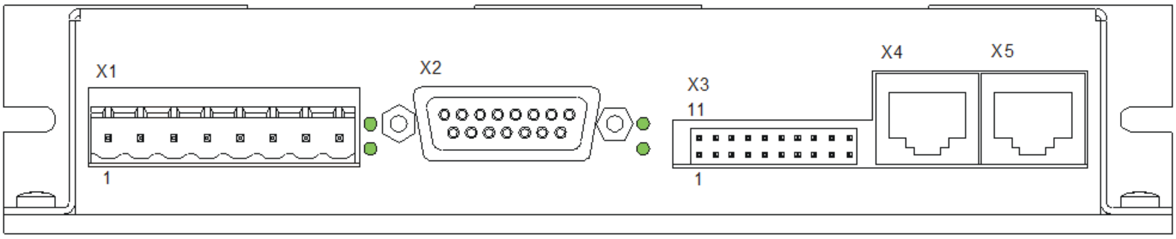
In case of a missing enabling signal, all two LEDs (STS1, STS2) blink about once every 1.5 seconds.



Hardware description IMD20

Connectors

All connectors are provided on the front of the drive module.



Frame connector X1 – Connection of the operating voltage 40V-95V, motor and brake

Pin	Signal
1	Power GND (power section)
2	→ Power +40–+95 V (power section)
3	← Motor 3 (W)
4	← Motor 2 (V)
5	← Motor 1 (U)
6	PE
7	← Brake
8	Brake GND (GND_24V)

Frame connector X2 – 15-pin Sub-D (SUBD15) – Connection of the encoder, hall sensors and limit switch

Pin	Signal
1	→ Hall A
2	← Encoder voltage 5 V (Digital 5V)
3	→ Encoder /Z
4	→ Encoder /B
5	→ Encoder /A
6	Logic +24V
7	→ Limit switch 1
8	GND_24V (Logic GND)
9	→ Hall B
10	Encoder GND (Digital GND)
11	→ Encoder Z
12	→ Encoder B
13	→ Encoder A
14	→ Hall C
15	→ Limit switch 2

Frame connector X3 – 20 pin – System connection

Pin	Signal
1	Stop1 IN
2	Stop2 IN
3	→ Analog input+ (Analog IN+)
4	→ Temperatur sensor motor (Temperature)
5	→ Input Enable
6	→ Input Ready In (Ready IN)
7	← Output Homing
8	→ Input chain limit switch In (LIMIT IN)
9	+24V (Logic 24V)
10	GND_24V (Logic GND)
11	Stop1 OUT
12	Stop2 OUT
13	→ Analog input - (Analog IN -)
14	→ Analog GND
15	→ Input Enable
16	← Output Ready Out (Ready OUT)
17	← Output Homing
18	← Ouput chain limit swirch Out (Limit OUT)
19	+24V (Logic 24V)
20	GND_24V (Logic GND)

Frame connector X4 – RJ45 - RS-232 and CAN IN

Pin	1	2	3	4	5	6	7	8
Signal	RS232 TxD	RS232 RxD	RS232 GND (Digital GND)	↔ CAN Low	↔ CAN High	CAN GND	-	-

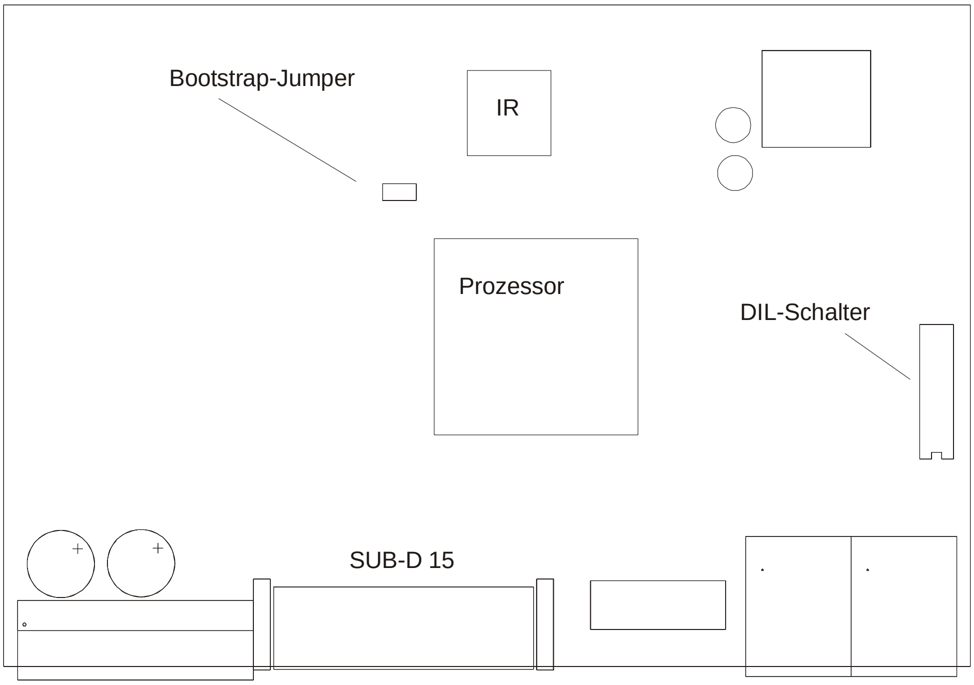
Frame connector X5 – RJ45 – CAN OUT

Pin	1	2	3	4	5	6	7	8
Signal	-	-	-	↔ CAN Low	↔ CAN High	CAN GND	-	-

Jumper settings

For the normal operation, no settings via jumpers are required. Only for the loading process of a new software version by means of the bootstrap loader, the bootstrap jumper has to be connected.

(See also “The Firmware Update / via Bootstrap Loader command” on page 49.)

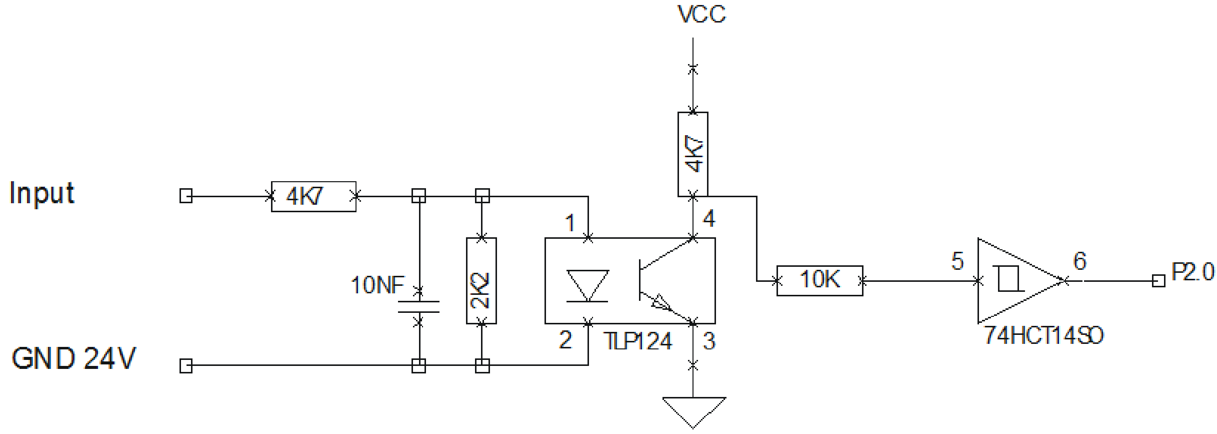


Wiring of the inputs and outputs

Digital Inputs limit switch, Enable

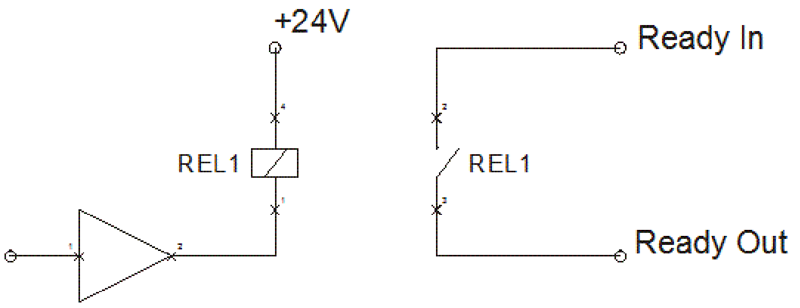
The evaluation of the inputs can be set by means of software.

(See “Inputs dialogue box” on page 75.)



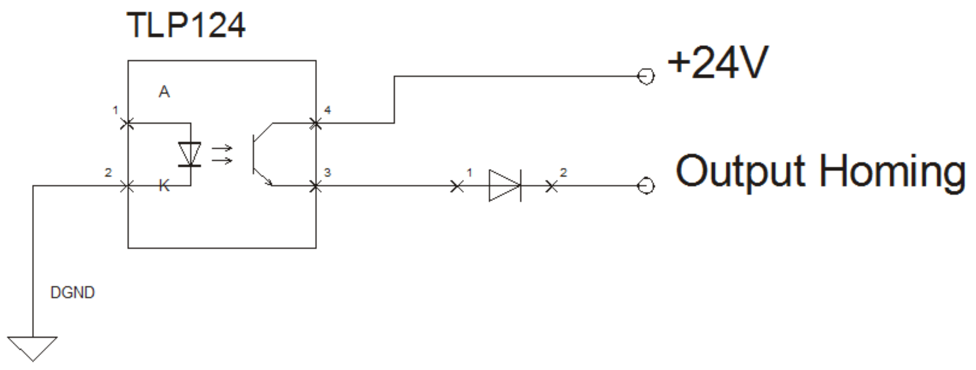
Digital output Ready

The Ready output is designed as a potential-free contact. It indicates the operational readiness of the module and is only set, if the module software works correctly (monitoring through internal watchdog timer). The design as a potential-free contact makes it possible to easily link several Ready outputs. This sum signal can be processed e.g. in the safety circuit.



Digital output Homing

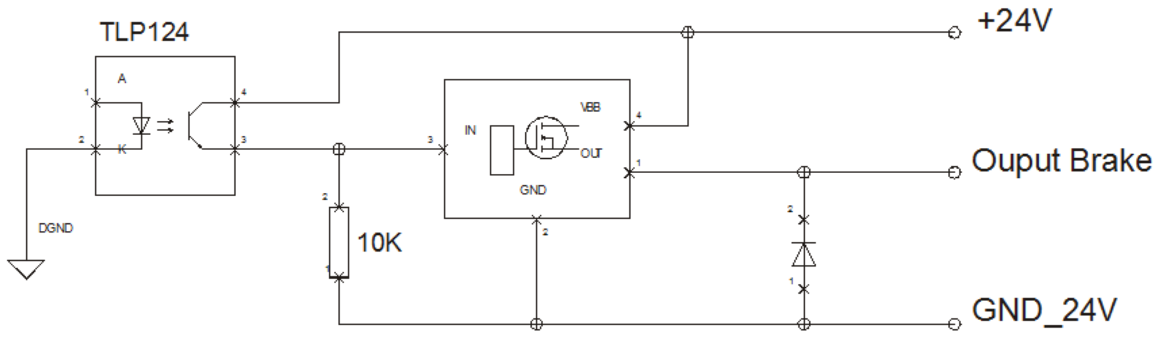
The Homing output can be used to shunt the limit switch in the safety circuit. It is set during the reference run (on limit switch).



Digital output Brake

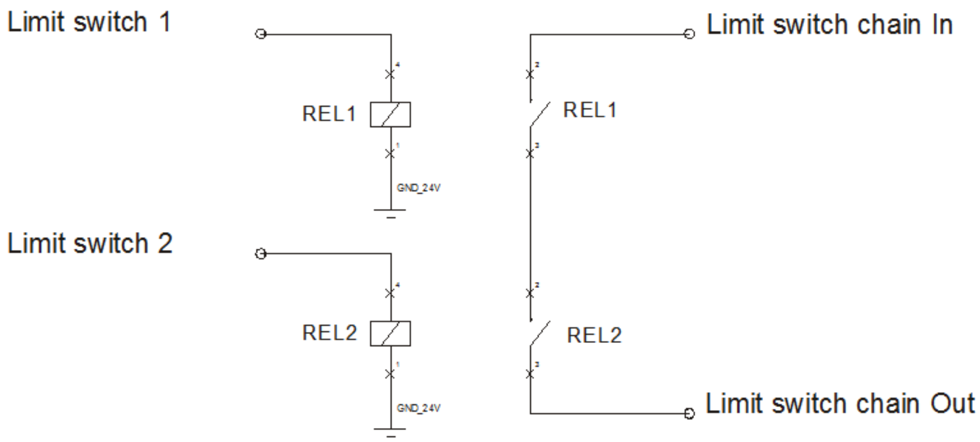
The Brake output is used for the brake control. The way in which the brake is controlled can be defined by means of the software

(See “Brake dialogue box” on page 59).



Digital output Limit switch chain

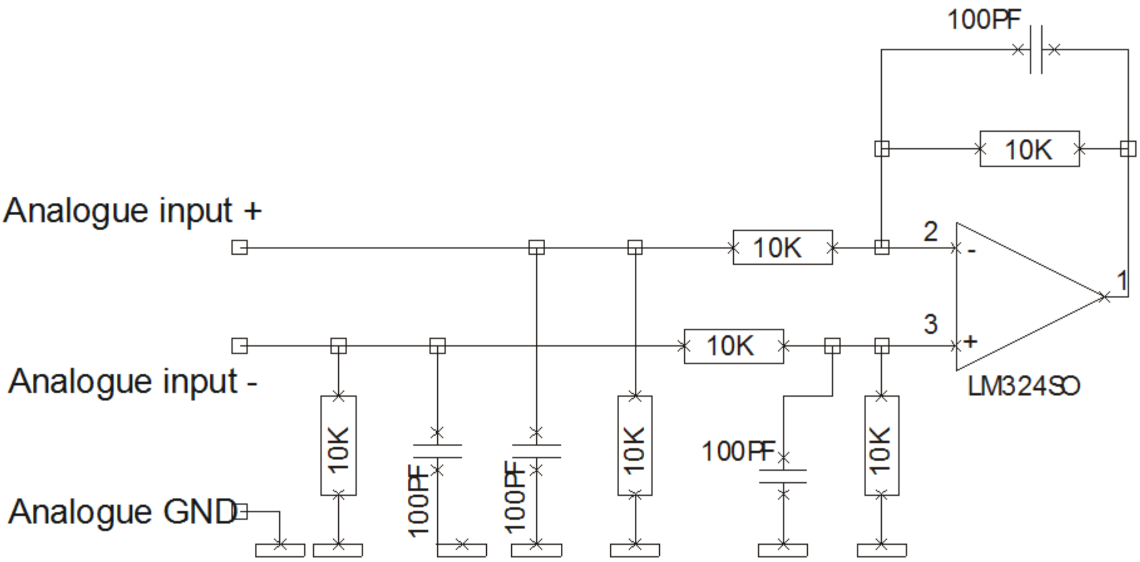
The Limit Switch Chain output is designed as a potential-free contact. It indicates, if both limit switches are not activated (closed potential-free contact). If one or both limit switch(es) is (are) activated, the contact is open. In the safety circuit, this signal can be used to monitor the limit switch.



Analogue input

The analogue input is preset for a voltage level in the area of -10 V … +10 V. The signal can either be connected to the positive input (+) or to the inverting signal (-). The reference potential is always Analogue GND.

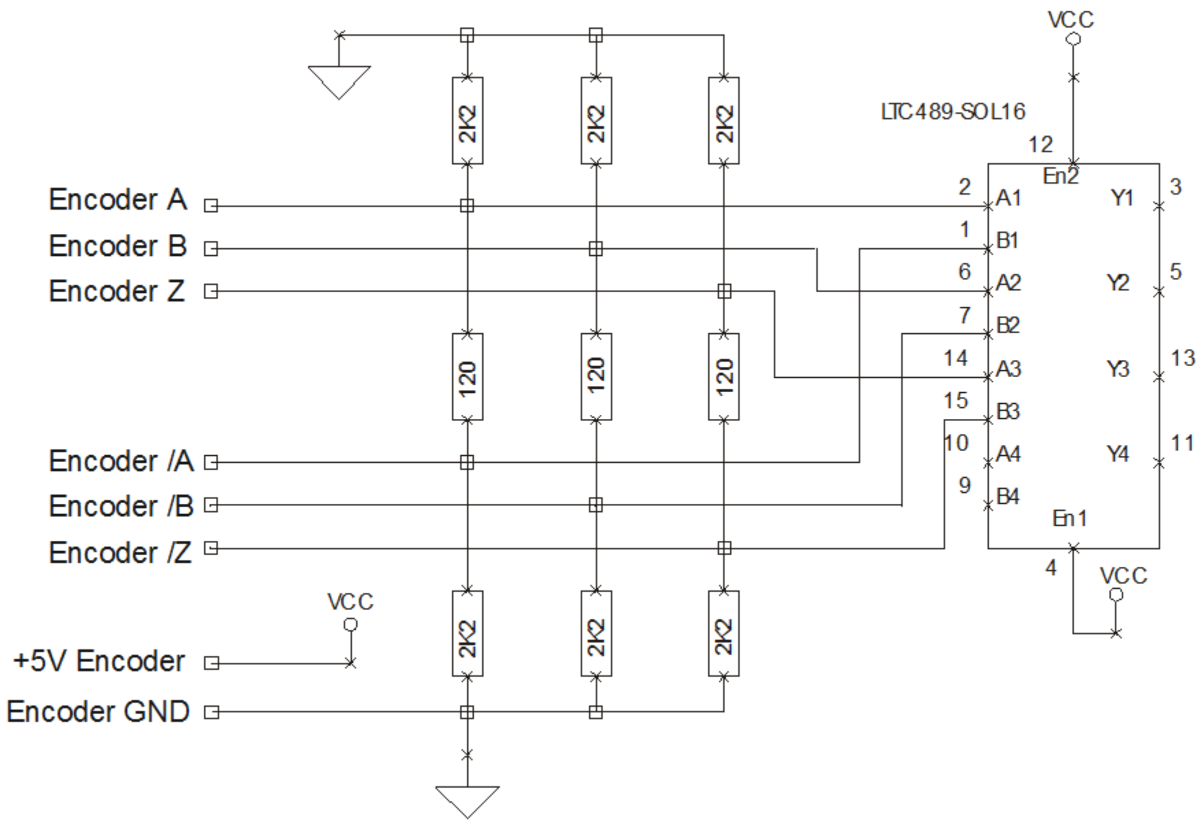
(See “Analogue Input dialogue box” on page 59.)



Encoder connections

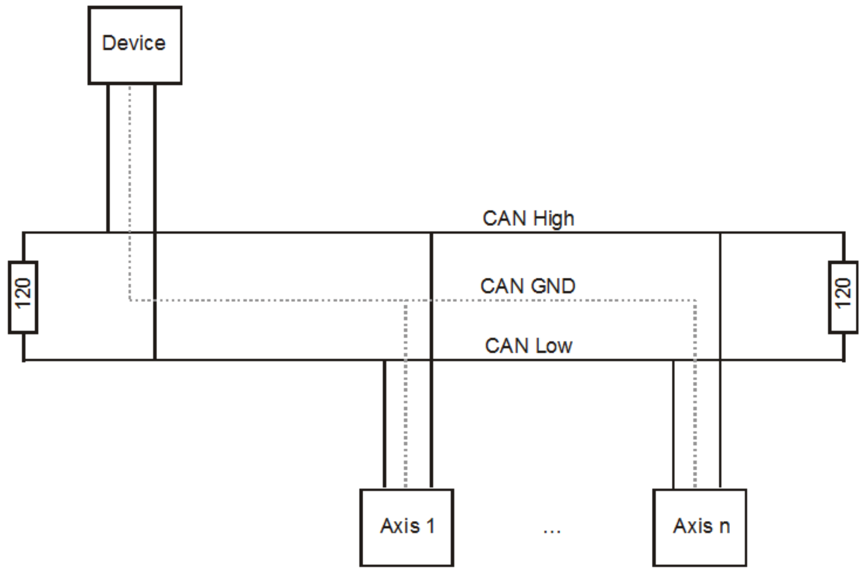
The drive module is preset for the connection of a quadrature incremental encoder with index signal. The transfer of the signals is carried out according to the RS422 specification. It is recommended to use shielded cables twisted in pairs for the encoder wiring.

The voltage supply of the encoder (5 volt) is provided by the drive module. The maximum supply current amounts to 100 mA.



CAN

The structure of the Can network is to be realised so as to guarantee that a 120 ohm terminating resistor is provided on both sides. On the drive module itself exists a terminating resistor. It is activated by the DIP switch CAN terminator. The stub lines leading from the bus to the individual modules should not be longer than 50 cm in case of a baud rate of 1 Mbit/s.

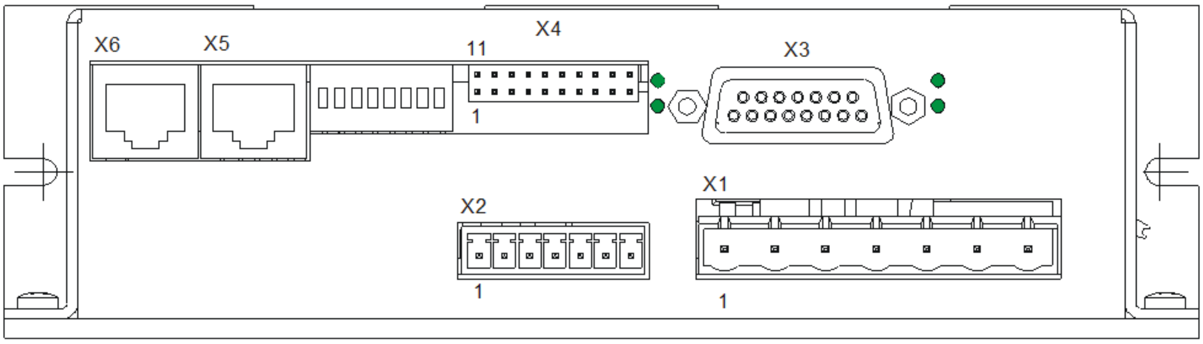


For the CAN wiring, it is recommended to use twisted shielded cables with a characteristic impedance of 108 to 132 ohms. In case of very small network extensions, it may be possible to dispense with the connection of the reference potential (CAN ground) (not recommended).

Hardware description IMD40

Connectors

All connectors are provided on the front of the drive module with the exception of the brake resistor connection on one side of the IMD40.



Frame connector X1 – Connection of the operating voltage AC 220V and motor

Pin	Signal
1	← Motor U
2	← Motor V
3	← Motor W
4	→ PE
5	→ PE
6	→ L (Mains phase)
7	→ N (Mains neutral conductor)

Frame connector X2 –Connection of operating voltage 24V, temperature and brake

Pin	Signal
1	→ +24 V (Logic 24V)
2	→ Temperatur sensor motor (Temperature)
3	← Brake
4	GND_24V (Logic GND)
5	Limit SW 1
6	Limit SW 2
7	GND_24V (logic GND)

Frame connector X3 – 15-pin Sub-D (SUBD15) - Connection encoder, hall sensors, limit switch

Pin	Signal
1	→ Hall A
2	← Encoder-Voltage 5 V (Digital 5V)
3	→ Encoder /Z
4	→ Encoder /B
5	→ Encoder /A
6	Logic +24V
7	→ Limit switch 1 (Limit SW1)
8	GND_24V (Logic GND)
9	→ Hall B
10	Encoder GND (Digital GND)
11	→ Encoder Z
12	→ Encoder B
13	→ Encoder A
14	→ Hall C
15	→ Limit switch 2 (Limit SW 2)

Frame connector X4 – 20-pin – System connection

Pin	Signal
1	GND_24V (Logic GND)
2	+24V (Logic 24V)
3	→ Input limit switch chain In (Limit IN)
4	← Output Homing
5	→ Input Ready In (Ready IN)
6	→ Input Enable
7	→ Analogue GND
8	→ Analogue input - (Analog IN -)
9	Stop2 OUT
10	Stop1 OUT
11	GND_24V (Logic GND)
12	+24V (Logic 24V)
13	← Output limit switch chain Out (Limit OUT)
14	← Output Homing
15	← Output Ready Out (Ready OUT)
16	→ Input Enable
17	Digital GND
18	→ Analogue inputr + (Analog IN +)
19	Stop2 IN
20	Stop1 IN

Frame connector X5 – RJ45 - RS-232 und CAN IN

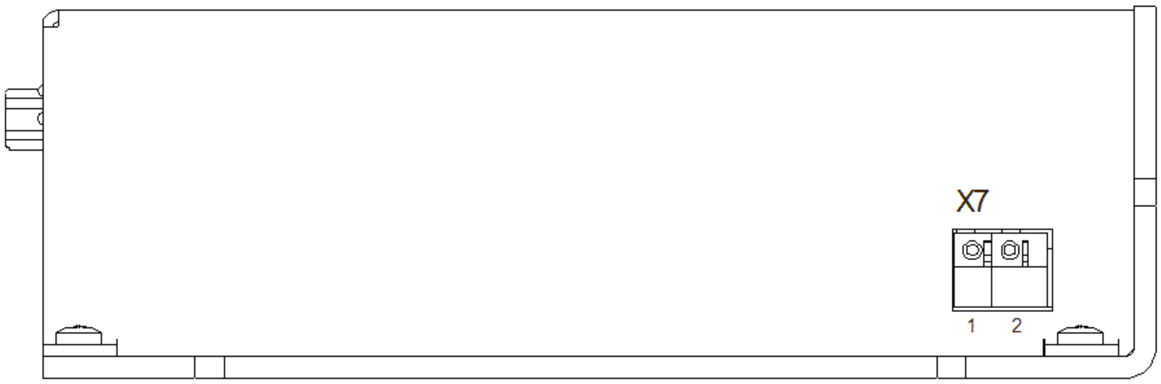
Pin	1	2	3	4	5	6	7	8
Signal	RS232 TxD	RS232 RxD	RS232 GND (Digtal GND)	↔ CAN Low	↔ CAN High	CAN GND	-	-

Frame connector X6 – RJ45 – CAN OUT

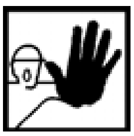
Pin	1	2	3	4	5	6	7	8
Signal	-	-	-	↔ CAN Low	↔ CAN High	CAN GND	-	-

Frame connector X7 – Brake resistor

The frame connector you can find at the side of IMD40. Here you can also connect the brake resistor (see frame connector X2 on page 28).



Pin	1	2
Signal	External brake resistor connector 1 Danger! Connector carries a high voltage!	External brake resistor connector 2 Danger! Connector carries a high voltage!



DANGER !

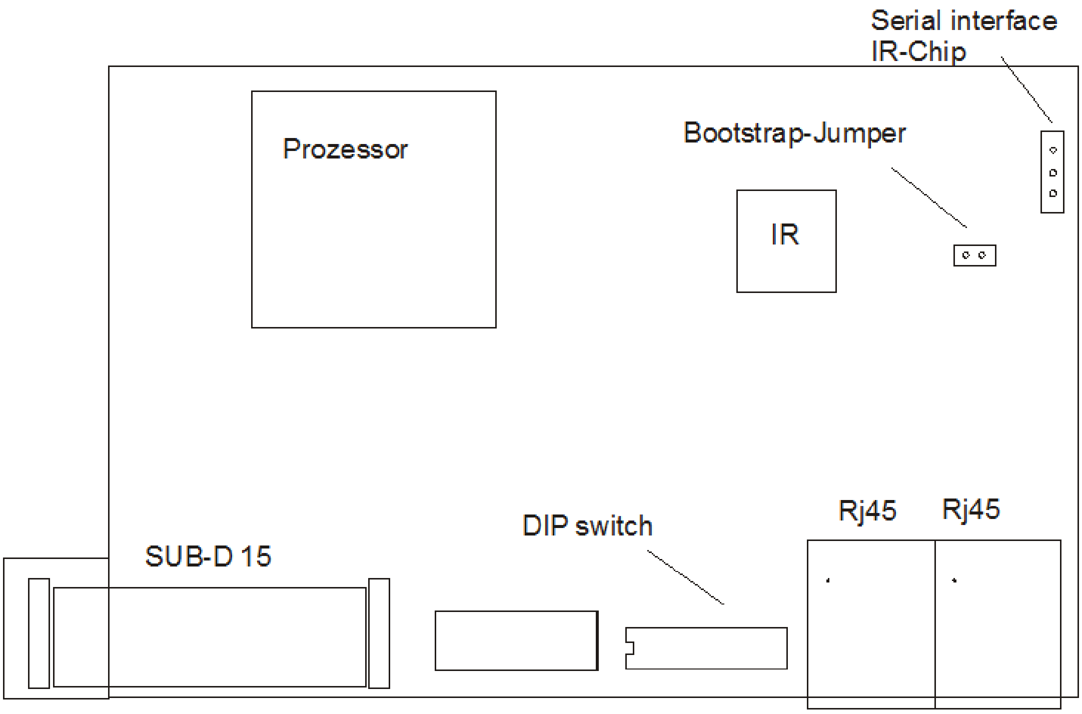
At work pin 1 and 2 carry a high voltage(DC bus voltage). Also after switch off pin 1 and pin 2 can carry a high voltage up to 5 min.(charging of the capacitor of the DC bus voltage). Acutal value of DC bus voltage see CAN object 0x2072

Jumper settings

For the normal operation, no settings via jumpers are required. Only for the loading process of a new software version by means of the bootstrap loader, the bootstrap jumper has to be connected.

(See also “The Firmware Update / via Bootstrap Loader command” on page 49.)

View at the upper side of the control board:

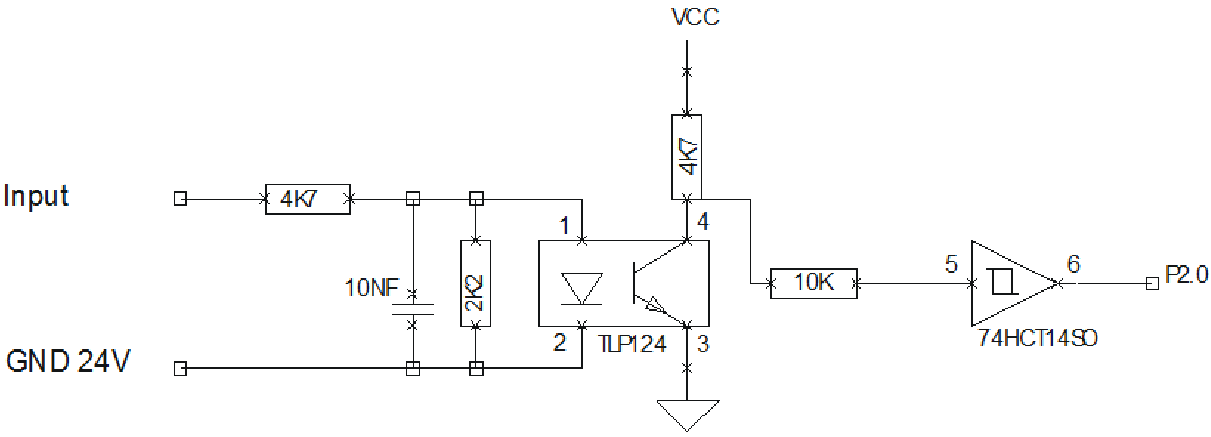


Wiring of the inputs and outputs

Digital inputs limit switches, reference switch, enable

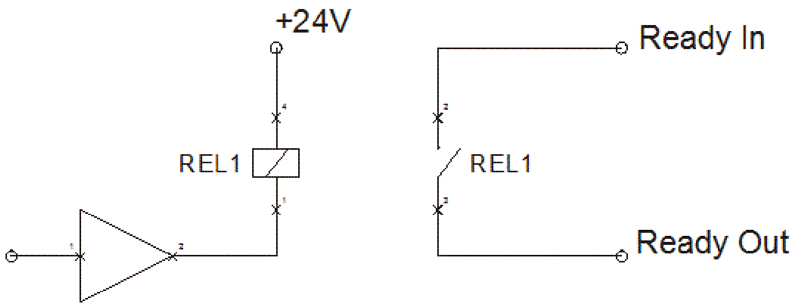
The evaluation of the inputs can be set by means of software.

(See “Inputs dialogue box” on page 75.)



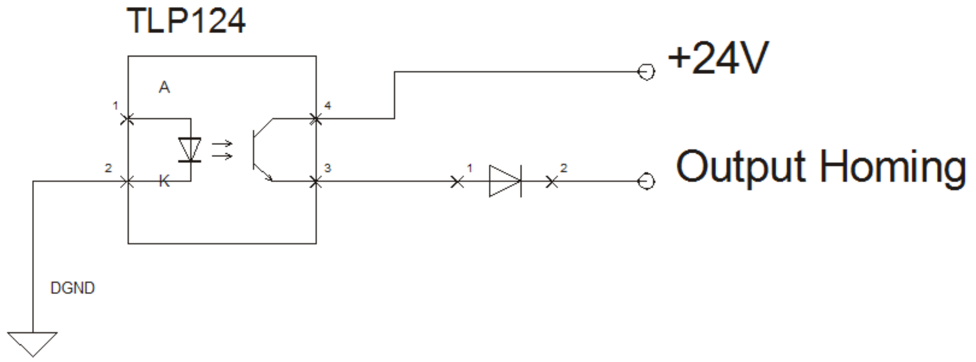
Digital output Ready

The Ready output is designed as a potential-free contact. It indicates the operational readiness of the module and is only set, if the module software works correctly (monitoring through internal watchdog timer). The design as a potential-free contact makes it possible to easily link several Ready outputs. This sum signal can be processed e.g. in the safety circuit.



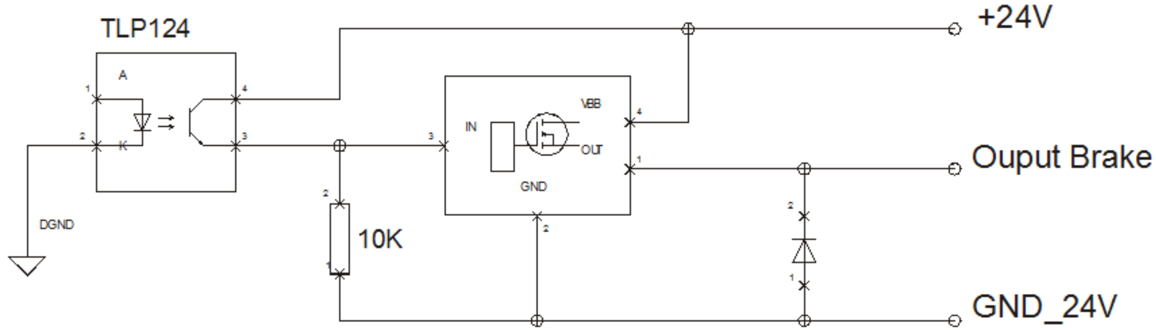
Digital output Homing

The Homing output can be used to shunt the limit switch in the safety circuit. It is set during the reference run (on limit switch).



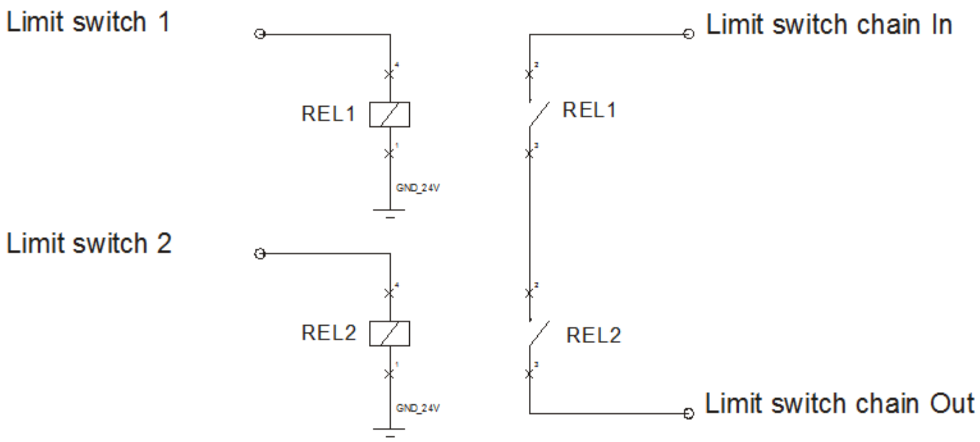
Digital output Brake

The Brake output is used for the brake control. The way in which the brake is controlled can be defined by means of the software.



Digital output Limit switch chain

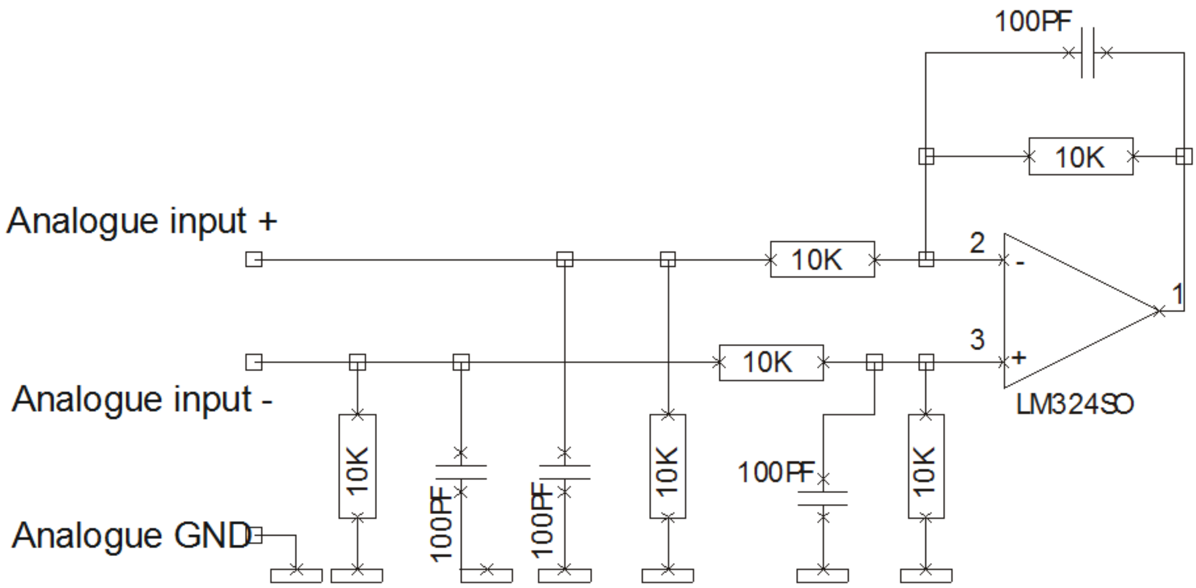
The Limit Switch Chain output is designed as a potential-free contact. It indicates, if both limit switches are not activated (closed potential-free contact). If one or both limit switch(es) is (are) activated, the contact is open. In the safety circuit, this signal can be used to monitor the limit switch.



Analogue input

The analogue input is preset for a voltage level in the area of -10 V … +10 V. The signal can either be connected to the positive input (+) or to the inverting signal (-). The reference potential is always Analogue GND.

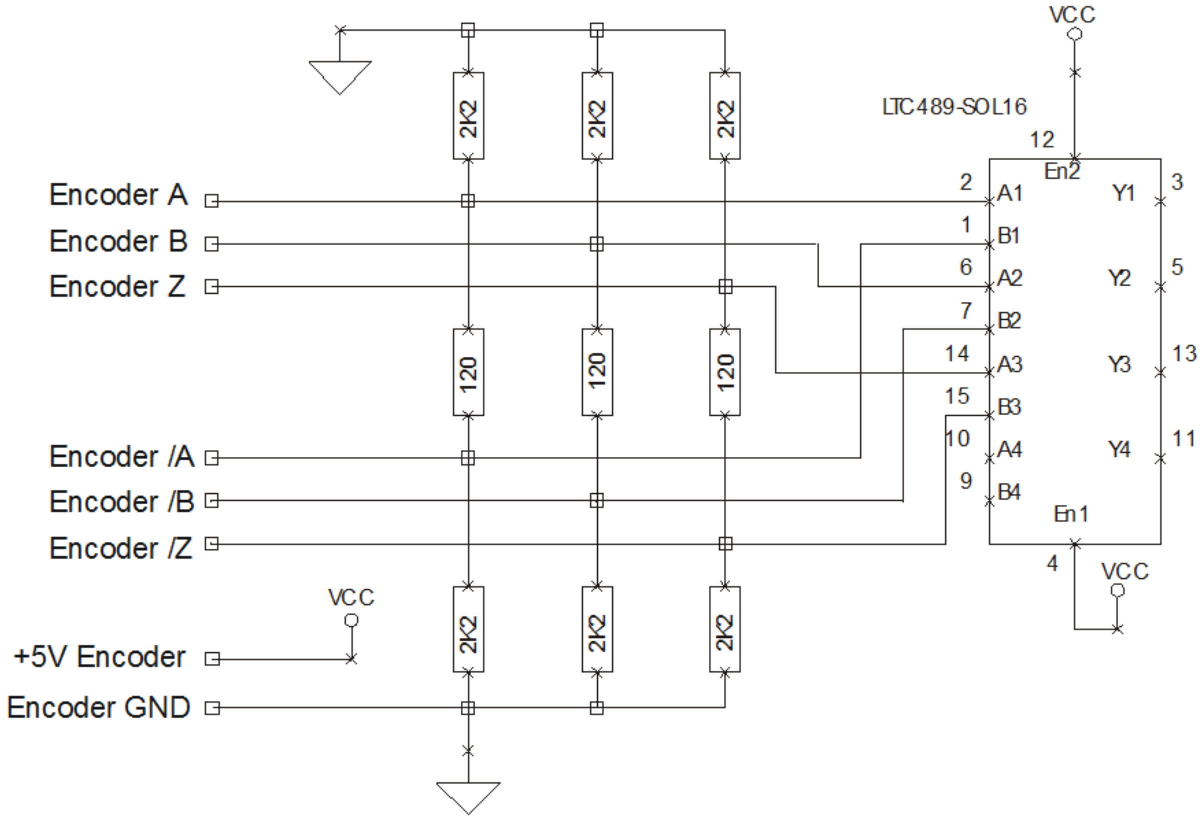
(See “Analogue Input dialogue box” on page 59.)



Encoder connections

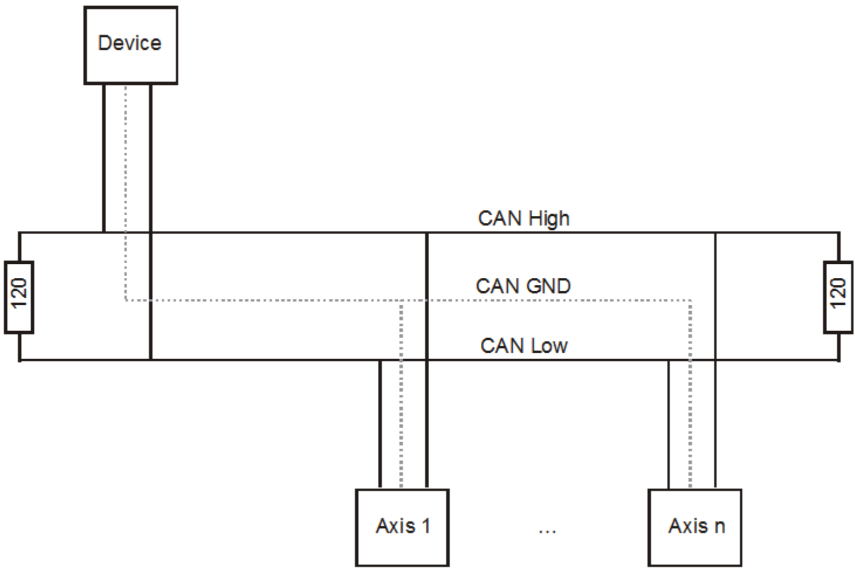
The drive module is preset for the connection of a quadrature incremental encoder with index signal. The transfer of the signals is carried out according to the RS422 specification. It is recommended to use shielded cables twisted in pairs for the encoder wiring.

The voltage supply of the encoder (5 volt) is provided by the drive module. The maximum supply current amounts to 100 mA.



CAN

The structure of the Can network is to be realised so as to guarantee that a 120 ohm terminating resistor is provided on both sides. The drive module exists a terminating resistor. It is activated by the DIP switch CAN terminator. The stub lines leading from the bus to the individual modules should not be longer than 50 cm in case of a baud rate of 1 Mbit/s.



For the CAN wiring, it is recommended to use twisted shielded cables with a characteristic impedance of 108 to 132 ohms. In case of very small network extensions, it may be possible to dispense with the connection of the reference potential (CAN ground) (not recommended).

Data transfer

The data exchange with the drive module can be realised via the two following interfaces:

- serial interface RS232
- Can bus

The RS232 interface and Can bus connection can be found on frame connector X4 (IMD20) and respectively frame connector X5 (IMD40).

Both interfaces can be used to access all objects (parameters) of the module. The access rights depend on the communication channel which was set by means of the DIP switch on the front side (see DIP switch on page 19).

Communication channel	**Switch position**	**Access right**
CAN	S6 0 Node address unequal 0	Can: Read and Write RS232: Read
RS232	S6 1 Node address unequal 0	Can: Read RS232: Read and Write
Analogue input (see “Modes of operation” on page 20)	S6 X Node address equal 0	RS232: Read and Write ("Controlword" and "Modes of Operation" cannot be written)

Transmission protocol for Can bus

In case of a communication via Can, the CanOpen protocol is used (see “CanOpen protocol” on page 89).

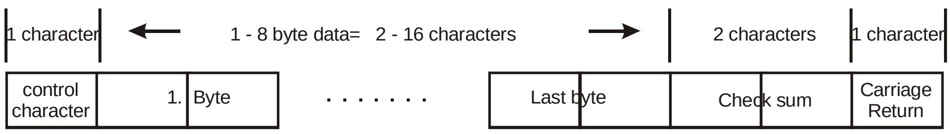
Transmission protocol for serial RS232 interface

For the serial communication (asynchronous, 1 start bit, 8 data bits, 1 stop bit and no parity bit), parts of this protocol are used, i.e. the SDO transfer, the emergency message and the boot-up message.

In case of the SDO, telegrams of a length of up to 8 bytes are exchanged between two participants (see “SDO” on page 91). These 8 bytes are transferred in hexadecimal format as ASCII characters, framed by an identifier and a checksum. Each message is terminated with a carriage return (CR, 0x0D). Contrary to SDO, not the entire 8 bytes have to be transferred, if the data contained are shorter than 4 bytes.

Likewise, the boot-up message and the emergency messages in case of a fault is sent from the module via the serial interface.

(See “Monitor window for serial interface” on page 52).



Control character	Meaning
C	SDO query (is received by the module)
B	SDO response (is sent by the module)
E	Boot-up message
1	Emergency telegram
F	Error during transfer F1: received too few characters F2: checksum error F3: received too many characters F4: error during the conversion of the characters

The checksum is formed on the basis of the 1 ... n data bytes (not the ASCII characters). It has to be taken into account that the control character is of no importance for the forming of the checksum. This only concerns the “pure” data bytes that are to be transferred. The checksum also has to be converted into ASCII characters again, before it is sent. This transfer protocol is applicable for both directions of transfer, i.e. both for the reading and the writing process.

Checksum = - ((Byte1 + Byte2 + ... + Byte_n) mod 256) or
= - (Division remainder of ((Byte1 + Byte2 + ... + Byte_n) : 256))

In the following, example 2 is used to show why a checksum of 0x10 (hexadecimal number) is created.

C *2F 60 60 00 01* ***10*** **CR**

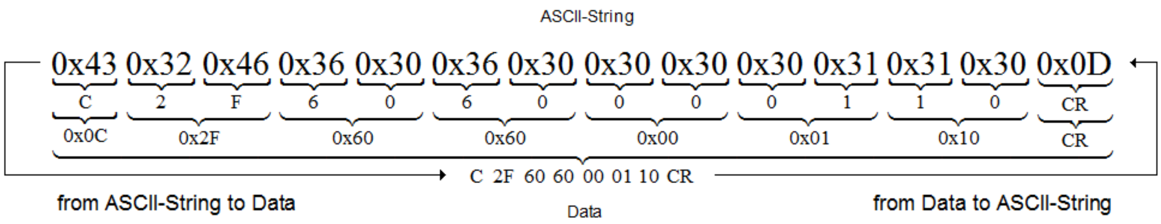
The control character ‚C’ and Carriage-Return ‚CR’ are not taken into account for the calculation.

Checksum	=	- (0x2F + 0x60 + 0x60 + 0x00 + 0x01)	(hexadecimal number)
	=	- (47 + 96 + 96 + 0 + 1)	(decimal number)
	=	- 240	(decimal number)
	=	- Division remainder of (240 : 256)	(decimal number)
	=	-240	(decimal number)
	=	0xFF10	(hexadecimal number)
	=	0x10	(hexadecimal number)

The two characters 1 and 0 for the last byte 0x10 of 0xFF10 are transferred as the checksum before the carriage return.

As already mentioned above, prior to transfer, we have to convert all data bytes (including the control character C but not the Carriage-Return ‚CR’) into the ASCII character. The so-called ASCII table is used for converting the data bytes into ASCII characters and recovering the data bytes from the ASCII string. As an example, let us take the data byte 0x3F. Here we then have the ASCII character 0x33 for 3 and the ASCII character 0x46 for F. It is important that the data to be converted into ASCII characters are in hexadecimal format. In our example we have converted the number 0x3F and not the figure 63 (0x3F in hexadecimal format = 63 in decimal format) into ASCII characters.

In our example, prior to the transfer, we convert all data bytes via the serial interface (except the Carriage-Return character) into ASCII characters.



The ASCII string is then the column of numbers

43 32 46 36 30 36 30 30 30 30 31 31 30 0D

This column of numbers is then sent directly to the receiver via the serial interface. On the basis of the non-converted Carriage-Return character, the output stage recognises the end of the ASCII string. Once the complete ASCII string has been received, it is then converted back into the original data. The output stage calculates the checksum from the converted data (0x2F 0x60 0x60 0x00 0x10). In the calculation of the checksum, the control character (0x0C) the received checksum (0x10) and the Carriage-Return (0xD) are ignored. The received checksum and the calculated checksum are compared to each other. If the two checksums are the same, the transfer is successful. The received data is further processed in the output stage. Our example involves the activation of the operating mode ”Profile Position Mode“. The output stage then switches into the desired mode. After the successful switch-over, the output stage sends the column of numbers

42 36 30 36 30 36 30 30 30 45 30 0D

back. For the receiver, this column of numbers is the ASCII string

0x42 0x36 0x30 0x36 0x30 0x36 0x30 0x30 0x30 0x45 0x30 0x0D

I. e. prior to sending, the output stage also first converts the data to be sent into an ASCII string.

With the exception of the Carriage-Return character 0x0D at the end, the receiver converts the ASCII string into the hexadecimal data bytes

B 60 60 60 00 E0 0D

The first data byte B is a control character and characterises the column of numbers as an SDO answer. The next 4 bytes are the actual data for the receiver. The second last data byte 0xE0 is the checksum calculated by the output stage. To find out if the transfer is correct or not, the receiver itself must form the checksum via the first 4 data bytes and then compare with the received checksum 0xE0. The Carriage-Return character 0x0D has the task of informing the receiver of the end of the ASCII string. The receiver does not need to convert the Carriage-Return character or further process it.

In the event of a faulty transfer, the power output stage sends a fault code back. E.g. if the checksums are not the same, the column of numbers

46 32 0D

is sent back by the output stage. The receiver interprets all data bytes of the this column of numbers as the ASCII string

0x46 0x32 0x0D

After the conversion (with the exception of the Carriage-Return character 0x0D), it receives the data bytes for the first two ASCII characters

F 2

The control character F indicates a transmission error and the number 2 specifies the error as a checksum error. The Carriage-Return character 0x0D can be disregarded. As you notice here, there is no checksum in the error message.

Note that the control character F only indicates one error in the transmission. An error within the power output stage - such as for example, the desired operating mode is not available – is reported to the receiver by the output stage via the returned data.

In case of a computer without serial interfaces, a USB-serial-adapter can be used. Unfortunately, not all adapters function perfectly, causing frequently transmission errors. In case of an error, the following measures can be taken.

- Update the driver software of the adapter.
- When you write your own application, define an error counter and, if necessary, reset the serial interface and try to make the transfer again. Two or three trials should be sufficient before ending the program. The program sequence is slowed down. However, this measure is better than to interrupt the program at the first transmission error.
- Absolute motion orders must be used with preference. In case of relative motion orders, in case of a transmission error, one has always the uncertainty whether the order has already arrived or not. In case of absolute motion errors, the order is simply sent again, and one should not be afraid that the motions are summed up, what is the case with relative motion order.

The RS232 monitor gives you the opportunity to watch the serial transmission and, where appropriate, to test it (see “Monitor window for serial interface” on page 52).

The CAN monitor allows you to watch the data transfer via the CAN-bus (see “Monitor window for CAN communication” on page 53).

Commissioning Overview

An essential condition for the successful commissioning is the correct pin assignment of all required signals and connections.

The DIP switch has to be configured according to the desired mode of operation and communication interface, before the module is switched on.

An auxiliary means for the commissioning is the supplied ACSetup programme. It contains a special menu point allowing a step-by-step commissioning of the module (see “Step-by-step commissioning” on page 56).

After the completion of the step-by-step commissioning, all drive- and motor-specific parameters should be set. If additional settings have to be carried out, e.g. the configuration of different communication parameters or device parameters which are not changed through the step-by-step commissioning, this can also be done by means of the setting programme (see “Object Directory dialogue box” on page 87).

Particularities as regards the commissioning of a gantry axis

In the gantry operation, two axes are synchronously controlled without a toothed belt or a bevel gear system. One axis works as the master and the other works as the slave. In order to commission a gantry axis, each axis (master or slave) has to be commissioned separately, before the machine is assembled. These parameters are stored as usual in the IMD20, IMD40 module. On the basis of the control and motion parameters of the master and slave axis, the ISEL CNC control determines the optimum motion behaviour for the gantry axis during the initialisation phase.

In a gantry axis, where the connection between the two axes is very stiff, it can sometimes happen that the two controllers work against each other during the movement. It then causes overcurrent-faults or vibrations in the axes. The following measures can be taken.

- The reference run in remote mode with index signal is to be used (see “Reference Run dialogue box – Homing dialogue box” on page 76)
- The “Couple Factor” can be reduced (see “Can Interpolation dialogue box” on page 73).
- Using of fixing screw swith rubber-washer to avoid a rigid connection.

By means of the CANSET programme, it can be stipulated, if an axis is in the gantry operation or not and which axis is the master and which the slave. In case of a slave axis, the user also has to define the maximum admissible deviation of position and the coupling factor between the master and the slave during the gantry operation in the “Can interpolation” dialogue box. The reference point is the start of the synchronous control. Every axis (master or slave) has its own reference switch or a limit switch serving as reference switch. The reference run uses the position of the switch and the reference distance to define the reference point. By sliding the switch and/or by changing the reference distance, you can influence the parallelism of the two axes. The procedure for putting a gantry axis into operation can be read in “Reference Run dialogue box – Homing dialogue box” on page 76:

The following settings have to be identical for the master and the slave:

- axis directions :

  The axis-directions of the two axes (master and slave) must logically be identical. I.e. with the positive movement direction of the gantry axis, the axis-direction of the master and axis-direction of the slave must also be positive. And with the negative movement direction of the gantry axis, the axis-directions of both axes must also be negative. Due to the freely definable axis directions, it is no problem to meet this requirement see (see “Direction dialogue box” on page 68)
- axis type: Linear and Rotary Axis (see “Can Interpolation dialogue box” on page 73)
- motion parameter unit µm or arcsecond ‘’ (see “Motor and Transmission dialogue box” on page 62)
- Reference run to ”Actual Position“ (see “Reference Run dialogue box – Homing dialogue box” on page 76.)

  If one axis has this reference run mode, you must also set this mode for the other axis. Otherwise, the reference run mode can be freely defined for both axes. For example, if the master axis moves to "Negative Limit Switch" during the reference run, the slave axis can move to "Index Signal - negative Start".
- Option „Remote Mode” of the reference run (see “Reference Run dialogue box – Homing dialogue box” on page 76.)

  If you want to have the controllable reference run mode for a gantry axis, you must activate the remote mode for both axes. If you do not want the controllable mode, this option must also be deactivated for both axes.
- Option ”Reference Distance Active“ (see “Reference Run dialogue box – Homing dialogue box” on page 76.)

  If you deactivate this option while starting up a gantry axis in remote mode, you must do so for both axes. And after starting up, this option must also be activated again for both axes.

All other parameters may differ for the master and slave.

Driving a rotary motor with a linear measuring system

Each Isel-motor has a built-in encoder on the motor shaft. The encoder is used to determine the movement position. It is also necessary for motor commutation. The encoder sits on the motor shaft, so it is unfortunately not able to detect such negative influences as backlashes, inaccuracies in the spindle..... As a result, there is often the desire to use a linear measuring system mounted parallel to the motion path instead of the motor encoder to detect the position of motion. The IMD20/40 output stages have only the connector X2 for IMD20 (see „Hardware description IMD20“ on page 22) and X3 on IMD40 (see „Hardware description IMD40“ on page 28) for the motor encoder. If you still want to use a linear measuring system, you must use signals A, /A and B, /B from the linear measuring system instead of those from the motor encoder. Please note that the index signals Z, /Z of the motor encoder and the hall signals are still necessary due to the motor commutation. The following table, the necessary pin assignment for connector is X2 of IMD20 and X3 of IMD40.

Pin	Signale	Linear measuring system signale
1	→ Hall A	
2	← Encoder voltage 5 V (Digital 5V)	← Linear measuring system voltage 5 V (Digital 5V)
3	→ Motor-Encoder /Z	
4		→ Linear measuring system /B
5		→ Linear measuring system /A
6	Logic +24V	
7	→ Limit switch 1	
8	GND_24V (Logic GND)	
9	→ Hall B	
10	Motor-Encoder GND (Digital GND)	Linear measuring system GND (Digital GND)
11	→ Motor-Encoder Z	
12		→ Linear measuring system B
13		→ Linear measuring system A
14	→ Hall C	
15	→ Limit switch 2	

The following points must be observed when using a linear measuring system.

- Inaccuracies detected by the linear measuring system represent disturbances for the controller. Therefore, the control noise level is generally higher. The movement is rougher. The controller parameters often have to be reduced. The control dynamics then suffer as a result of this.
- The output stage uses the signals of the linear measuring system to commutate the motor. Therefore, the linear measuring system should deliver approximately as many increments per motor revolution as the motor encoder would. Otherwise, the commutation is too imprecise. E. g. you have a spindle pitch of 4 mm and the motor encoder provides 4000 increments per revolution. Per motor revolution the output stage would get 4000 increments from the motor encoder. And the axis moves about 4 mm per motor revolution. The linear measuring system should then have a resolution of 1 µm in order to supply 4000 increments per motor revolution to the output stage. It is of course even better with a more accurate linear measuring system.
- Also with regard to commutation, you must adjust the values for the Hall sensor positions, the encoder index signal and the encoder resolution (see „Motor and Transmission dialogue box“ on page 62).

Remarks to the different hardware versions

The module IMD20/40 is supplied with the firmware for onboard-microcontroller and the start-up software ACSetup runnable on windows computers. The firmware file ACCON.HEX is in the same folder as the start-up software ACSetup. Use the menu “Extras\Firmware Update\Normal” of ACSetup to load the file ACCON.HEX once to the IMD20/40 in order to update the firmware for the onboard microcontroller. ACCON.HEX and ACSetup are optimally tailored to each other. Apart from the firmware updating where the firmware on the microcontroller hat an older version than the start-up software ACSetup, you should to use the firmware and the start-up ACSetup with the same version. In a case of version conflict you get at the beginning a warning message of ACSetup if you try to communicate with the module IMD20/40. Parallel to this, you always can use the menu “?\Info” to view the versions of the ACSeup and the firmware.

The lifetime of our product is much longer than the duration of availability of the onboard microcontroller. If the chip manufacturer discontinues the microcontroller we are forced to use a new microcontroller. But our software-adaption will guarantee the full compatibility for the module IMD20/40. As a user you should take care to use the right firmware for the onboard microcontroller. In the case of the wrong firmware you may get the error message “hardware error” from ACSetup. In the worst case, the module IMD20/40 stops the responding. In order to reactivate the module IMD20/40 you have to bridge over the “bootstrap jumper” and then use the menu “Extras\Firmware Update\Via Bootstrap Loader” to load the right firmware to the module (see “Jumper settings” for IMD20 on page 23 and “Jumper settings” on page 30). An incorrect firmware never causes damages to the module.

In the first time we have used microcontroller C168. Here, the microcontroller type is not always showed in the version number. Currently we use the microcontroller C269. The string “C269” is always included in the version number. It means a version number without the string “C269” indicates a module IMD20/40 with C168. Note, the name of firmware file for microcontroller C269 is ACCON_C269.HEX.

Examples for the motion control

Example 1:

Communication: Can

Baud rate: 1 Mbit/s (RS232: 57600 Baud)

Node address: 2

Mode of operation: Velocity control with ramp profile (profile velocity mode)

Set the "Modes of operation" (6060_h) parameter via SDO to the value 3.

Send		**Receive**	
ID	Data	ID	Data
602	2F 60 60 00 03 00 00 00	582	60 60 60 00 00 00 00 00

Set the "Control word" (6040$_h$) parameter via SDO successively to the values 0x06, 0x07, 0x0F. (Activate the "Operation Enable" status.)

Send		Receive	
ID	Data	ID	Data
602	2B 40 60 00 06 00 00 00	582	60 40 60 00 00 00 00 00
602	2B 40 60 00 07 00 00 00	582	60 40 60 00 00 00 00 00
602	2B 40 60 00 0F 00 00 00	582	60 40 60 00 00 00 00 00

The "Profile acceleration" (6083$_h$) parameter is used to set the desired acceleration, e.g. 100000 (0x000186A0).

Send		Receive	
ID	Data	ID	Data
602	23 83 60 00 A0 86 01 00	582	60 83 60 00 00 00 00 00

Set the "Target velocity" (60FF$_h$) parameter, e.g. 50000 (0x0000C350). The motion is launched.

Send		Receive	
ID	Data	ID	Data
602	23 FF 60 00 50 C3 00 00	582	60 FF 60 00 00 00 00 00

Example 2:

Communication: RS-232

Baud rate: 19200 Baud (Can: 20 kBit/s)

Node address: 7 (is only of importance for optional Can accesses)

Mode of operation: Positioning (profile position mode)

Set the "Modes of operation" (6060$_h$) via RS232 to the value 1.

Send	Receive
Data (ASCII characters)	Data
C 2F 60 60 00 01 10 CR	B 60 60 60 00 E0 CR

Set the "Control word" (6040_h) parameter successively to the values 0x06, 0x07, 0x0F. (Activate the "Operation Enable" status.)

Send	**Receive**
Data (ASCII characters)	Data
C 2B 40 60 00 06 00 2F CR	B 60 40 60 00 00 CR
C 2B 40 60 00 07 00 2E CR	B 60 40 60 00 00 CR
C 2B 40 60 00 0F 00 26 CR	B 60 40 60 00 00 CR

The "Profile acceleration" (6083_h) parameter is used to set the desired acceleration, e.g. 100000 (0x000186A0).

Send	**Receive**
Data (ASCII characters)	Data
C 23 83 60 00 A0 86 01 00 D3 CR	B 60 83 60 00 BD CR

Set the "Profile velocity" (6081_h) speed, e.g. 300000 (0x000493E0).

Send	**Receive**
Data (ASCII characters)	Data
C 23 81 60 00 80 38 01 00 43 CR	B 60 81 60 00 BF CR

Set the “Target position” ($607A_h$), e.g. 300000 (0x000493E0).

Send	**Receive**
Data (ASCII characters)	Data
C 23 7A 60 00 E0 93 04 00 8C CR	B 60 7A 60 00 C6 CR

Launch the absolute motion by setting Bit 4 in Control Word (0x001F).

Send	**Receive**
Data (ASCII characters)	Data
C 2B 40 60 00 1F 00 16 CR	B 60 40 60 00 00 CR

Query of the status word (6041_h)

Send	**Receive**
Data (ASCII characters)	Data
C 40 41 60 00 1F CR	B 4B 41 60 00 27 12 DB CR

Example 3:

Communication: Analogue input

Baud rate: (Can: 1 Mbit/s, RS232: 57600 baud)

Node address: 0

Mode of operation: Velocity control with analogue input (±10 V)

Set the control range by configuring the maximum speed with the help of the ACSetup programme (see “Velocity dialogue box” on page 70).

Activate the “Operation Enable” status via the enabling signal (dig. input 4).

Velocity target via analogue signal (±10 V corresponds to ± set maximum speed).

Fault states

If an error occurs, the module switches to the fault state, in which the final stage is switched off. The LEDs on the front side indicate the current error by a blinking sequence (see "EMCY" on page 96).

Overview or possible fault states:

Internal error code	**EEC**	**MEC**	**LED display**(1)	**Description**
1	0x6100	0x00xx	12	Internal software error
2	0x2320		13	Short circuit
3	0x4210		14	Overtemperature power amplifier
4	0x4220		15	Overtemperature motor
5	0x5010		16	Internal error of motor controller
6	0x7305		17	Fault Encoder Track A
7	0x7306		18	Fault Encoder Track B
8	0x7307		19	Fault Encoder Track Z
9	0x8100	0x0002	21	CAN error
10	0x7308		22	Deviation of position between the master and the slave is greater than the maximum admissible value.
11	0x8120		23	Node guarding failed
12	0x5441		24	Negative limit switch active
13	0x5442		25	Positive limit switch active
14	0x5444		26	No enabling signal
16	0x7309		28	Fault in slave axis
17	0x730A		29	master axis currentless
18	0x6010		31	Watchdog reset
19	0x3100		32	DC Bus voltage to low
20	0x6104		33	Synchronisation object was late
21	0x5530		34	Error during the deletion / writing of the Flash memory
22	0x6101		35	Data buffer overflow of the interpolation
23	0xFF10		36	Time-synchronisation-error during interpolation
24	0x3110		37	DC Bus Voltage to high
25	0x7400		38	Error hall sensor
26	0x2330		39	I^2t current limitation active
27	0x730B		41	Following Error – only with the modes "Profile Position Mode" or "Homing Mode" but not during start-up process with ACSetup
28	0x730C		42	Encoder step loss
29	*Earlier:* 0x6102 *Aktual:* 0x6105		43	*Earlier versions*: Violation of the software limit switch – but only in the Profile Position Mode *Actual version:* Homing movement path is over (A violation of the software limit switch is currently displayed in bit 11 of the Status word – see page 110.)
30	0x6103		44	Overflow of the 32-bit number range – but only in modes: Profile Position Mode and Interpolated Position Mode

(1) 1st digit = number of blinking impulses of both LEDs, 2nd digit = number of blinking impulses of one LED.

ACSetup programme

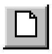
Menus

Commands of the File menu

The New command (File menu)

Use this command to create a new DCF file with default parameters.

Shortcuts

Tool bar: 

Keyboard: CTRL+N

The Open command (File menu)

Use this command to open an existing DCF file. If the online mode is enabled, the parameters of the module have to be read in once again.

Shortcuts

Tool bar: 

Keyboard: CTRL+O

The Save command (File menu)

Use this command in order save the active set of parameters in the DCF format under the current name and in the current directory. If the document is saved for the first time, the ACSetup programme indicates the Save As dialogue box in which you can name your set of parameters. If you would like to rename an existing document or move it to a different directory, you can also choose the Save As command.

Shortcuts

Tool bar: 

Keyboard: CTRL+S

The Save As command (File menu)

Use this command in order to save and name the active set of parameters. The programme opens the Save As dialogue box in order for you to enter the name for your set of parameters.

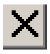
The 1, 2, 3, 4 commands (File menu)

Use the numbers and file names given at the end of the File menu in order to open the last four files you closed. For this open the number of the file to be opened.

The Exit command (File menu)

Use this command in order to exit your ACSetup session. Alternatively, you can choose the Close command from the application's system menu. The programme asks, if you would like to save the current set of parameters with the unsaved changes.

Shortcuts

Mouse: 

Keyboard: ALT+F4

Commands of the Connection menu

The Online Mode On/Off command

Use this command in order to enable or disable the online mode (see “Online operation” on page 54). In the online mode, all parameters have the same values in the ACSetup programme and in the drive module.

When the online mode is enabled, all parameters of the module are read in and then compared to the values of the current set of parameters in the setting programme. In case of a mismatch, you can choose, whether the module values or the values from the setting programme are to be used as current parameters.

Shortcuts

Tool bar: 

The Active Connection / CAN command

By means of this command you can configure the ACSetup programme so that the CAN bus is to be used as the active connection to the drive module, i.e. all parameters are read and written via CAN.

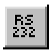
Shortcuts

Tool bar: 

The Active Connection / RS232 command

By means of this command, you can configure the ACSetup programme so that the serial interface is to be used as the active connection to the drive module, i.e. all parameters are read and written via RS232.

Shortcuts

Tool bar: 

The RS232 Settings command

This command opens the RS232 configuration. Here, you can choose a serial interface (COM1 - COM4) and set the baud rate.

The CAN Settings command

This command opens the CAN Configuration dialogue box. Here, the CAN driver and the node number of the desired drive module can be activated or set, respectively.

Commands of the Commissioning menu

The Step-by-Step Commissioning command

This command launches a wizard for the step-by-step parameterisation of the drive module. For this, the dialogue boxes for the setting of the drive and control parameters are run through in the given order. In this way, a save commissioning is guaranteed.

All setting dialogue boxes can be directly accessed with the help of the random order command.

Shortcuts

Tool bar: 

The Random Order command

With random order command, all setting dialogue boxes (e.g. current controller, mode of operation...) are displayed as tabs in the Commissioning dialogue box.

Shortcuts

Tool bar: 

The Reset command

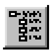
This command launches a reset of the drive module. The reset corresponds to the switch-off and subsequent switch-on of the module. All parameters are loaded with the last values saved.

Commands of the Settings menu

The Object Directory command

Use this command in order to access the Object Directory dialogue box. Via the object directory, you can access all drive module parameters. The parameters here are sorted according to an index and displayed with their CanOpen characteristics.

Shortcuts

Tool bar: 

Commands of the Extra menu

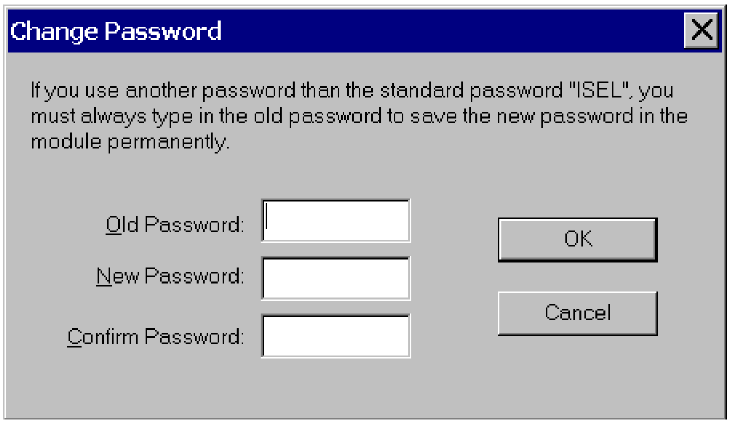
The Extended Functions command

By means of this command, you can switch the menu bar to a simplified view giving the most important functions or to an extended view with the possibility to access all existing commands.

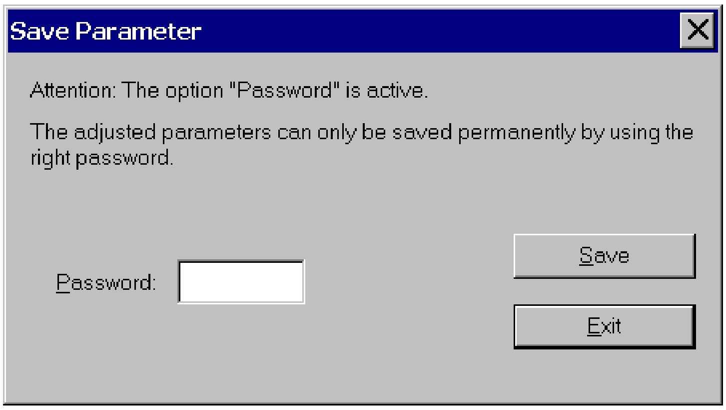
During the first start of the programme, only the simplified menu bar is displayed in order to facilitate the commissioning process.

Password

With the allocation of a password, the ACSetup programme allows users to protect the parameters in the drive module from unauthorised access. After a firmware update via the bootstrap loader has been realised, the module has the default password “ISEL”. Via the menu Tools\Password, it is possible to define a new password at any given point in time.



As long as the default password is valid, the users do not notice during the application of the ACSetup programme that a Password is required. If the password differs from the default password, however, users are required to enter the right password in order to permanently store the parameters in the module. It has to be taken into account that the module can be commissioned as usual. It is just not possible to permanently store the parameters, i.e. the set values are lost, when the module is switched off.



The Firmware Update / Normal command

Use this command in order to load a new firmware version into the module. The Hex file (e.g. ACCON_C269.HEX) must be in the same directory as the ACSetup programme and can be selected in menu Firmware update.

The update can only be realised via the serial interface.

All current settings of the drive module parameters are kept during an update. The update process must not be interrupted (e.g. by switching off the module, interrupting the connection, etc.).

If the update process could not be completed successfully, the drive module is not operational. In this case, a new update has to be realised with the help of the Firmware Update / via Bootstrap Loader command.

The Firmware Update / via Bootstrap Loader command

Use this command in order to load a new software application into the drive module, if there is no executable programme.

For this, it is necessary to install the bootstrap jumper on the module (see “Jumper settings” für IMD20 on page 23 and “Jumper settings” für IMD40 on page 30). After the module is switched on again, the update process can be launched. After a successful update, this jumper has to be removed again.

Existing parameter values of the drive module are overwritten with default values (see “Firmware update” on page 88).

Commands of the View menu

The Tool Bar command

By means of this command, you can show or hide the tool bar. The tool bar contains shortcuts for certain menu commands.

The Status Bar command

Use this command in order to show or hide the status bar. The status bar gives information on the programme or the selected menu.

The Serial Interface command

This command switches on or off the monitor window for the serial interface RS232. In this window, all characters of the active serial connection that were sent or received are displayed.

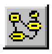
The CAN Monitor command

This command switches on or off the monitor window for the CAN transfer. In this window, all signals (sent and received) are displayed that are transferred between the ACSetup programme and the active drive module.

The Drive Status command

Use this command in order to display the status window for the current operating state of the drive module according to CANOpen / DS402.

Shortcuts

Tool bar: 

The Language command

With this command you can select the language of ACSetup. At the moment english and german are supported.

Commands of the ? menu

The Help Topics command

With this command, you receive an overview of the existing topics of the online help and can consult the desired topic.

The Info command

This command shows a dialogue box with the current programme version.

Programme user interface

Tool bar

The tool bar offers shortcuts for frequently used menu commands:

Opens a new parameter file

Opens an existing parameter file

Saves the open parameter file

Makes a connection to the drive module

Chooses CAN as active connection to the module

Chooses the serial interface as active connection

Indicates the operating state of the module

Launches the commissioning wizard

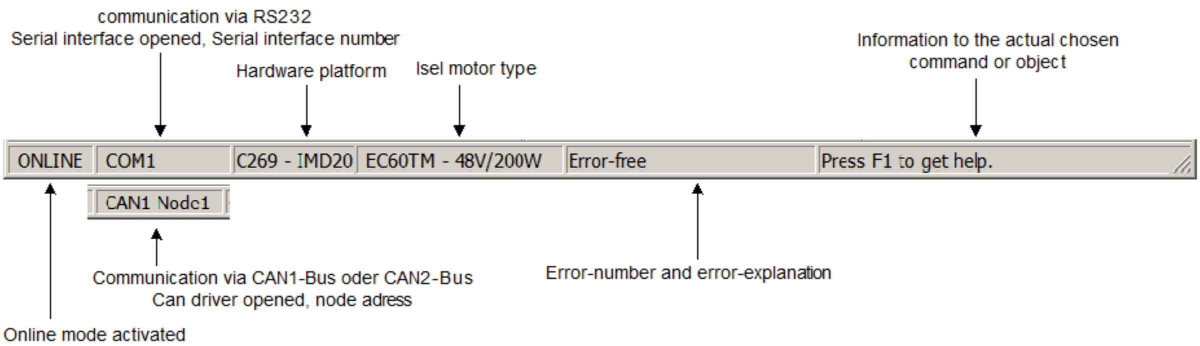
Shows all setting dialogue boxes as tabs

Displays a tree diagram with all module parameters (objects)

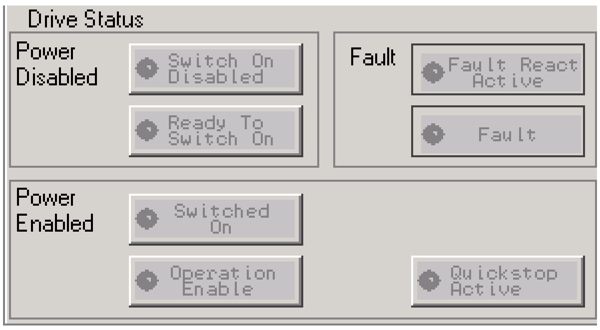
Launches the direct help

Status bar

The status bar gives information on the programme and on the opened interfaces and shows the current state of the connection to the drive module, the type of power output stage, type of motor, the amplify-error-number and their explanation.



Drive status



This display can be used to consult and change the operating state of the drive module in case of an active online mode. The status is queried about 5 times every second via the active connection.

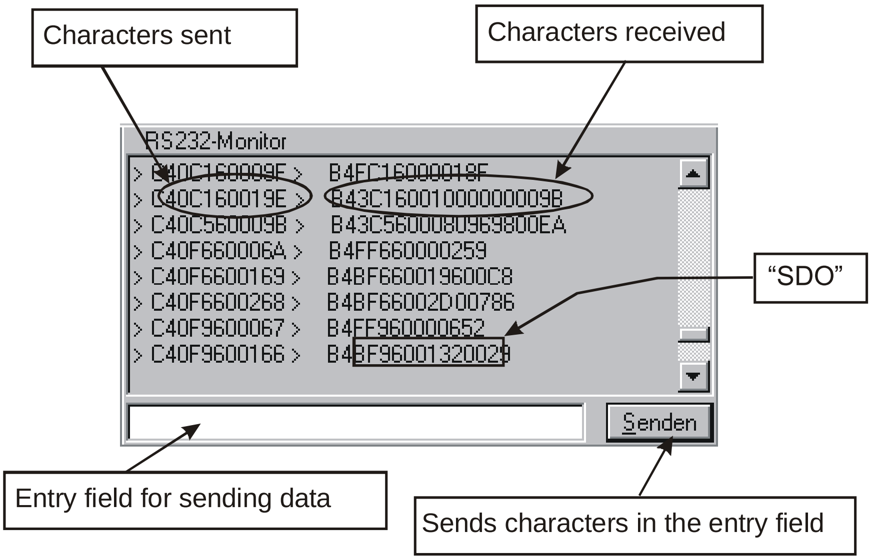
The individual states and transfers are defined in the CANOpen DS402 specification for drives (see “Device control” on page 109). The states are controlled via the “Control word” parameter (CANOpen object 6040h).

A light green LED indicates the current status. Dark green LEDs show the possible transitions to other operating states. A switch-over to the states illustrated in grey is not possible from the current state.

If, on the DIP switch, the “Analogue input” mode of operation is set (current or velocity controller), there are only the two possible states: “Ready To Switch On” and “Operation Enable” that are controlled by the enabling signal. A change via this dialogue box (or the “Control word” is thus not possible).

With the help of the View / Drive status command, the window can be faded in or out. If the window is faded out, the status query is interrupted.

Monitor window for serial interface



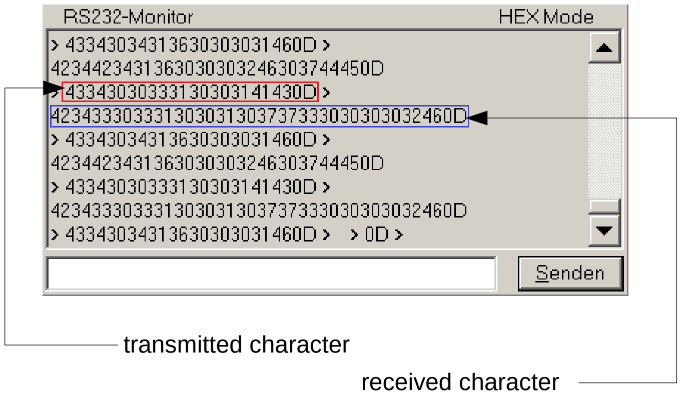
Here, all data are shown that are transferred via the serial interface (RS232). The characters sent are marked with two arrows by the ACSetup programme. Characters received are written behind this.

The transfer format of the serial interface was stipulated by analogy with the SDO transfer of CANOpen. The content of an SDO message ((max.) 8 bytes) is transferred in the Hex format as ASCII characters. An identifier is prefixed and two characters with a checksum for the data backup are suffixed (see also “Transmission protocol for serial RS232 interface“ on page 35).

For test purposes, different data can be sent to the module via the entry field and the <Send> button.

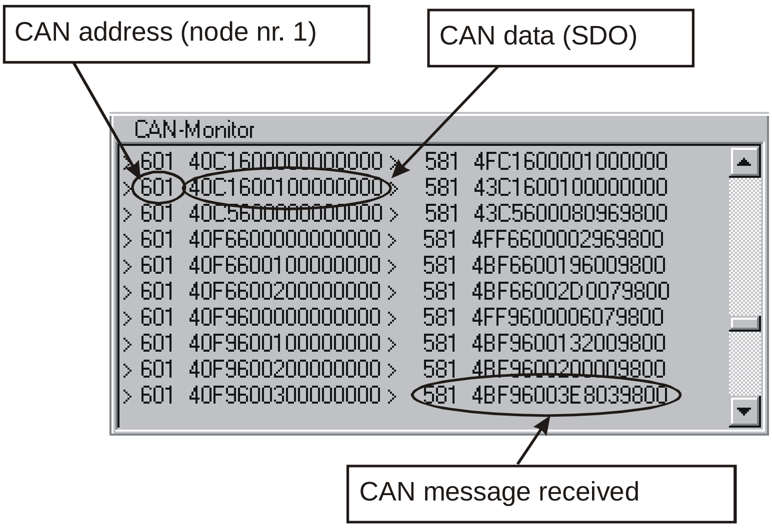
In the image above, the data telegrams are depicted as ”pure“ data. The Carriage-Return character (0x0D) is not displayed here.

If desired, the user can also have the data telegrams as ASCII characters. Here you can even see the Carriage-Return character (0x0D)



Switching between the two display modes is done by inputting one of the two commands **!hex** or **!dec** in the input field and then ultimately sending. Please note, that you can enter only "pure" data in the input window. I.e, here in this example, one inputs the string "C40031001AC" i.e. into the input window. Then, the concluding Carriage Return character is automatically appended when sending. In the display window, one gets to see the ASCII string "43343030333130303141430D" as "sent characters".

Monitor window for CAN communication



In this window, CAN data are shown which are exchanged between the ACSetup programme and the selected drive module.

For the communication, the CANOpen SDO protocol is used.

The setting of the node number for the connection with the desired module can be realised via the Connection/CAN settings menu command.

Programme functions

Create a data connection

The positioning module IMD20, IMD40 can internally store a configurable set of parameters, in which all drive and control settings are included. If no settings have been carried out yet, the parameters contain default values.

Furthermore, the ACSetup administers an active set of parameters that is read in, while the DCF file is opened. During the start of the programme, a default set of parameters is created.

Offline operation

If the online mode is not enabled, the ACSetup programme can be used to either edit a set of parameters from an existing DCF file or to create a new set of parameters. All changes in the setting dialogue boxes affect the internal values only and can then be saved in a file.

Online operation

In the online operation, the change of a parameter is saved both in the ACSetup programme as well as in the module.

For this reason, two sets of parameters have to be aligned with each other, as soon as the online mode is activated. For this, first of all, all parameters of the module are read in. Then, these values can be compared to those of the setting programme. In case of discrepancies, the system asks, in which direction the adjustment is to be carried out, i.e. if the set of parameters of the module is to be copied into the setting programme or vice versa.

In this way, it is e.g. possible to load a finished set of parameters from a file into the module. The loaded parameters can then be permanently stored in the module (e.g. via the Object Directory dialogue box).

For the activation of the online mode, use the Enable / Disable Online Mode menu command. The current state is also given in the status bar.

Before the online mode can be enabled, it might be necessary to first of all set the parameters of the interface and the active connection (see “The RS232 Settings command” on page 47 and “The CAN Settings command” on page 47).

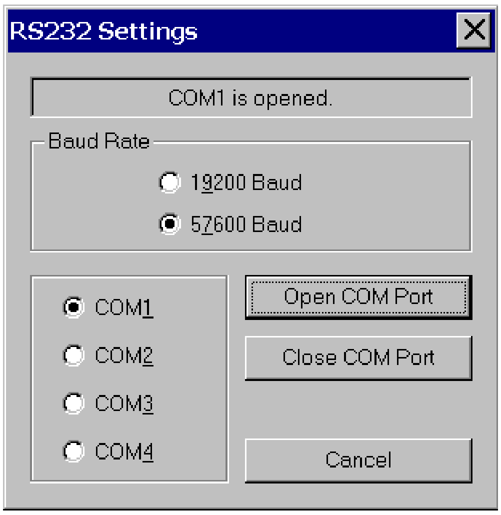
Choose Active Connection

The data communication with the positioning module can be realised either via the serial interface RS232 or via CAN. The Active Connection CAN / RS232 command is used to select a connection, before the online mode is switched on.

The connection parameters on the module and in the ACSetup programme must, of course, be identical (baud rate, COM no.) and the interface or the CAN driver must be open.

RS-232 settings

With the Connection / RS232 Settings command, you open a dialogue box in which you can define the baud rate and the interface to be used. Furthermore, it is shown here, which interface is currently used or active. The current interface is also given in the status bar.



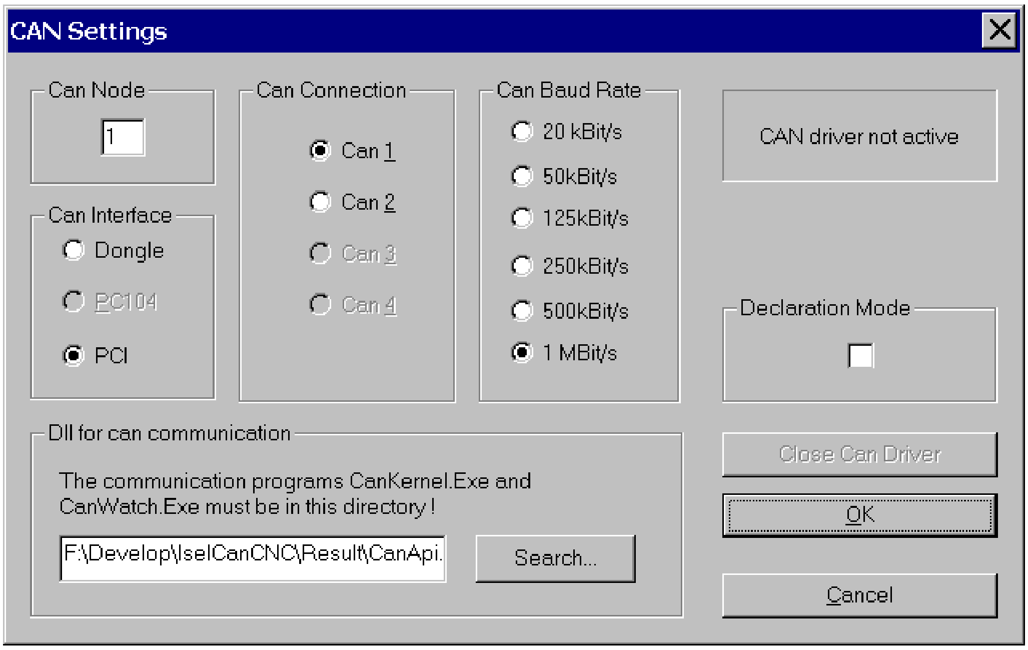
CAN settings

With the Connection / CAN Settings command, you open a dialogue box with which you can set the CAN node number, the baud rate, the Can connection and the Dll functions library for the CAN connection to the drive module. Furthermore, the current status of the CAN driver is indicated here. It has to be taken into account that the Dll functions library **CanApi.Dll** and the communications programme **CanKernel.Exe** as well as the monitoring programme **CanWatch.Exe** must be in the same directory.

If the CAN driver has already been launched from a different application, the ACSetup programme can normally no longer use the Can bus in order to communicate with the IMD20 / IMD40 module. Only the serial interface is concerned. If you nevertheless want to use ACSetup with the CAN bus, you have to activate the “Declaration mode”. In this case, you can observe the different parameters via the “Settings\Object Directory” menu. Writing the parameters, however, is only possible with some objects. The use of the ACSetup programme in the “Step-by-step commissioning” mode or in the “Random order” is not allowed. You should also take into account that the CAN driver has to be activated, before the declaration mode can be used.

The currently set node address and the status of the CAN driver are also indicated in the status bar.

(See also “Can transmission speed” on page 133).



Commissioning / Adjustment of parameters

Basically, all parameters of the drive module can be set via the Object Directory dialogue box. For the configuration of the CAN communications parameter, this is also both necessary and reasonable. However, for the commissioning of the actual drive functions, it would be a very unclear and lengthy procedure to find out the different parameters from the directory and to enter appropriate values.

For this reason, all relevant parameters were sorted in dialogue boxes according to functional groups. For this reason, a comfortable parameterisation is possible. In the Commissioning menu, there are two possibilities to access the setting dialogue boxes:

Step-by-step commissioning

This option should be used for the first commissioning of the drive module. The required setting dialogue boxes (differing according to the mode of operation) are run through in a predefined order. Thus, the highest level of security is ensured and uncontrolled movements of the axis e.g. due to wrongly connected encoder lines and non-adapted control parameters can be avoided to a large extent. The execution of the next step often depends on the successful completion of the current step.

During the first start of the programme, a simplified menu bar is shown which e.g. allows only the step-by-step commissioning. A permanent storing of data after the commissioning can be carried out via the “Object Directory dialogue box” of the Settings\Object Directory main menu.

If the drive module is operated together with the ISEL control card UPMV4/12, the ACSetup programme has to generate an enabling signal so as to ensure that the final stages can be switched on. For this, the path of the INI file belonging to the UPMV4/12 has to be entered in a special dialogue box.

Random order

This option allows to access all existing setting dialogue boxes. These are grouped as tabs in a superordinate dialogue box. The default buttons of this dialogue offer the following functions:

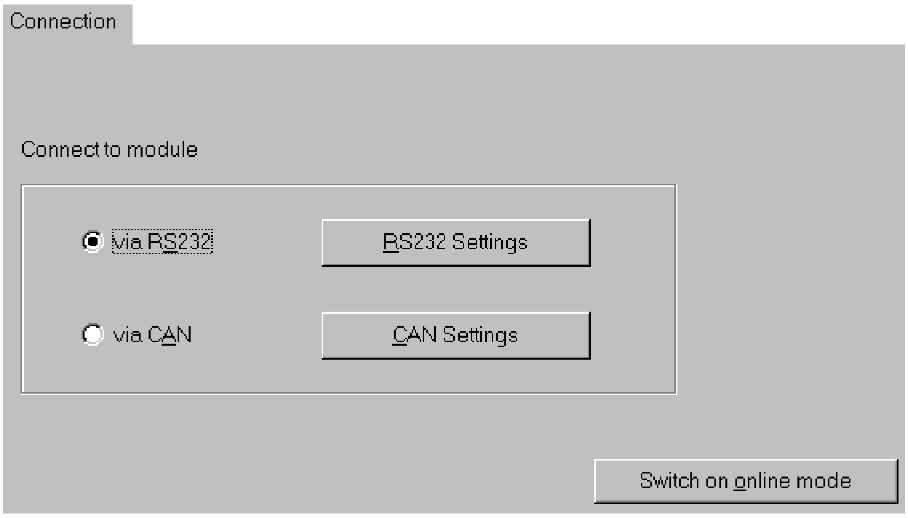
Changes are applied. The dialogue box is closed. A permanent storing take place. The permanent storing of data can be realised via the “Object Directory dialogue box” of the Settings\Object Directory main menu.

Changes are not applied; the parameters are not saved.

The set of parameters is only transferred to the module. A permanent storing is only carried out automatically, if the default password “ISEL” is in use. Otherwise it is not carried out. The permanent storing of data can always be realised via the <OK> button or via the “Object Directory dialogue box” of the Settings\Object Directory main menu.

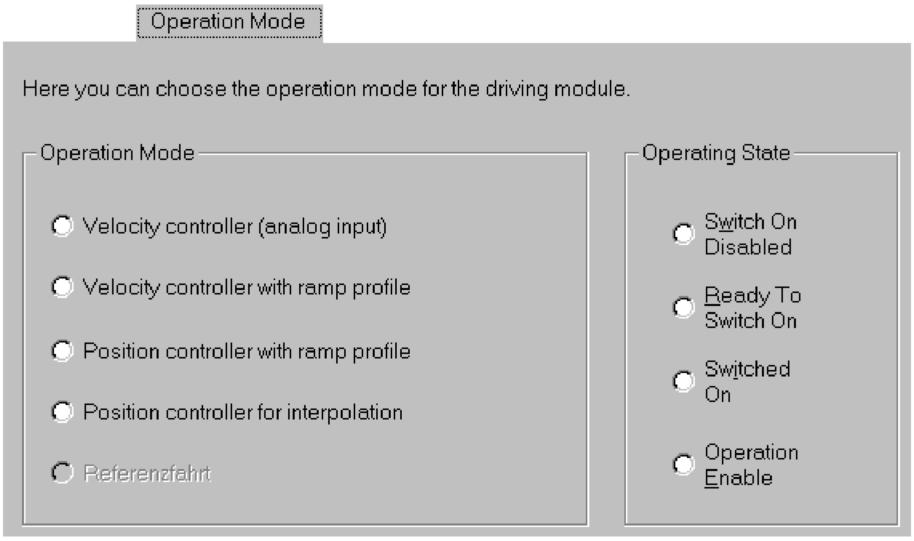
The commissioning via this option is reasonable, if one wants to make a quick change or detail improvements. In cases in which the drive module can move because of its own weight, even if the system is dead, a new commissioning shall also be carried out with the help of this option. For this, the parameters of the current, velocity or position controller have at first to be set to default values. The control is thus activated. On the basis of these default values, all other parameters can be determined. Here, in the beginning, it is hardly possible to carry out the commissioning via the “Step-by-step commissioning” option due to the inactive control.

Connection dialogue box



Here, you can select the active connection and adjust the corresponding settings. Afterwards, the online mode can be activated. During the step-by-step commissioning, the online mode is automatically launched with the <Next> button.

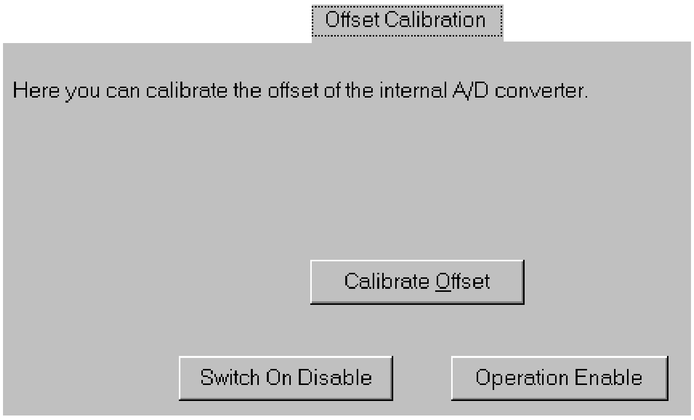
Mode of Operation dialogue box



The current mode of operation is given in this dialogue box and can be changed, if the DIP switch was not used to set the analogue reference input (velocity control preset). In case of the step-by-step commissioning, the number of the following setting dialogue boxes depends on the mode of operation chosen here. The changing of the operational status is only possible in case of the Commissioning in a random order.

(See “Modes of operation” on page 20 and “Mode of operation” on page 112.)

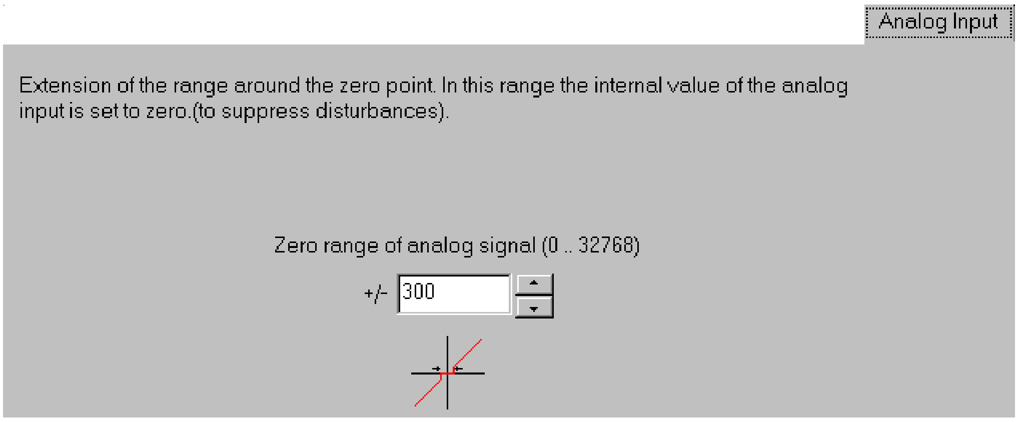
Offset Calibration dialogue box



With the offset calibration the offsets are measured and the drive module is calibrated.The offset calibration is realised automatically in the drive module and launched via the corresponding button.

(See “Offset analogue input” on page 158.)

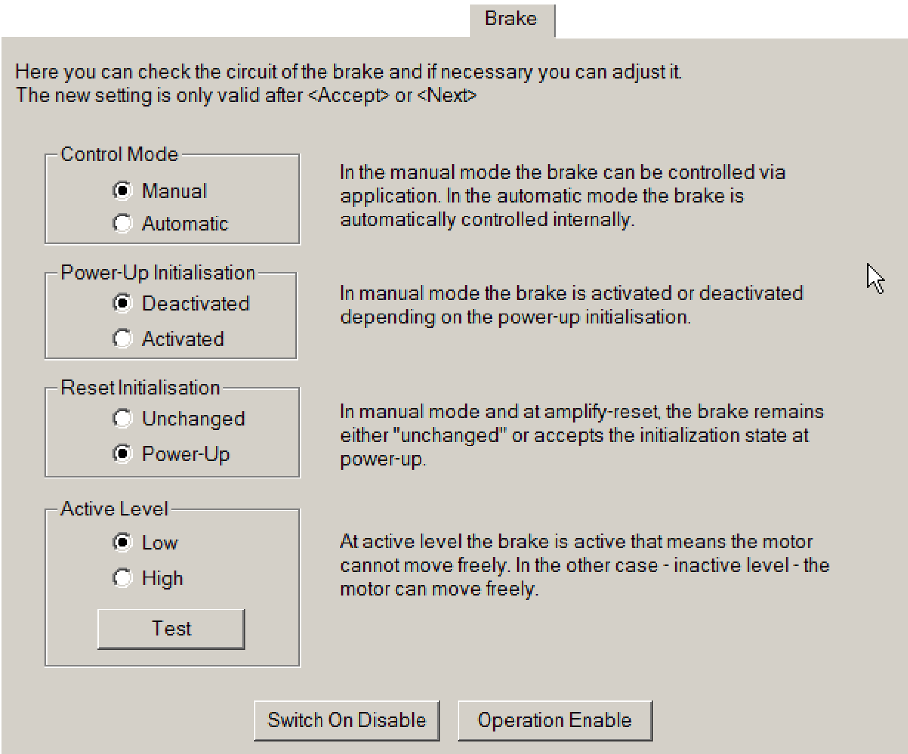
Analogue Input dialogue box



This setting possibility serves to eliminate small zero point deviations and voltage fluctuations on the analogue input. The digitalised voltage value is set to zero in the set area around the zero point. This parameter has only an effect on modes of operation with analogue input.

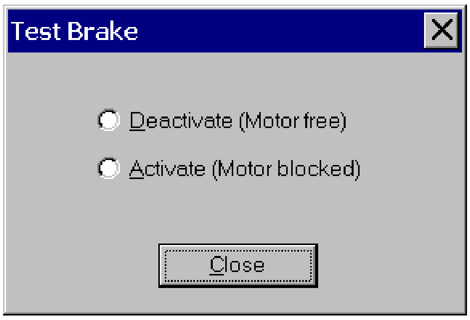
(See “Offset analogue input” on page 158.)

Brake dialogue box

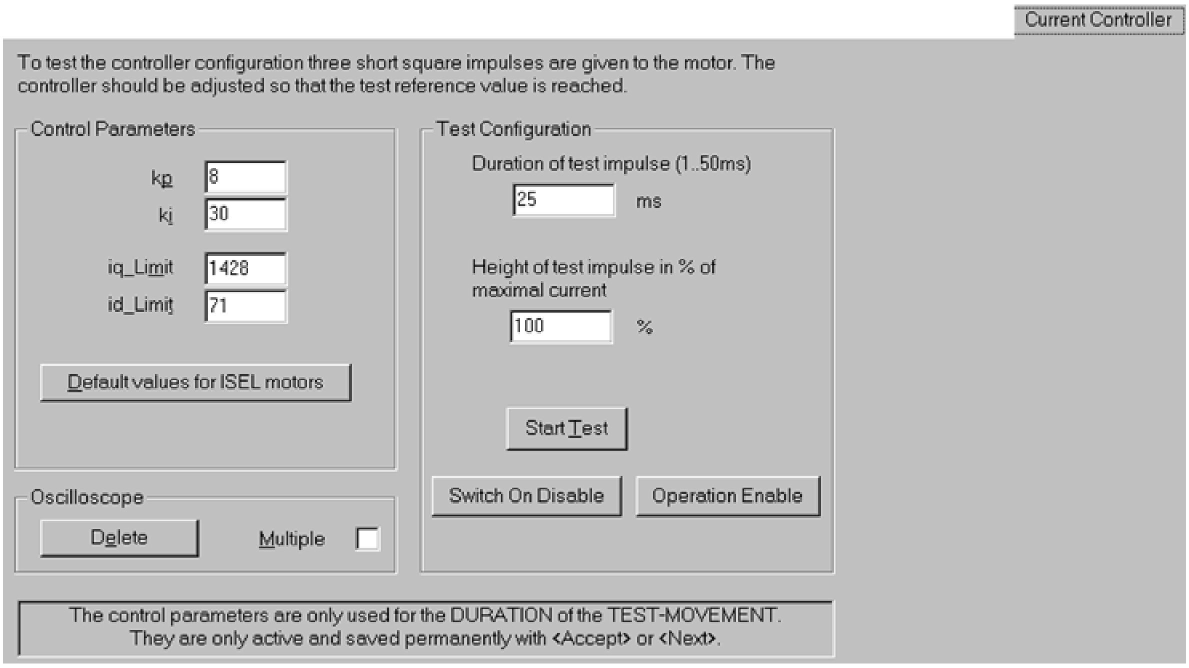


Output 3 (X1 pin 7 at IMD20, X2 pin 3 at IMD40) is specifically designed for the control of a connected brake. The use of the brake is optional. If a brake does not exist, the manual control mode has to be chosen. In this mode, the application software has to operate the output or brake itself. In the automatic mode, the brake is activated automatically, if the final stage is currentless. The motor can no longer move freely. If the final stage is activated, the brake is switched off. Then, the motor can be moved. All settings must in any case be controlled with the help of the <Test> button.

(See “Drive Data” on page 129 and “Digital outputs” on page 133).

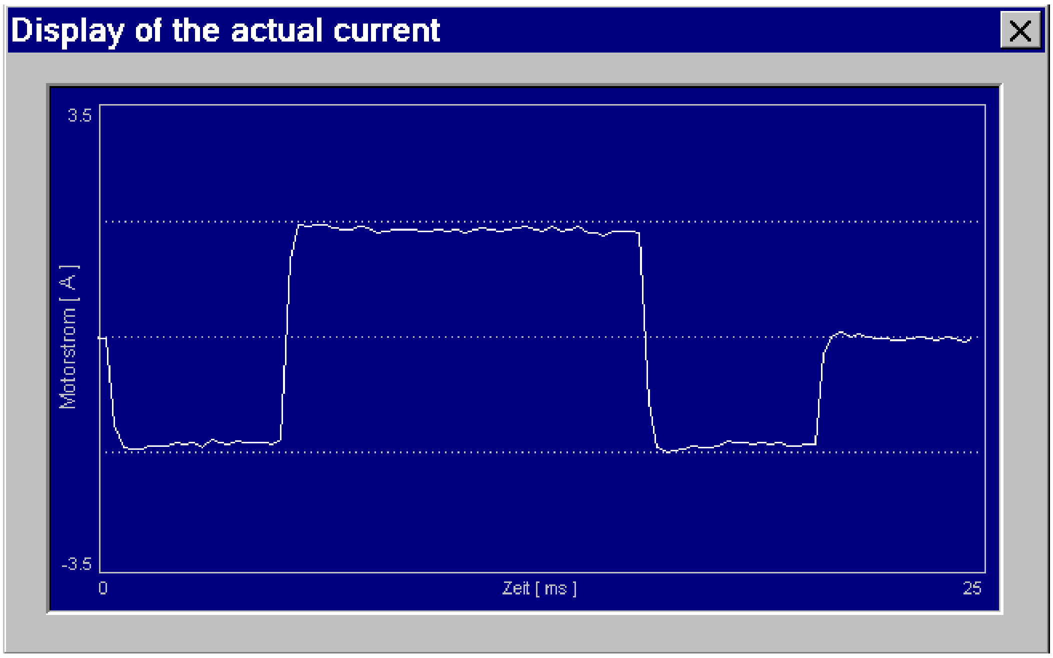


Current Controller dialogue box



The setting of the PI current controller is realised via the parameters **kp** (proportional amplification) and **ki** (integral amplification). Based upon the default values, the controller settings can be optimised. In order to facilitate the setting process, an internally generated rectangular signal (three impulses) is applied on the controller's input for the test. Here, the duration and the amplitude of the test signal can be set. The path of the actual current is recorded internally and then queried and displayed by the ACSetup programme (see Figure). For the setting of the controller, it has to be made sure that the target value (dotted line) is achieved and maintained.

Note: If the motor does not have to drive a large mechanical load and if the duration of the test impulse is too long, it may be that the motor goes into saturation and that the current decreases.



Display of actual current

On the basis of the actual current display, the step response of the controller can be assessed.

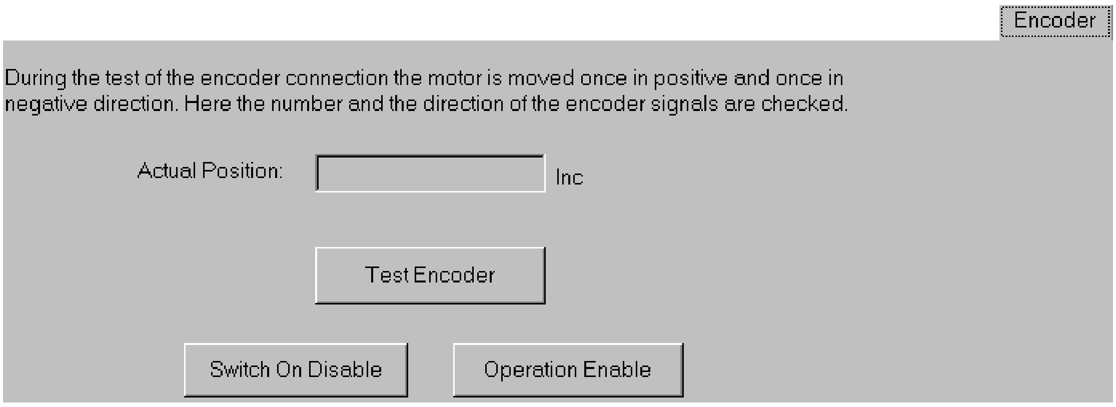
Only if the target value is complied with, the set current values for the rated and peak current correspond to the actual currents.

If the current step response is taken up again, normally the old actual current display is deleted and the new current actual value curve is displayed. If you wish to maintain old current actual value curves, to better assess the effects of parameter changes on the flow regulator, you can activate the switch "Multiple" in the field "Oscilloscope". The current processes can then be seen successively in four colours white, yellow, red, green. The complete display can be deleted at any time with the "Delete" button.

The values iq_Limit and id_Limit limit the output of the current controller. Here, these values can be changed. The values should not be set too high or too low. If the values are too high the control is not correct. The motor is noisy and the motor current is too high. If the values are too low the full power of the motor cannot be used. In the dialogue box "Motor and Transmission" these two values can be calculated out of the motor data. However, a manual adjustment of both values can be necessary.

(See “Motor and Transmission dialogue box” on page 62 and “Current control” on page 112.)

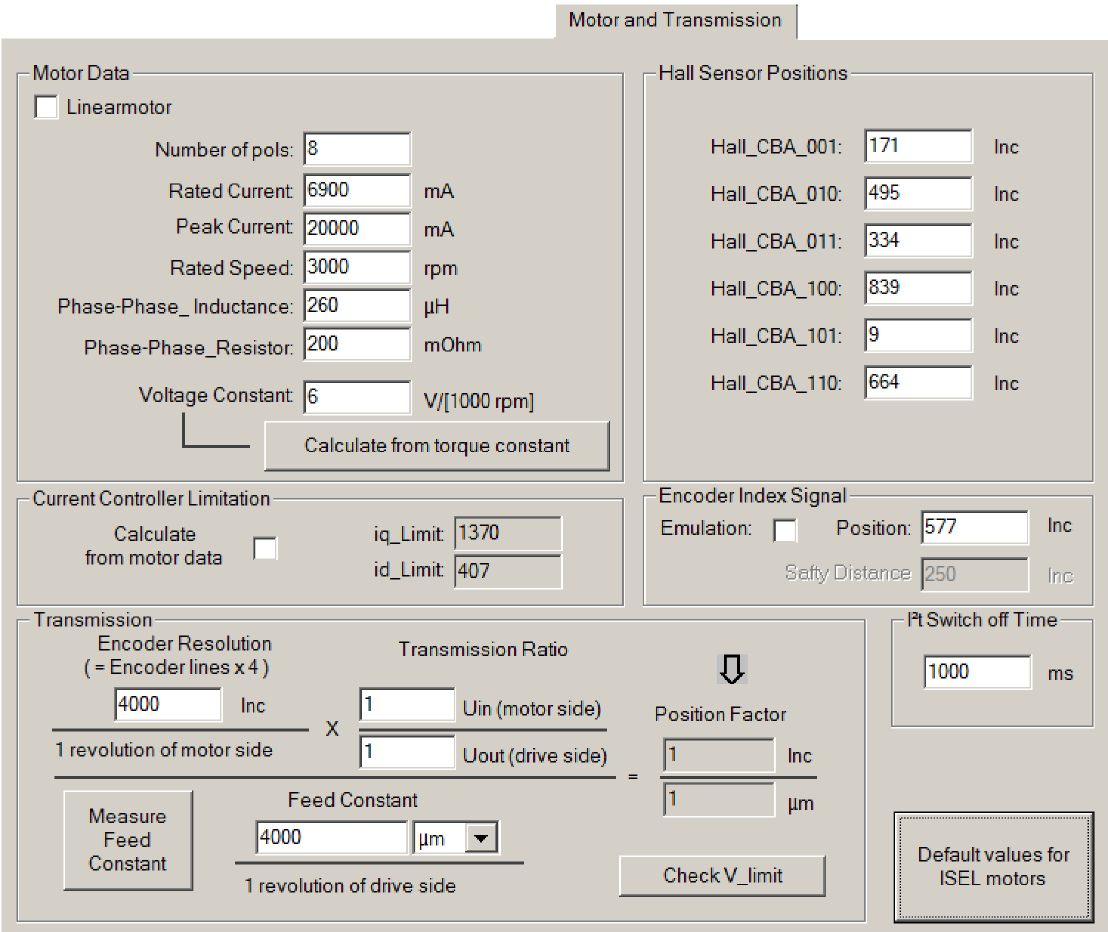
Encoder dialogue box



An operational encoder connection is an essential condition of the commissioning of the velocity and position controller. If the encoder is not connected or not connected correctly, the motor would make uncontrolled movements.

The test of the encoder connection is carried out internally in the drive module. At first, it is tested, if a certain number of impulses (e.g. 100) is exceeded at standstill (about 1sec.). This could be caused by an open connection or faults. Afterwards, the motor is moved in one direction, until a preset number of impulses (1000 Inc.) is achieved or until a preset time (100 ms) has been exceeded. A pause of about 2 seconds is to make sure that the motor is once again brought to a standstill. Afterwards, the same procedure is made for the other direction.

Motor and Transmission dialogue box



In this dialogue box the motor data and transmission data can be input. The current controller limitaion can be calculated from the motor data (iq_Limit, id_Limit). In the dialogue box "Current Controller" on page 60 the current controller limitation can be changed manually.

The conversion of units of length or angular measurement into the internally used Increments format is made in the drive module. For this, the objects of the Factor Group defined in the CANOpen specification DS402 are used.

The conversion factor (Position factor 6093_h) is the value which is used for all conversions. It is derived from the formula given, where the encoder resolution, the transmission ratio and the forward feed constant can be set. However, it is also possible to manually enter any value as conversion factor, if this is necessary. The units for the speed and acceleration are always derived from the position unit as Unit/Sec. or Unit/Sec.².

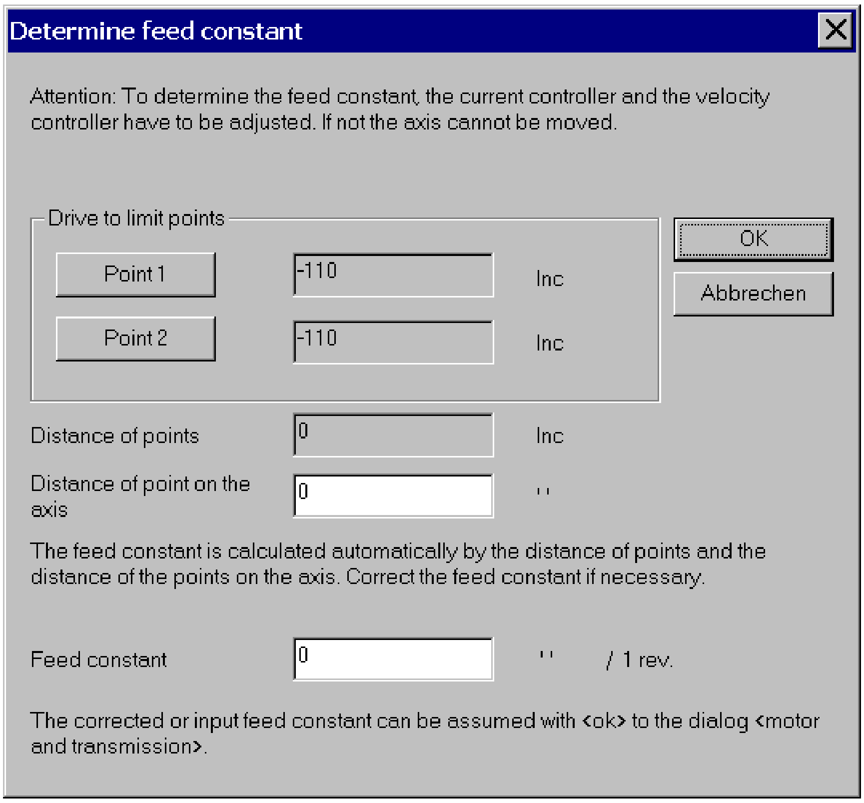
In the example above, a linear axis with a spindle pitch of 10 mm/rev without transmission is required. The position is to be given in the [µm] unit. Thus, the units for the speed and acceleration would be µm/sec. or µm/sec.², respectively.

If a rotary axis is used, it is reasonable to use a unit of angular measurement. Here, one can choose between arc second [“], arc minute [‘] and degree [°].

If a conversion is not wanted, the same value can be given for both the forward feed constant and the encoder resolution; the conversion factor would be 1.

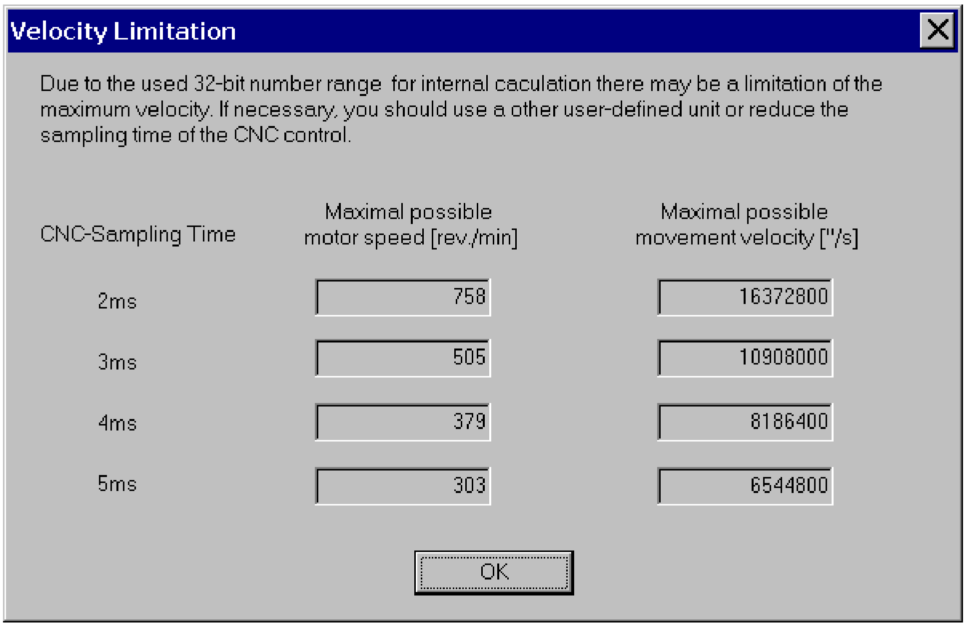
The encoder resolution must be indicated as the quadruple of the number of lines on the encoder (4-edge evaluation).

With the button „Measure Feed constant“ you can determine the feed constant out of the distance of two points on the axis. The following dialogue appears:



With the buttons „Point 1” and “Point 2” you can drive to two different points on the axis. Measure the distance of the two points and input the distance in the edit field “Distance of points on the axis”. The feed constant is calculated automatically while entering the measured distance into the edit field. Correct the calculated feed constant if necessary. With “OK” you can assume the feed constant into the dialogue box motor and transmission.

Internally, the power output stage uses the 32-bit range of numbers. With a very small conversion factor, which is often the case with a rotary axis with user unit arc seconds ["], it may be the case that the maximum possible motor speed is not reachable. Using the button "Check V_limit", you can check whether it may or may not happen in the output stage. In the case of a speed limit, the following dialogue appears.



To resolve this problem, the only other option you have is to use a different user unit, e.g. instead of arc seconds ["], you should take arc minute ['] to get a bigger conversion factor. A change in the CNC sampling time can also help to a limited extent. The change in the CNC sampling time of the Isel can control is done using programme CANSET.EXE and in the menu "configure CNC control \CAN\Hardware\Configuration CAN Hardware ...\Sampling Time".

The three hall sensors according to the 3 motor phases delivers directly after switch-on the information in which of the six magnetic field conditions the motor is located. The raw positions of the six magnetic conditions are given by the six here entered hall sensor positions. With this raw position information the motor can turn one revolution, until the first index signal comes. At this point of time the IMD20/IMD40 uses the entered encoder index position to commutate the motor precisely. At linear drives it is often the case that the used length measuring system delivers no index signal. For this case the option encoder index emulation must be used to generate the index signal. With the parameter “Safety distance”, an area around the possible index signal can be determined, in which the index signal is simulated. This area is necessary to disconnect possible errors on the hall signals. The bigger the area, the less exact the commutation can be, if the hall signals contain errors.For torque motors it is recommended to use the encoder index emulation (see “Motor Data” on page 127).

In the case you have a ISEL motor please press the button "Default values for ISEL motors". In the case you want to use your own motor together with IMD20/IMD40 communicate with us. We will help you to determine the hall sensor and index signal positions.

The motor encoder is used as standard for tracking the movement position and motor commutation. The motor encoder sits directly on the motor shaft. Therefore, it is not possible to detect inaccuracies along the motion path. In order to increase the accuracy, the use of a linear measuring system is one of the options. Connecting a linear measuring system to the IMD20/40 is described in section „Driving a rotary motor with a linear measuring system“ on page 39. Depending on the resolution of the linear measuring system and the spindle pitch, you still have to adjust the 6 values in the group "Hall Sensor Positions", the value for the "Position" in the group "Encoder Index Signal" as well as the value for the "Encoder Resolution" in the group "Transmission". The following example shows you how to make the adaptation for an ISEL standard motor.

First select the motor used via the button "Default values for ISEL motors". For example, the motor encoder provides 4000 increments per motor revolution. The spindle pitch is 5mm. The linear measuring system has a resolution of 1µm. For each motor revolution, the output stage receives 5000 increments from the linear measuring system (5mm pitch → 5000 µm / motor revolution → 5000 increments / motor revolution). There are 6 values in the group "Hall Sensor Position". The field "Hall_CBA_001" contains the original value 171. By using the rule of three calculation:

$$new\_value = \frac{original\_value \times linear\_measuring\_system\_inkcremente\_pro\_motor\_revolution}{motor\_encoder\_incremente\_pro\_motor\_revolution}$$

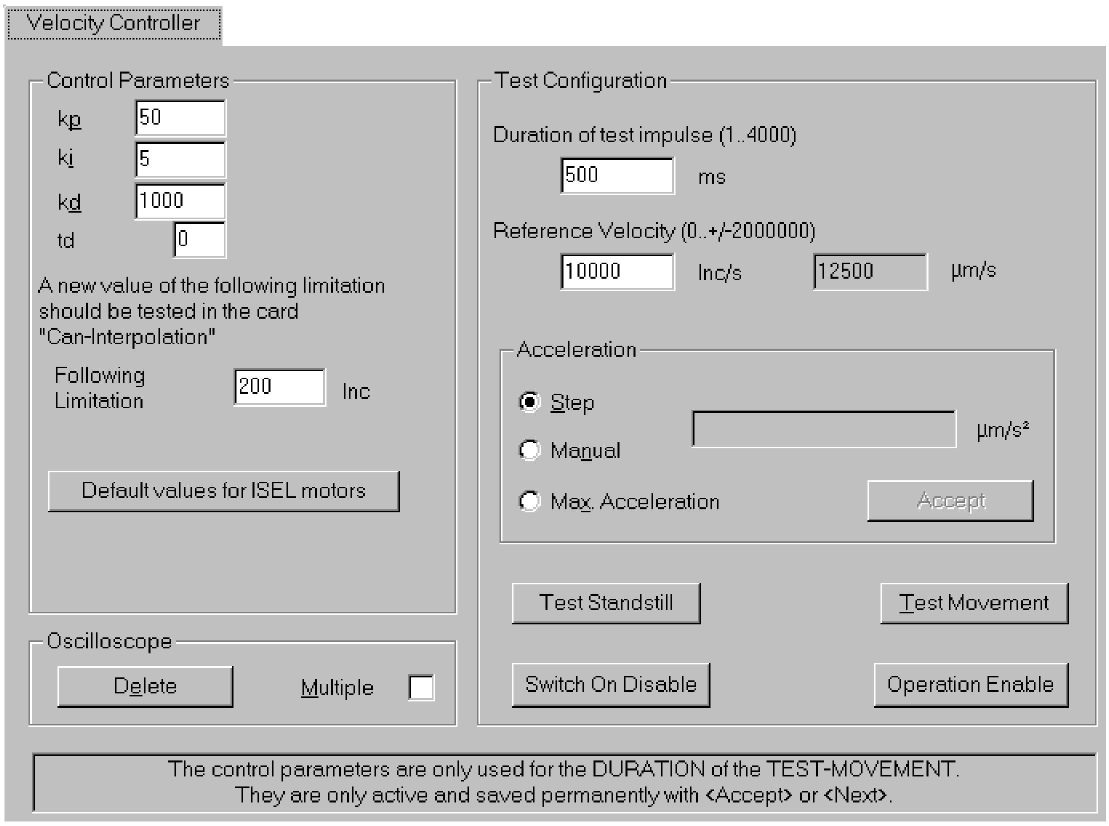
you get 213 as the new value for this field. Enter the new value in the field "Hall_CBA_001". The same applies to the 5 remaining fields in the group.

For the field "Position" in the group "Encoder Index Signal" you also do the same calculation. For the original value 577, you get the new value 721 (= 577 x 5000 / 4000). The value 721 must then be entered in the field "Position".

The new encoder resolution of 5000 increments per motor revolution must be entered into the field "Encoder Resolution" in the group "Transmission".

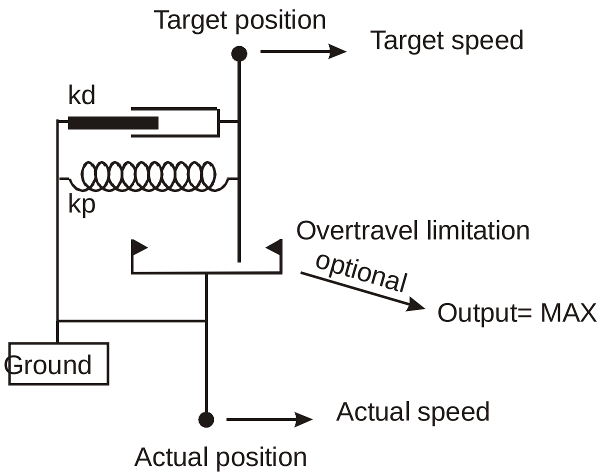
The adaptation for the linear measuring system is ready. Afterwards, the adjusted parameter set must be permanently stored in the output stage by pressing the "Apply" button.

Velocity Controller dialogue box



The setting of the PID velocity controller is carried out by analogy with the setting of the current controller (see “Current Controller dialogue box” on page 60). Furthermore, there are the parameters **kd** (derivative gain) and the related derivative time **td.** The td factor affects the sample time of the differential portion according to the function: sample time of the differential portion = controller-sample-time x (1 + td). This value should only be set greater than 0 in case of heavy axes. An increase of the td value by one corresponds to a duplication of the kd value.

Actually, the velocity controller is a position controller with the function to count the target position with a constant rate. This rate corresponds to the actual target velocity. Due to this principle, the **Following limitation** is a further parameter to take into account. Since, if the motor cannot follow its target position, a very great deviation would be the result, which would have to be caught up with a higher speed.



The deviation between the target and the actual position cannot be greater than the following limitation, since the target position follows the actual position, if this distance is exceeded. A too great value of an following limitation might lead to a permanent vibration of the axes. For the optimisation of the following limitation (see “Can Interpolation dialogue box” on page 73).

If the **Limit hard** option is enabled, the controller output is set to its maximum values, when the following limit is achieved. Otherwise, it has a value corresponding to the deviation.

For the testing of the controller setting, a test signal can be used, too. For this, three rectangular impulses are defined as target speed (the overall duration and amplitude can be set). The actual speed can be used to verify the controller setting. New control parameters are transferred only briefly to the drive module during the test and are then replaced again by the previous parameters. The parameters can be tested both at standstill as well as in motion.

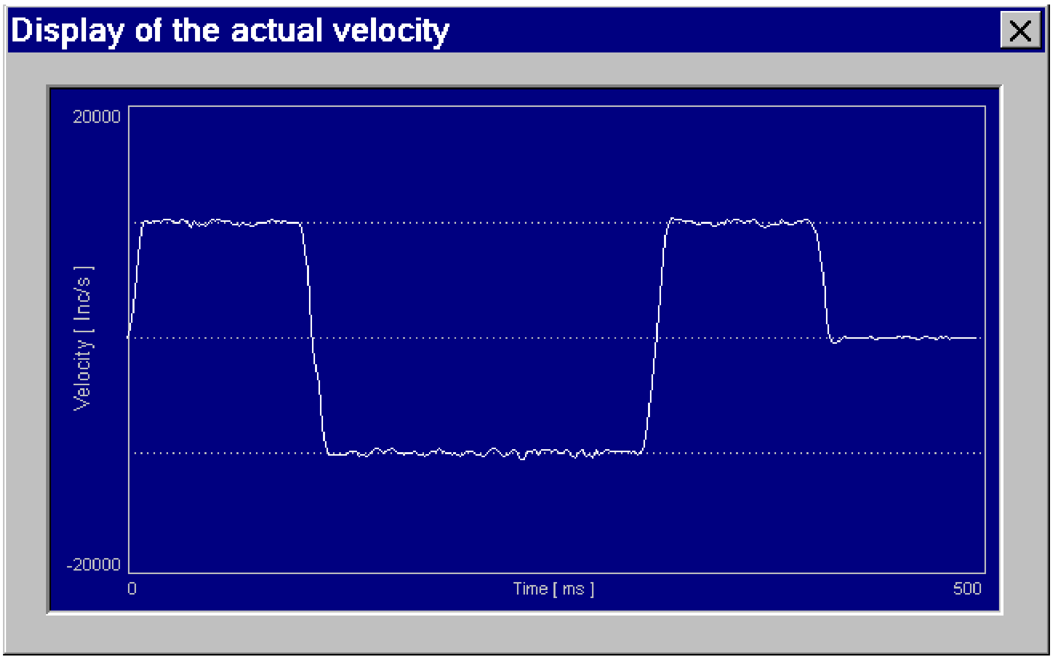
In the beginning, it is best to use the default values and to only then try different settings.

Some basic setting rules:

- In case of a strong following, reduce kp and / or increase kd.
- Too high a D proportion leads to a rough motor operation and to loud noises.
- An increase of td corresponds to a duplication of the D proportion (half kd).
- ki should not be greater than kp. Often, very small values suffice.

  If the axis does not come to rest, reduce the value for ki. A large value of ki can cause the axis to constantly oscillate a few increments around the desired position.
- For the setting of the following limitation, a higher target speed should be chosen. The reduction of the following limitation leads to a reduction of the following and a flattening of the ramps. Reduce the following limitation up to the point, at which the ramp slope is still as high as in case of a high following limitation. It might be possible to increase kp further. Then, the “Limit jard” option can be activated in order to achieve a higher acceleration.

The requirements for the controller can differ considerably from application to application. A smooth course of the actual speed with a small following (see Figure) e.g. is a good compromise for many application purposes.



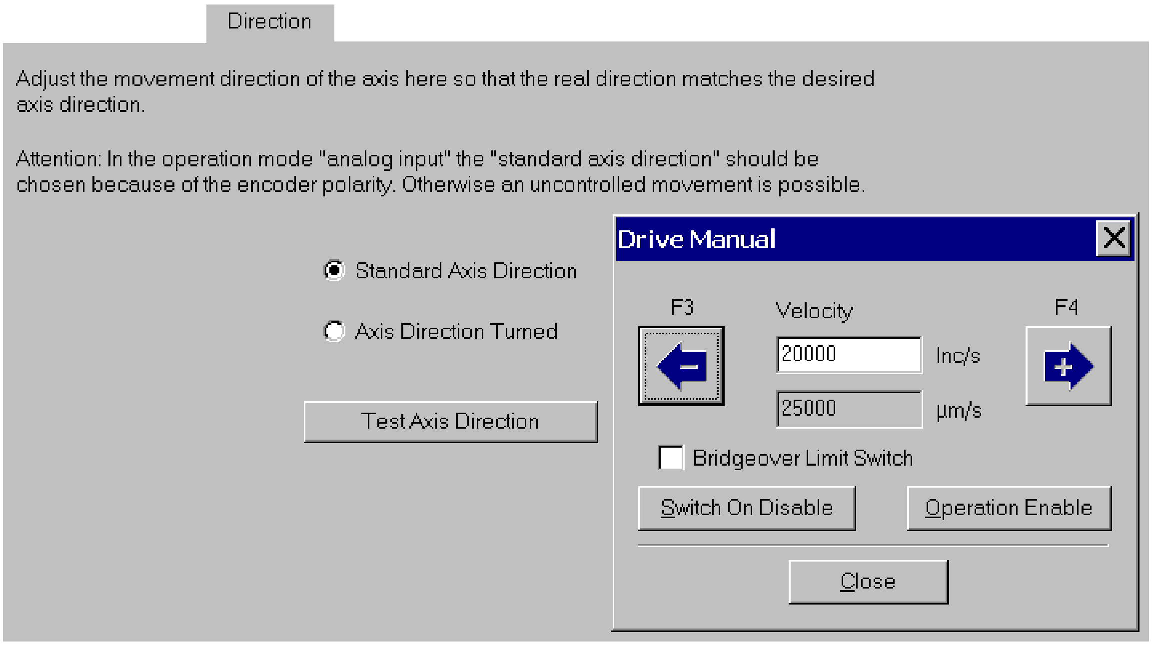
Display of actual velocity

When testing the speed controller, the controller gets a step function as an input signal. The acceleration of a speed step function has a major influence on the step response. Too great an acceleration value usually leads to a large overshoot, which is often the case with the option 'Step' in the field 'Acceleration'. If a small overshoot is required, you still have the option "Maximum acceleration". Here, you can test whether or not the set maximum acceleration is good enough. If necessary, you can still change the acceleration with the option "Manual", to find an optimum value. Using the button "Apply", the currently measured acceleration becomes the new maximum acceleration.

If the speed step response is taken up again, normally the old speed display is deleted and the new speed actual value curve is displayed. If you wish to maintain old speed actual value curves, to better assess the effects of parameter changes on the speed controller, you can activate the switch "Multiple" in the field "Oscilloscope". The speed curves can then be seen successively in four colours white, yellow, red, green. The complete display can be deleted at any time with the "Delete" button.

(See “Velocity controller parameters” on page 115).

Direction dialogue box



Every axis (linear axis, rotary axis) driven with the positioning module should have a defined axis direction.

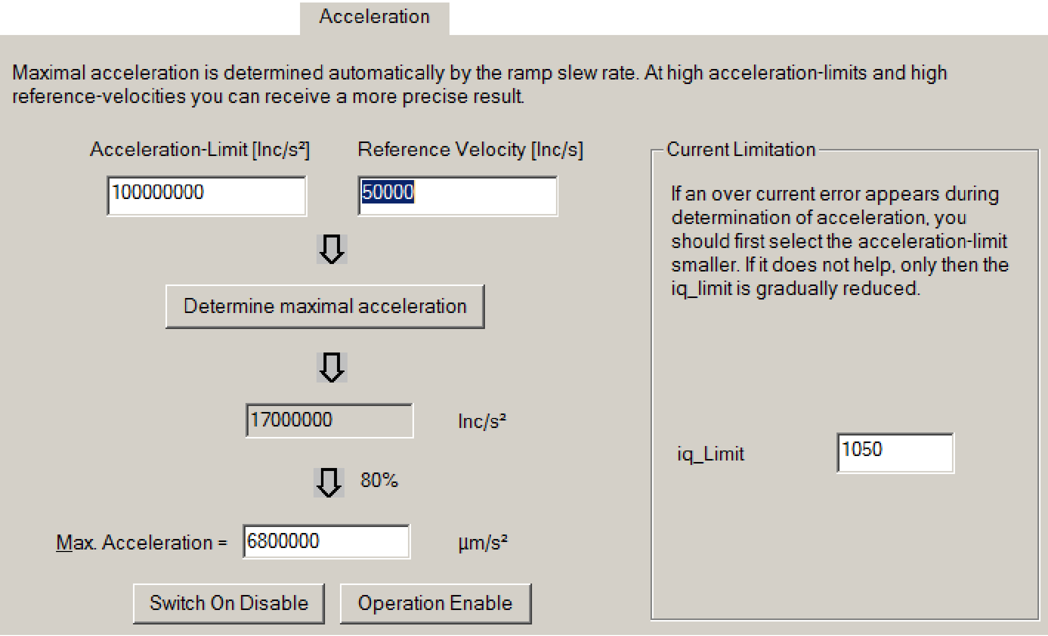
If the motor is connected correctly, it should rotate to the right in case of a positive direction and to the left in case of a negative direction (viewed from the front).

Here, the rotation direction of the motor can be adapted so that it corresponds to the defined axis direction of the connected mechanical facilities. For this purpose, a dialogue box can be displayed by means of which the axis is moved into the positive or negative direction.

If the analogue mode of operation (current or velocity controller) is activated, the default axis direction has to be chosen. Otherwise, the movement of the motor during the switch-on is uncontrolled.

Note: If the axis direction is changed, the limit switch inputs of the positive and negative limit switch are also inverted (see “Inputs dialogue box” on page 75 , “Reference Run dialogue box” on page 76) and “Axis direction” on page 128).

Acceleration dialogue box



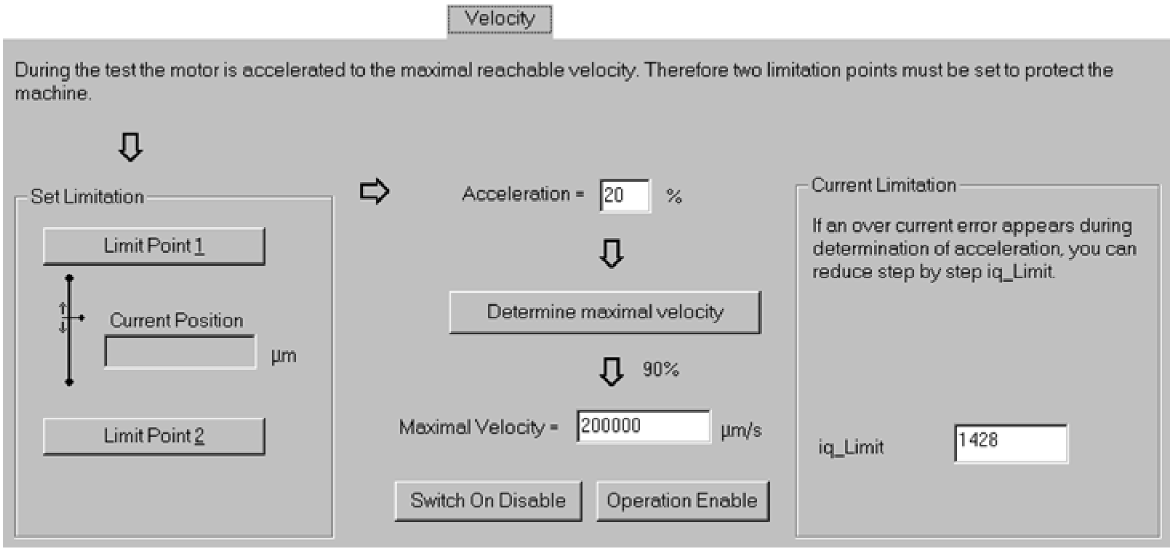
The maximum acceleration is the limitation for the acceleration of all motion functions with a ramp (profile velocity mode, profile position mode). I.e. the acceleration cannot become greater than this value, even if a different parameter (e.g. profile acceleration) contains a higher value. During the interpolation move (interpolated position mode), the cnc master control uses the maximum acceleration for ramp calculation.

The determination of the maximum acceleration can be carried out automatically. For this, a rectangular signal is used in order move the motor back and forth. At the same time, the build-up times of the ramps are measured and averaged. This value is limited to 80 % in order to have a certain reserve for the control.

Important: If a current short circuit error (error number 13) occurs during the automatic determination of the maximal acceleration, the user should first gradually reduce the acceleration limit value until the determined acceleration value reaches approximately the acceleration limit value. If the short circuit error is still present, the current limit must be gradually reduced (mainly iq_Limit) until error 13 no longer occurs.

(See “Limits of movement” on page 131).

Velocity dialogue box



The maximum speed is the limitation for the speed of all motion functions with a ramp (profile velocity mode, profile position mode). I.e. the speed cannot become greater than this value, even if a different parameter (e.g. Target velocity 60FFh) contains a higher value. During the interpolation move (interpolated position mode), the cnc master control uses the maximum acceleration for ramp calculation.

In the mode of operation "Velocity controller with analogue input", this value corresponds to the maximum achievable speed (-10 V …-+10 V relate to –max. velocity ... +max. velocity). See "Mode of Operation dialogue box" on page 58 and "DIP switch" on page 19.

The determination of this parameter can be carried out automatically. Here, it is, however, necessary to define a motion sector, since the axis might cover a long way during the acceleration to the maximum speed and the subsequent braking and may not drive into the mechanical stop.

In order to determine this motion area, at least one limit point has to be set in which the axis is driven to the corresponding spot via a Teach dialogue. In the beginning, both limit points contain the actual position.

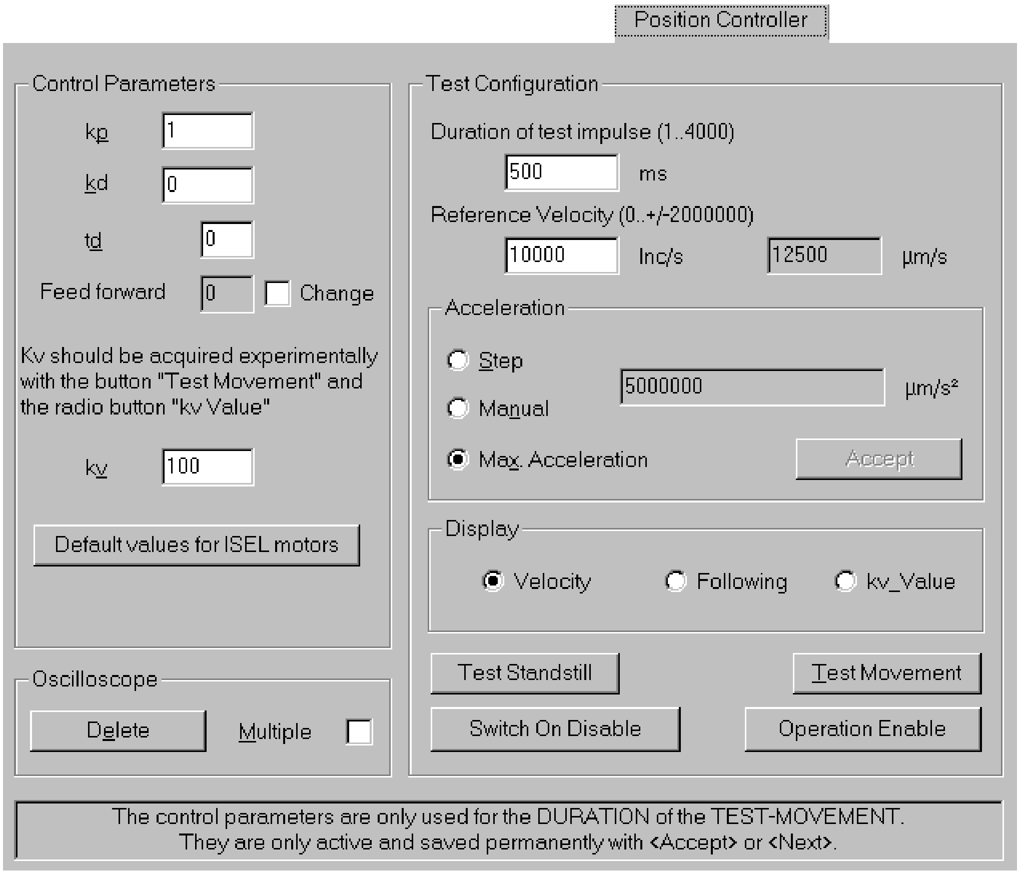
The acceleration is by default set to 20 % of the maximum acceleration in order to go easy on the mechanical parts, if e.g. the motion sector becomes too small.

The maximum speed is determined internally and then limited to 95 %, in order to dispose of a good reserve for the regulation.

Important: If a current short circuit fault (fault number 13) occurs during the automatic determination of the maximal velocity the user must reduce step by step the current limitation (in the main iq_Limit) until the fault 13 does not occur anymore.

(See "Limits of movement" on page 131).

Position Controller dialogue box



The position controller is a PD controller with velocity feed forward. This is superimposed on the velocity controller, i.e. the velocity controller has to be set first. A feed forward always leads to better dynamics, but also to an overshoot. Normally, the default values for he feed forward should be applied.

For the test of the settings, a test signal is again available consisting of three impulses with different signs. The ramp slope can be chosen between the (previously determined) maximum acceleration, a maximum ramp (jump) and a desired acceleration.

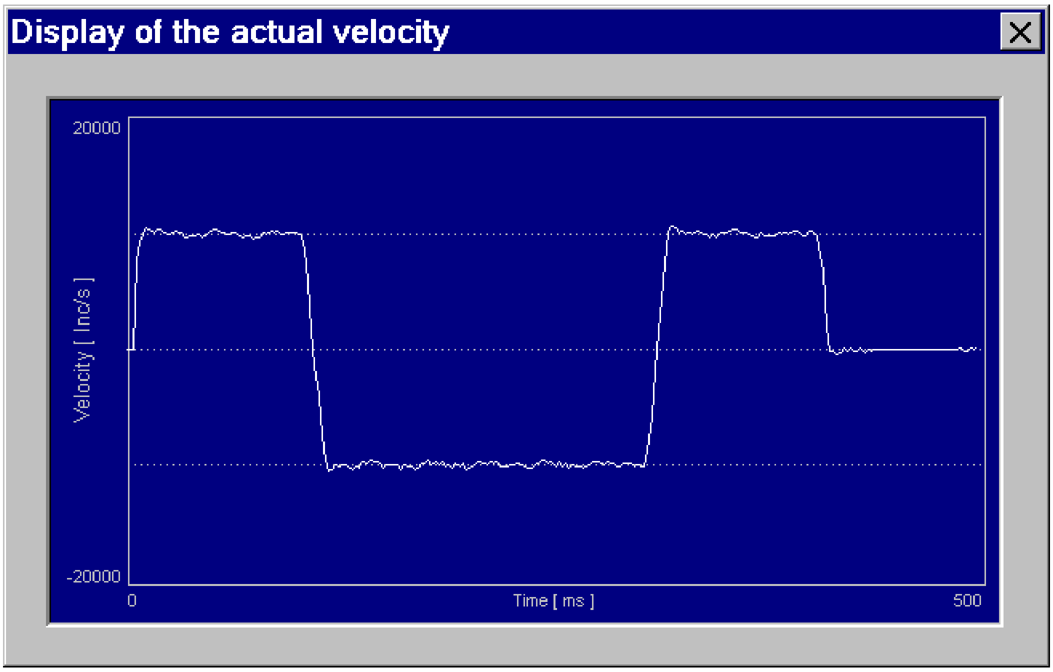
In order to be able to assess the settings, the display options step response and following error (control deviation) are available.

The setting of the regulation parameters can be realised as follows:

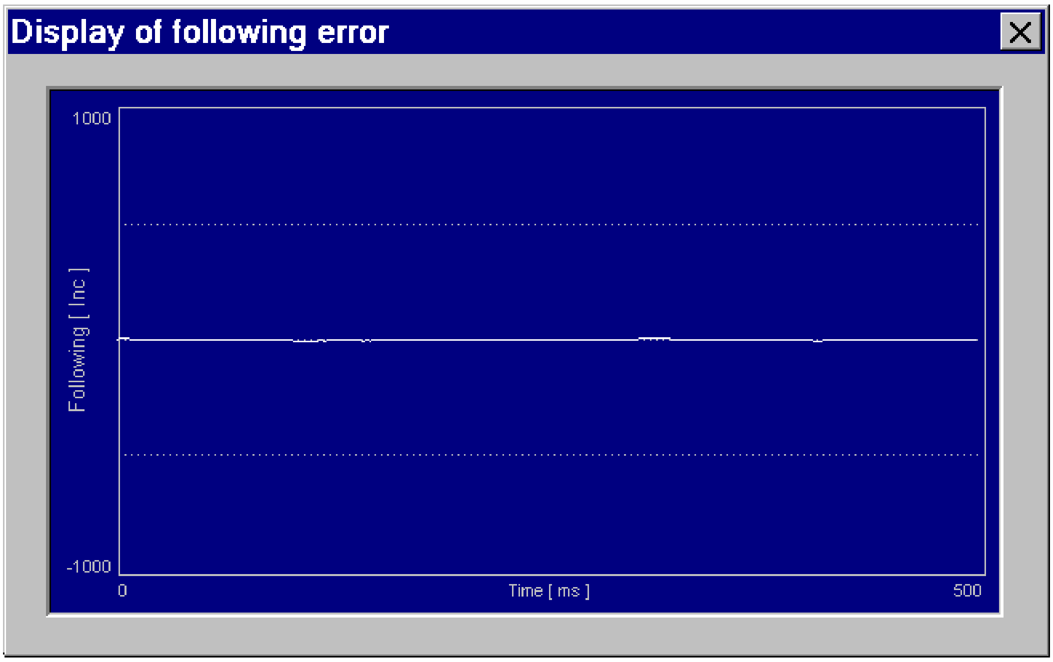
- Load default values.
- Set test acceleration to maximum acceleration and control the course of speed.
- Increase kp until a considerable overshooting can be noticed.
- Increase kd (and / or td) and adapt kp, if necessary, in order to smoothen the course of speed. Test different target speeds. Normally, td should be set equal to 0. In case of severe disturbs such as a toothed belt feed forward, it is sometimes reasonable to increase td instead of kd in order to achieve a high stability at a low noise level. It has to be taken into account that an increase of td by 1 leads to a duplication of kd.
- Set the display to following and change the feedforward factor(and possibly also kp), until the following error becomes minimal. For this, increase the target speed.
- Set the test acceleration to jump and control the transient behaviour of the control cycle even at higher target speeds (speed display). If the axis switches to a permanent vibration, limit the following limitation of the velocity controller.

Set the test acceleration to jump and control the transient behaviour of the control cycle even at higher target speeds (speed display). If the axis switches to a permanent vibration, limit the following limitation of the velocity controller.

It can be useful for the optimisation to once again vary the velocity controller in order to achieve a better course of speed and a smoother motor operation. During the commissioning in a random order, however, one has to make sure that the parameters of the velocity controller have to be stored with <Apply> before every switch-over to the position controller.



Display of the actual velocity (speed)



Display of the following error

The speed amplification factor Kv kv is a characteristic parameter for the dynamism of an axis. This parameter is not used within the positioning module. A superordinate CNC control such as the ISEL control can read this parameter and use it for the optimisation of the sequence of motions. The Kv factor is to be determined experimentally by selecting the “kv_value” radio button.

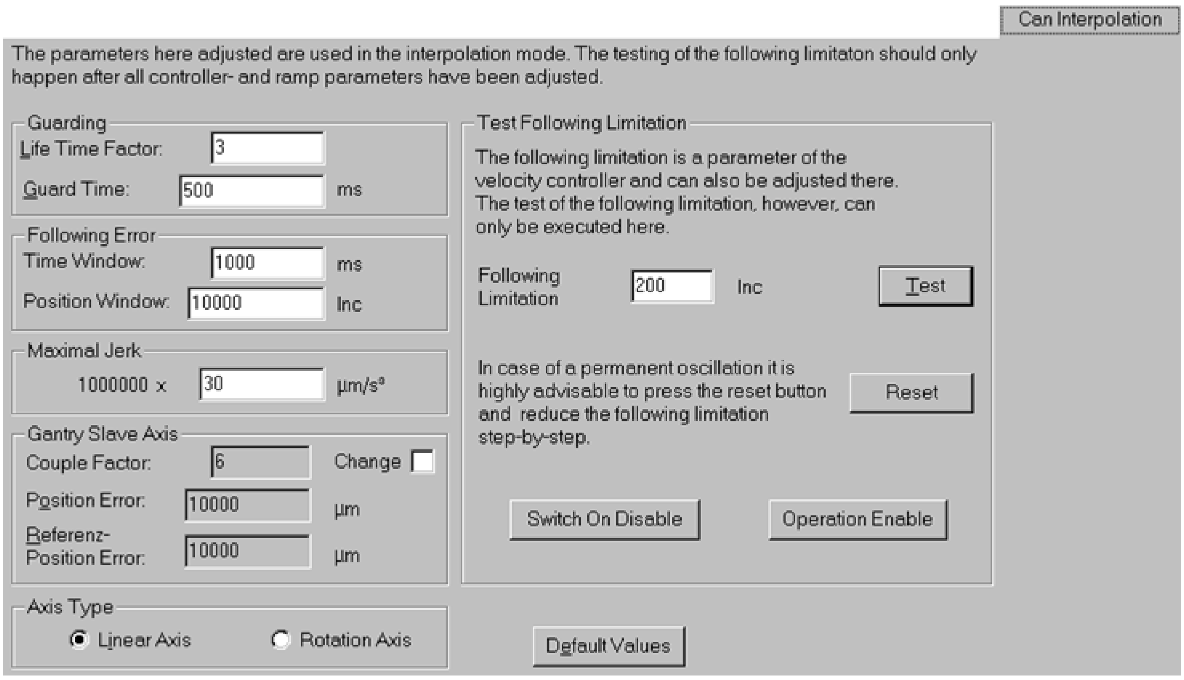
When testing the position controller, the controller gets a step function as an input signal. The acceleration has a major influence on the step response. Too great an acceleration value usually leads to a large overshoot, which is often the case with the option 'Step' in the field 'Acceleration'. If a small overshoot is required, you still have the option "Maximum acceleration". Here, you can test whether or not the set maximum acceleration is good enough. If necessary, you can still change the acceleration with the option "Manual", to find an optimum value. Using the button "Apply", the currently measured acceleration becomes the new maximum acceleration.

Note: During the controller optimising, with ACSetup only the step response of one of the three regulators can be shown. When you want to show the step response of current, speed and position controppller simultaneously, you can start the program ACSetup several times parallelly. At each instance of ACSetup you can optimise a controppler. The oscillograms of all controppllers remain on the screen. In this way you can better evaluate the mutual influences of the controppllers. However, the start of several ACSetup-instances is only possible in case of the CAN-connection. In case of the RS232-connection, it is logical that only one instance is possible. Since modified regulator parameters are only temporarily valid for the time of the step response registration, before each changing of one to another instance the modified regulator parameters must be saved by means of the button “apply”.

If the speed step response or following error is taken up again, normally the old speed display is deleted and the new speed actual value curve is displayed. If you wish to maintain old actual value curves, to better assess the effects of parameter changes on the position controller, you can activate the switch "Multiple" in the field "Oscilloscope". The speed curves can then be seen successively in four colours white, yellow, red, green. The complete display can be deleted at any time with the "Delete" button.

(See “Position controller parameters” on page 118).

Can Interpolation dialogue box



The parameters set here were designed for the modes of operation “Position controller with ramp profile” and “Position controller for interpolation” (see “Mode of Operation dialogue box” on page 58).

The two parameters “Life Time Factor“ and “Guard Time“ can be used to activate the watchdog function of the positioning module (see “Monitoring – Guarding” on page 99, “Guard time” on page

102 and “Life time factor” on page 102). If the value of one the two parameters equals 0, the watchdog function is switched off.

The monitoring of the following error is carried out by means of the two parameters “Time window” and “Position window”. If the actual position for the “Time window” time deviates from the target value by the “Position window” amount, an internal flag is set. A superordinate control can query this flag and react accordingly.

The axis’ vibration behaviour can be influenced by the “Maximum jerk” parameter. The smaller the jerk value, the less the vibration of the axis during the accelerating and braking processes. In return, the acceleration and braking process takes longer accordingly. The jerk value can be read in and used by the CNC control. Within the final power stage, this parameter is not used. The CNC control of ISEL uses this parameter.

Concerning the axis type, the axis can either be set as a linear or as rotary axis. The CNC control of ISEL uses this parameter for the display. Otherwise, the axis type is not used.

In case of a gantry axis, where two axes are to be moved synchronously, the maximum position error can be freely defined. This limit value is monitored only by the slave axis. The master axis does not take into account this limit value (see “Synchronous control - Gantry Axis” on page 135). If the deviation of position between the master axis and the slave axis exceeds this value, the slave axis is immediately stopped with the error code 22 (see “Fault states” on page 45). The related master axis is informed of this error via the CAN bus and stops immediately with the error code 28.

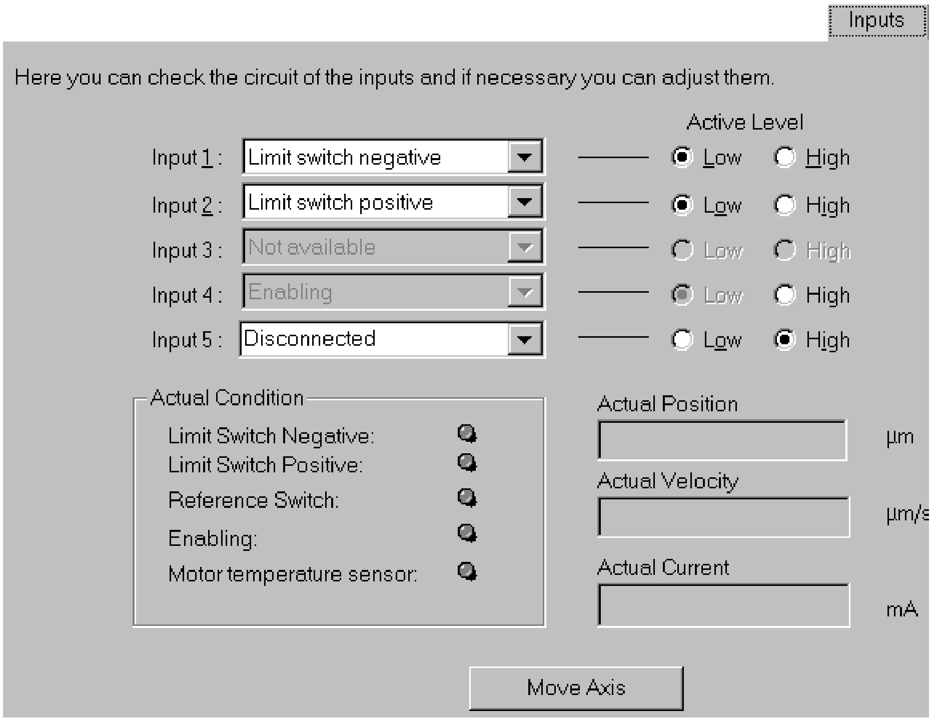
During the reference run, especially immediately after the switch-on, where the master and the slave axis are still not aligned to each other, it is often necessary to tolerate a larger position error between the master and the slave axis. You therefore have the option of defining a different position error for the reference run.

In case of a gantry axis’ slave axis, the user also has the possibility to define the coupling factor. The greater the factor, the “closer” the slave axis is connected with the master axis. Thus, the deviation of position between the two axes is also smaller during the gantry operation. This means: the greater the coupling factor, the better the gantry operation. But the regulation on the slave axis is unfortunately “rougher” The user should always apply the default value as initial value and change it by steps of 1 upwards or downwards. In the declaration mode, the user can observe the actual and maximum deviation of position between the master and the slave axis by means of the “Settings\Object Directory” menu.

Too large an following limitation value can lead to a permanent vibration during interpolation. Too small a value restricts the motion speed of the axis. It is recommendable to test the set value. After the “Test” button has been activated, the test runs automatically. A potential permanent vibration of the axis can be interrupted by the “Reset” button. An optimum value is the maximum possible value at which the permanent vibration was not yet noticed. The value of the following limitation can be changed via the “Following limitation” data entry field

(See “Velocity Controller dialogue box” on page 65)

Inputs dialogue box



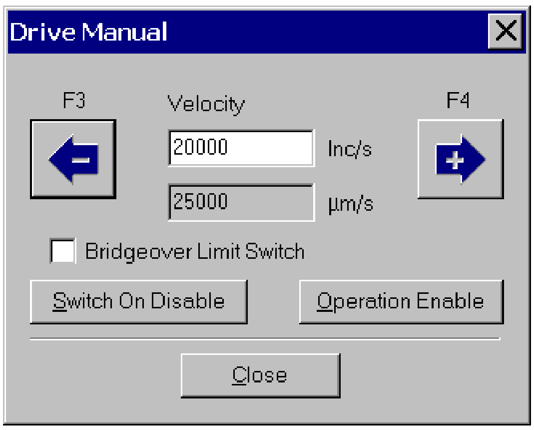
The positioning module disposes of three digital inputs for two limit switches, an enabling signaland as signal for the motor temperature sensor. The allocation of the signals to the inputs can be made at random, apart from the enabling signal which is always to be led to input 4 (hardware connection of input 4 with final stage release).

Input 5 is meant for the motor temperature sensor. If the sensor is not connected, it must be configured as “not connected” to avoid an incorrect error message. Naturally it is also possible to use this input for other aims. In that case the input must be configured as “free use”. A typical example is the use of this input for the fast motion stop (see Fast stop on inputs on page 149).

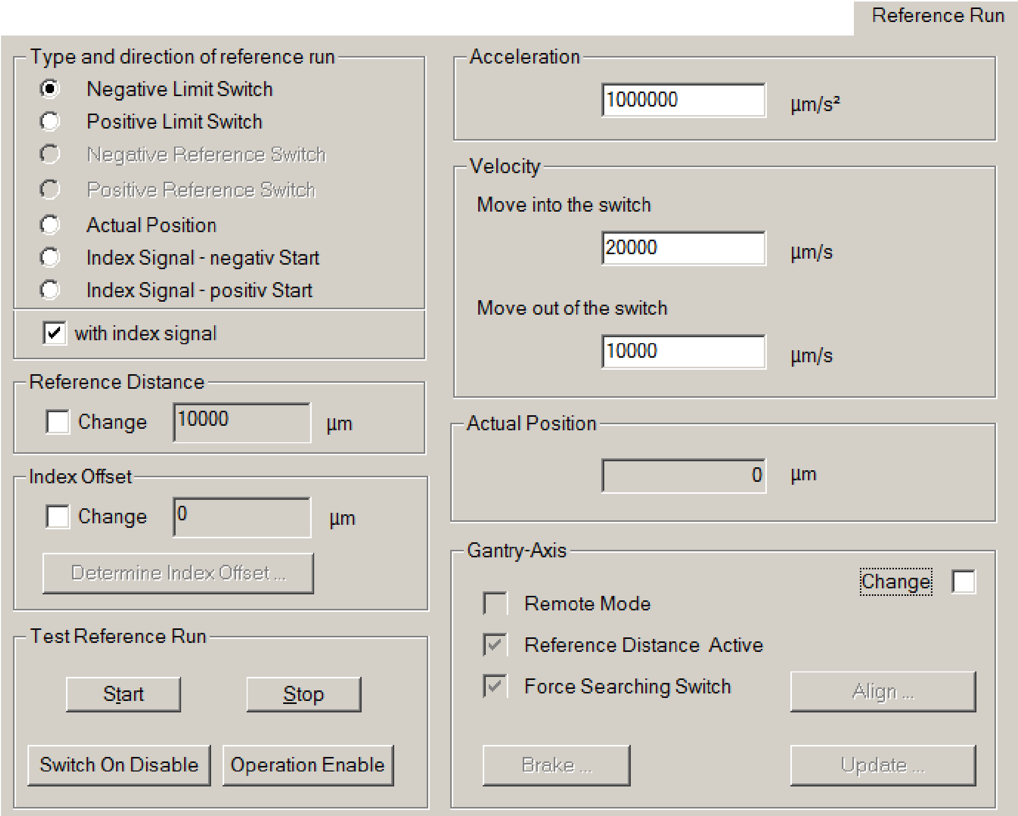
The actual status of the inputs can be read via the LEDs. This display corresponds to the state as it is evaluated by the control (light green = input is active, grey = input is not active).

(See “Drive Data” on page 129 and “Digital inputs” on page 133).

In order to test the limit switch on one axis, said axis can be displaced to the limit switches by means of a Teach dialogue (move axis). If the activation of the limit switches via the emergency stop chain of the plant would lead to the switch-off of the final stages, the option <Bridgeover limit switch> can be used to set an output of the final stage that should be used to separate the limit switch of this axis from the emergency stop chain.



Reference Run dialogue box – Homing dialogue box



The reference run (in other words: homing) serves to define the zero point of an axis. For this, the axis moves with the “Move into the switch” speed in one direction, until the switch indicated becomes active. Afterwards, the axis (slowly) moves with the “Move out of the switch” speed in the other direction, until the switch switches off again. Finally, the axis is brought to a defined distance (reference distance - in other words home offset) from the switching point. This is then the zero point of the axis. The reference acceleration should amount to about 1/10 of the axis acceleration (see “Acceleration dialogue box” on page 69) in order to go easy on the mechanical axis during the reference run. In case of a reference run with index signal first the axis moves after switch off to the index offset without considering the index signal. After this the axis moves as long as it recognizes the index signal. After recognition of the index signal the axis moves to the reference distance and then sets the zero point of the axis.

If the index signal is in the range where the switch bounces, it may happen that the index signal will recognize at next revolution. With the index offset one wants to prevent that the index signal and the switching point of the switch are too close to each other. To determine the index offset first you have to drive a reference run without index signal and then a reference run with index signal (Dialog "Determine Index Offset" over button "Determine Index Offset..."). After completion of the both reference runs the index offset is determined by the positions the reference runs have reached. For this the difference position of the two reference runs is calculated. If the difference position is between 1/4 of the encoder resolution and 3/4 of the encoder resolution the index offset is set to zero. If the difference position is out of this range the index offset is set to 1/2 encoder resolution. With "Assume Index Offset" you can assume the set respectivly changed index offset into the dialog "Reference Run". If you operate a gantry axis in "Remote Mode", you have another method for determining the index offset (see “Commissioning a gantry axis in the remote mode” on page 81).

The dialogue box comprises all parameters influencing the execution of the reference run. By means of <Type and direction of the reference run>, you define e.g. which switch is used and in which direction the axis is to go. In order to achieve a high precision of the machine’s zero point, a combination with the index signal is possible, provided the index signal is connected to the encoder. If an index signal is present, you should generally prefer a reference run method to an index signal.

Higher repeatability of the reference point is achieved here. You can also use higher reference run speeds here, because the repeatability does not depend on the speeds. This means that the reference run is completed faster. Likewise, the speeds and the reference distance can be set here.

Furthermore, there is the possibility to test the reference run. You have to take into consideration that IMD20/IMD40 do not support a reference switch, that means the user must define a limit switch instead of a reference switch. Internally, the module is switched to the Homing Mode – Reference run (a CanOpen operation mode) for the execution of the reference run; then the reference run is started and, after its termination, the previous mode of operation is switched back to.

In case of a reference run on one of the two limit switches, note that, during the reference run, the output for the shunting of the limit switch (safety circuit) is set in order to prevent the final stages from switching off, when the limit switch is activated.

If a limit switch is active, you must move the axis out of the limit switch in order to test the reference run. In order to move out the axis from the limit switch, you can use for example the dialog box “Inputs” (see section “Inputs dialogue box” on page 75).

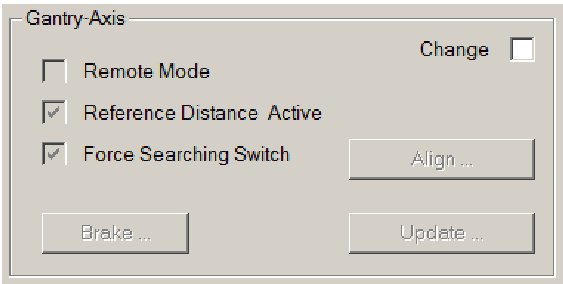
To prevent endless movements in the case of defective reference switches, you can set the two objects "Homing Switch Search Path (udu)" (object 0x6510 - subindex 0x16) and "Homing Switch Leave Path (udu)" (object 0x6510 - subindex 0x17) (see section “Path limitation of homing” on page 122 and section “Drive Data” on page 129)

The "Gantry-Axis" area is intended for the configuration and commissioning of a gantry axis. For a single axis, you should not make any further changes here. The next picture shows the default settings for a single axis.

Commissioning of reference run of a gantry axis

For each axis (master and slave) of a gantry axis you have to set the same parameters as for a single axis. In principle, the motion parameters and the reference methods of the two axes can be different in value. The parameters, which must be identical, can be found in the section „Particularities as regards the commissioning of a gantry axis“ on page 38. The cnc control calculates the optimum movement behaviour for both axes from the specified values. Particularly important are the reference distances, which can be used to compensate for the position error of the two axes at the zero point, i.e. to ensure the parallelism of the two axes.

The "Gantry-Axis" area is intended for the configuration and starting up of a gantry axis. If you only have one single axis, you should not make any further changes here. The picture shows the default settings for a single axis. This setting also applies to a gantry axis in standard mode (see option "Remote Mode").



Option ”Remote Mode“:

Here you can choose between two reference run modes of a gantry axis. If this option is not active, you have the standard mode that you also have for a single axis. In standard mode, each axis (master and slave) performs its reference run independently. Apart from monitoring the position error by the higher-level controller, there is no interaction between the two axes. Each axis executes the various phases of the reference run, such as "Move into the switch", "Move out of the switch", "Searching for the index signal",... for itself. The parallelism of the switches and the encoder index signals is imperative so that the two axes do not stand at an angle to one another during the reference run movement. A certain variance from absolute parallelism is tolerable due to the flexibility of the mechanics. But too much obliqueness then leads to overcurrent errors in the axes. Manual and time-consuming alignment of the limit switches and the encoder index signals is generally necessary. And the manual alignment of the limit switches also very often requires the use of external switches, which is also a cost factor. And in many cases, such as gantry axes from rotary axes, the assembly of additional switches is not easy and sometimes not possible. If you want to operate the gantry axis in standard mode, you must not change the standard settings here. You don't have anything more to do in this area either. Your effort will focus more on manually aligning the switches, the index signals, determining the reference distances for the purpose of axis parallelism, ... The procedure for this is explained in detail in the section “Commissioning a gantry axis in standard mode” on page 79.

If the option "Remote Mode" is activated, you select the so-called remote mode of reference run for the gantry axis. In this mode, a higher-level controller coordinates and synchronises the various reference run phases on the two axes. The reference runs on both axes are no longer completely independent of each other as in standard mode. The remote mode no longer requires the parallelism of the switches and the index signals. The switches or the index signals can be located anywhere and at any distance from each other. External switches do not necessarily have to be present. Standard internal micro-switches or only index signals alone are sufficient. This mode must be selected for both the master and slave axes. In "Remote Mode", the determination of reference distance, index offset,... is largely automated. The procedure for this is explained in detail in the section “Commissioning a gantry axis in the remote mode” on page 81.

Here are some practical tips for using the "remote mode" option of the reference run

- The range of motion of an axis is determined by the negative and positive position switches. If the movement ranges of the two axes of a gantry axis are too offset, there is a problem here with defining the movement range for the gantry axis. With a gantry axis consisting of two rotary axes, it is actually always the case. Because the two axes of a gantry axis do not have to have the same setting in "Type and direction of reference run" (see Section “Reference Run dialogue box – Homing dialogue box” on page 76), you can configure the axes as follows. On one axis, their two limit switches serve to limit the range of motion. One of them can serve as a reference switch at the same time. On the other axis, only one switch is required, which is configured as a reference switch. This switch can be located anywhere in the movement space. It no longer needs to be aligned with the switches of the other axes.
- If possible, the reference run with index signal should always be preferred. With the help of the index signal, the software can determine the reference distances and reference offsets very precisely in order to achieve a high degree of parallelism. The two axes no longer work against each other. And no cracking can be heard during movement.

Option “Reference Distance Active“:

The reference distances are there to correct obliqueness in the limit position switches or the encoder index signals. After the reference distances have been approached, the two axes are then parallel. For normal operation, this option must be active for both axes. It applies to both the standard mode and the remote mode of the reference run. This option remains deactivated during commisioning of the gantry axis in remote mode, where the reference distances are still unknown. During a reference run during the start up phase, the reference distances are then determined automatically.

Option “Force Searching Switch“

In the standard reference run mode, this option must be active for both axes. In remote mode, the switches or index signals no longer have to be parallel. This results in two situations.

a. The switches of the two axes are very far apart

   Here it can happen that the motors lie between the two switches before the reference run. This option should be selected for the axis whose switch is in the rear reference run approach movement direction. The option is not selected for the other axis. During reference run, the two motors move up to the switch at the rear. Only this switch is queried. When this switch is activated, the movement reverses. In the drive-out phase, the switches of both axes are first queried.

b. The switches of the two axes are close to each other.

- Both axes are very rigidly connected.

  The flexibility of the mechanics is very low. This option should only be active for the axis whose switch is at the back (see point a.).
- Both axes are connected quite elastically.

  This option should be activated for both axes.

During scommissioning, where you still want to determine the reference distances, this option must be active for both axes. In addition, you must ensure that each axis is in front of its switch before the reference run in the commisioning phase.

Commissioning a gantry axis in standard mode

In this mode, the control executes the reference run in 2 steps.

- In the first stage, the controller converts a reference run method with index signal on both axes into the equivalent reference run method without index signal. For example, reference run method 1 (negative limit switch with index signal) is converted to method 17 (negative limit switch) (see Section “Homing Mode – Reference run” on page 120). A reference run to index signal only (methods 32 and 33) is not changed. All reference distances are set to zero. Then the reference movement is executed on both axes. After the reference run, the two axes are then directly at their switches.
- In the second step, the original reference methods and the original reference distances are reset. After executing this reference run, the two axes are then in the final zero point.

As already mentioned above, each axis (master and slave) executes the reference run separately. Apart from the error and position error monitoring, there is no further information exchange between the two axes (see section “Commissioning of reference run of a gantry axis” on page 77). This procedure can only be successful in the first stage if the switches are parallel to each other. Otherwise the axes would be at an angle to each other at the end and lead to overcurrent errors. A reference run to the index signal in the second stage also requires the parallelism of the index signals, because the axis moves at different speeds when searching for the index signal and then when leaving the index signal after its occurrence. If the index signals are oblique to each other, they also occur at different times. The different travel speeds of the axes after the index signal on an axis causes the axes to no longer move synchronously. Overcurrent errors are the consequences.

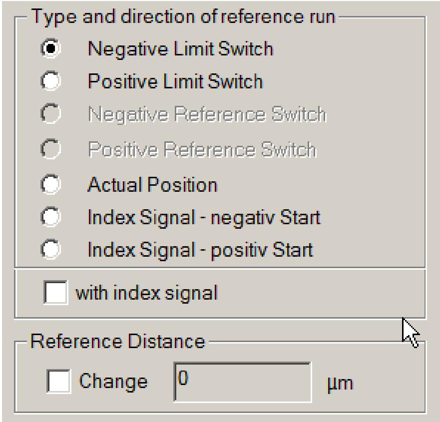
The following steps explain how you can align the axes in practice to achieve the necessary parallelism

1. The individual axes can be put into operation with AC Setup as usual. All settings for controller, brake,... are ready. For the reference run, all parameters except the reference distance and the index offset must already be set. The two axes, which are already mounted on the machine frame, must not yet be firmly connected to each other.

2. Mark one point as the reference point on the master axis and one point on the slave axis.
   - For a linear gantry axis, the reference points must lie on a straight line perpendicular to the direction of movement of the gantry axis. In most cases, you can set the end of an axis as the reference point.
   - With a rotary gantry axis, you can mount the traverse, which then aligns the two rotary axes. You define a point as the reference point on each rotary axis. It does not matter where the reference points are. All they have to do is face each other. The traverse must then be removed again in order to be able to carry out the reference runs later in steps 4 and 5.

3. Two instances of ACSetup are now started, one for the master and the other for the slave axis. You can then carry out the reference run of the first step mentioned above in the tab "Reference

Run" (ACSetup\Commissioning\Rundom Oder). All you have to do is uncheck the "With index signal" checkbox. In the case of a reference run without index signal (reference run methods 17, 18, 19 and 21), you do not need to continue. For a reference run to index signal only (reference run methods 32 and 33), continue with step 5. Do not forget to set the reference distance to zero.



4. Carry out the reference run on both axes. The two axes are not yet firmly connected. Therefore a reference run is possible without further ado. After the reference run, the two axes are then directly at the respective switch. For each axis, you can now measure the distance between the switch and the reference point (see step 2). Using the two distances, you can now move a switch so that the distances between the switch and the reference point are equal. For a reference run only to switch, you can continue with the step 6. Otherwise, please mark the position of the two switches. You then need these positions in step 5 to set the index offset. Then check the checkbox "with index signal“.

5. Carry out the reference run of both axes. After the reference run, the two axes are directly at the respective index signal. Now you have to release the shaft couplings on both axles. With a vertical gantry axis, you probably need to support the axes to prevent them from sagging. Using the distances between the reference points and the index signals, you can then move the axes (not the motors) by hand to achieve equal distances between reference point and index signal. The motors are still powered. Therefore, they stick to the positions of the index signals. For a reference run to index signal only, retighten the shaft coupling and continue with step 6. When referencing the switch and index signal, you must also set the distance between the switch and the index signal. In step 4 you have already marked the positions of the switches. The switch-index signal - distance should be about half the movement length of a motor revolution. The reason why about half of a motor revolution should be, can be read in the section... on page.... You can easily calculate the movement length of a motor revolution with the data from the "Motor and Gearbox" tab (ACSetup/Commissioning/Rundom Order). Now you can retighten the couplings. Do not forget to set the index offset in both axes to zero. Determining the index offset of an axis with the button "Determine Index Offset ..." is no longer necessary or useful here.

6. The switches and the index signals are now parallel to each other. You can still set the same reference distances for both axes. The set-up of the gantry axis is roughly finished. The next step explains how you can make a fine adjustment later, when the system is already assembled.

7. Your system is now fully assembled. You execute the reference run of the gantry axis. You can do this with the program CANSet.
   - Open the initialization file for the System (CANSet\File\Open)
   - Initialize the system (CANSet\Test CNC Control\Reset)
   - Carry out the reference run (CANSet\Test CNC Control\Reference Run)

Exit CANSet immediately after the reference run. All axes then become currentless. Start two instances of AC Setup, one for the master axis and the other for the slave axis. You can see the current position of the axis on the tab "Reference Run" (ACSetup\Commissioning\Rundom Ordner). A possible distortion between master and slave means that the positions of the two de-energised axes are not the same. By means of the position difference you can correct the reference distances accordingly in order to reduce any distortion between master and slave.

Remember that a tension-free master-slave connection and the perpendicularity of the axes are two pairs of shoes. If your mechanics are not optimally constructed, the correction of squareness can lead to more tension between the axes and vice versa.

Commissioning a gantry axis in the remote mode

The time-consuming procedure in standard mode to achieve the reference distances for the parallelism of the axes after the reference run is completely automated in remote mode during the test run of the reference run. In addition, the index offset of both axes is determined in the same way. This means that you no longer need to operate the button "Determine Index Offset ..." button separately in "Remote mode".

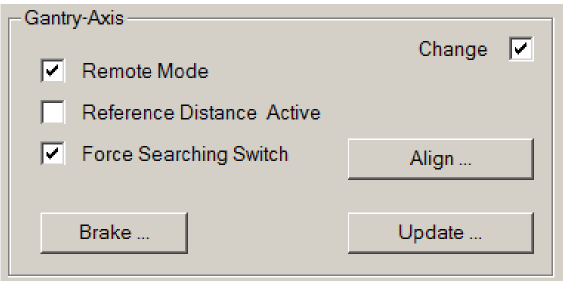
Follow the steps below to start up a gantry axis in "Remote Mode". Please note that the starting up of a vertical gantry axis is a little more complex than is the case with a horizontal gantry axis, because a vertical axis can sink during start-up due to its own weight. To prevent this, we need the brakes installed in the respective axles. The additional steps for brake control are in sections with italic fonts. For a horizontal gantry axis, you can simply skip these steps.

1. The individual axes can be put into operation with AC Setup as usual. All settings for controller, brake,... are ready. For reference run, all parameters except the reference distance and the index offset must already be set. The two axes, which are mounted on the machine frame, can already be firmly connected to each other.

2. The two axes must then be aligned parallel to each other, e.g. with the aid of dial gauges or with angles. To move the axes during alignment, you can use the dialogue window "Drive Manual", for example, which is called up via the button "Move Axis" in the tab "Inputs" of ACSetup (ACSetup\Copmmissioning\rundom Ordner). Please note that you can run several instances of ACSetup simultaneously via the CAN bus. The parallelism set here is the measure of all things. During the test run of the reference run, reference distances are determined in such a way that this parallelism is achieved again later during operation after each reference run. If the connection between the two axes is very rigid, manual alignment is no longer necessary. Please make sure that both axes are in front of their respective switches before the reference run test run and that there is no limit switch error.

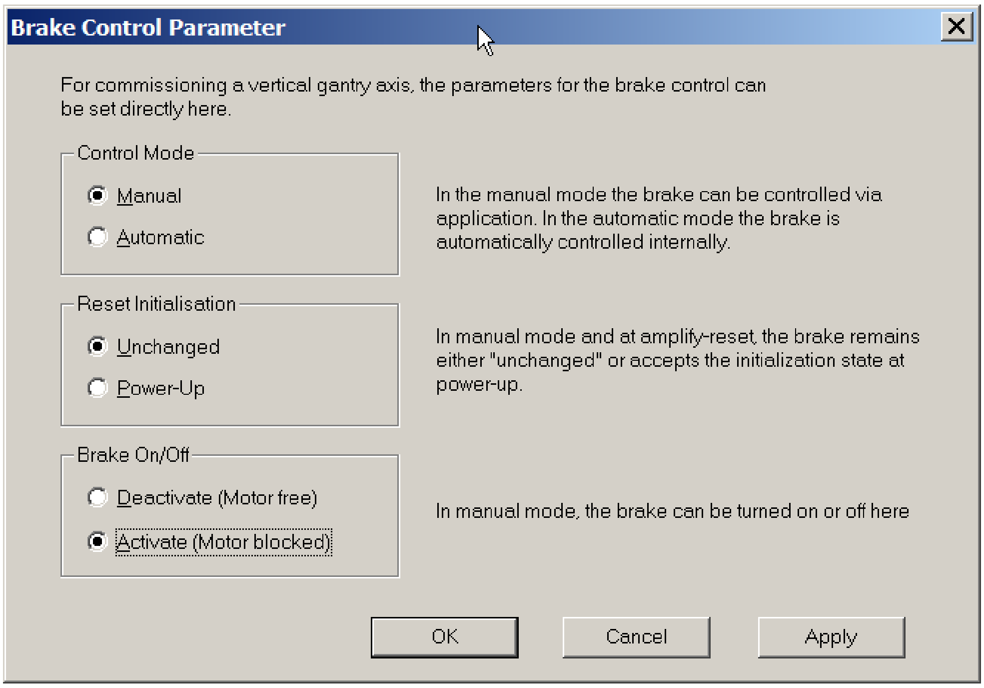
3. On both ACSetup instances, select the tab "Reference Run", preferably via "Random Order" (ACSetup\Commissioning\Random Order"), because you have the greatest flexibility here.

4. For the test run of the reference run, you must set the following options for both axes.



5. *You can skip this point for a horizontal gantry axis. With a vertical gantry axis, there is a danger that the axes may fall during the initialisation phase due to their own weight, which then nullifies*

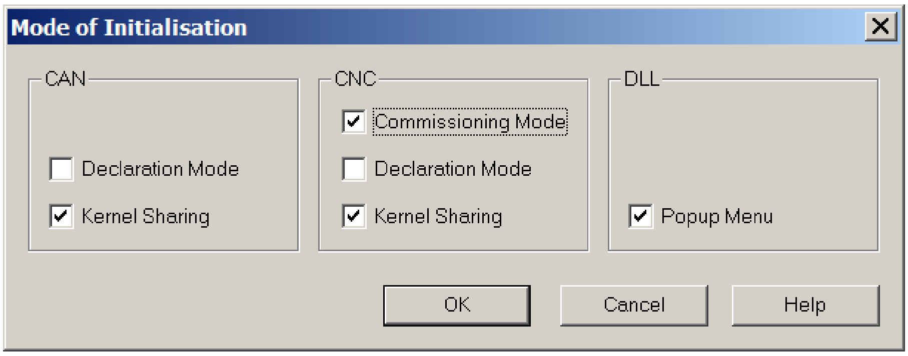
the set parallelism. To prevent this, you must set the brakes of both axes to manual mode to be able to control the brakes manually during the test run. Please do not forget to note down the original setting so that you can reset it later. Apply the settings as shown in the picture.



With "OK" or "Apply" the axis is then held by the brake. You should note that no limit switch may be active at the held position.

6. Accept the settings and exit the setup dialogue. Close the CAN driver in the dialogue "CAN Settings" menu (ACSetup\Connection\CAN-Settings). The connection of ACSetup to the CAN bus is disconnected. However, you do not have to end the two instances of ACSetup. You will need them back later.

7. The programme ACSetup is a commissioning software and works axis-oriented. The reference run of an axis can thus be executed. However, the reference run of a gantry axis consisting of two axes cannot be made here. For this you can use the programme CANSet. Start CANSet and open the initialisation file of the system. Then activate the commissioning mode of the cnc-control in the dialogue “Mode of Initialisation” (CANSet\Configure CNC Control\Mode of Initialisation).



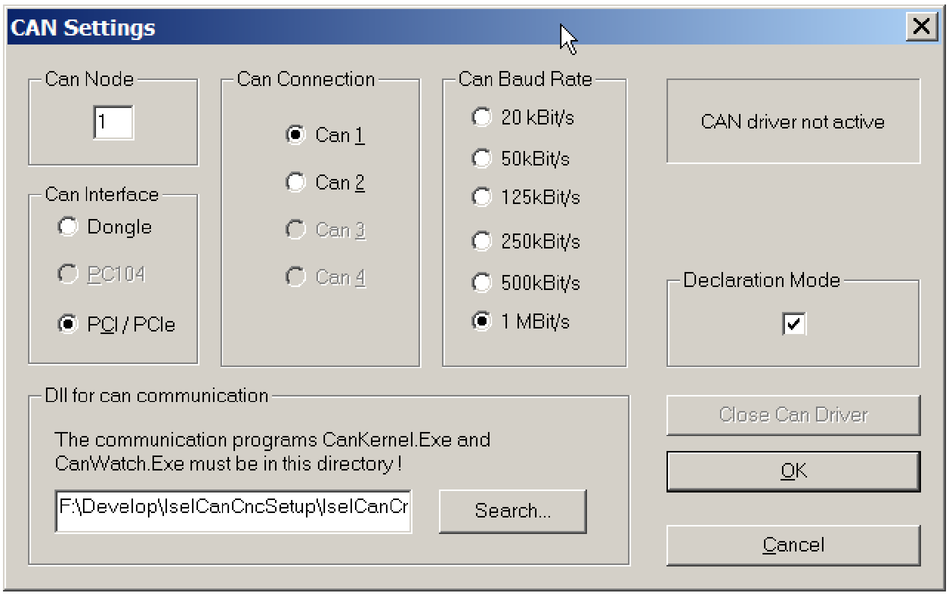
Please note: This mode must not be active in normal operation.

8. You can then initialise the system via the menu "Reset" (CANSet\Test CNC Control\Reset). During the initialisation phase, the output stages are reset. With a horizontal gantry axis, it does

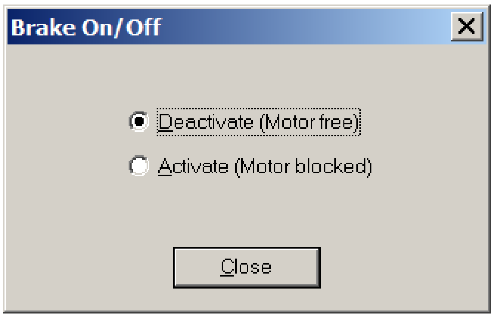
not happen any further. But a vertical gantry axis could sag here if its master and slave axes are not held by the brakes. It is therefore very important to activate the brake in advance in order to fix it on a vertical axis (see step 5).

9. *After a successful reset, the motors are now energised. With a vertical axis, you must now deactivate the brakes of the master and slave in order to be able to move the axes freely.*

   First, the running ACSetup instances are reconnected to the CAN bus. Connect ACSetup to the CAN driver via the menu "CAN-Settings" (ACSetup\Connection\CAN-Settings). Because the CAN bus is currently used by the CANSet, you must activate the declaration mode. Otherwise you cannot establish a connection to the CAN driver.



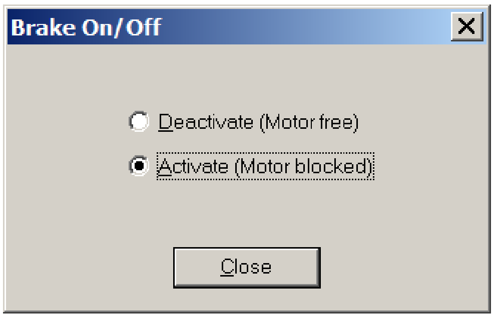
   Open the object directory (ACSetup\Settings\Object Directory) and then go to the object "60FE: Digital Outputs" (Object Directory\Device Profile Objects). The dialogue for switching the brake on/off is displayed by double-clicking on this object. Please note that this window only appears when the axis brake is in manual mode, which you have already done in step 5 and when the on-line mode is active.



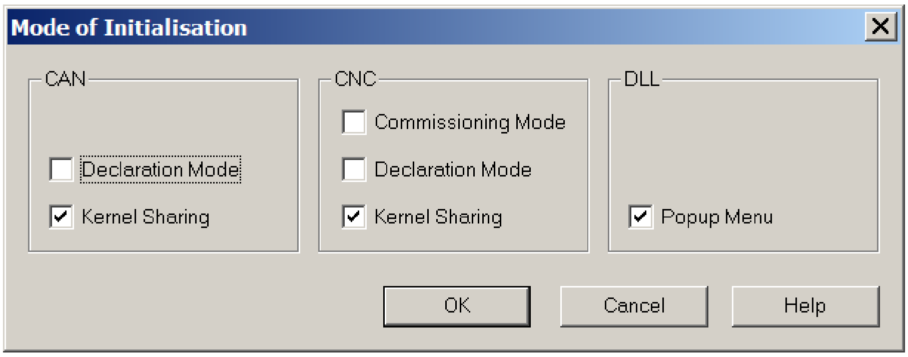
   After deactivating the brake, the motors are freely movable. Please do not forget to deactivate both the brake of the master and the brake of the slave. And if you have another vertical gantry axis, you should repeat the same procedure for the other gantry axis.

10. Use the menu "Reference Run" (CANSet\Test CNC Control\Reference Run) to execute the reference run on all gantry axes that you want to start up. During the reference run, the positions of the limit switches and the index signals are recorded and stored internally in the respective output stages. And as long as the power amplifiers are not switched off, the data remains here.

11. *When the reference run is complete, you should now reactivate the two brakes for a vertical gantry axis. You still have the dialogue for this open (see point 9). After that you have to close the CAN driver to disconnect ACSetup from the CAN bus. You can continue to run the AC Setup instance because you will need it later.*

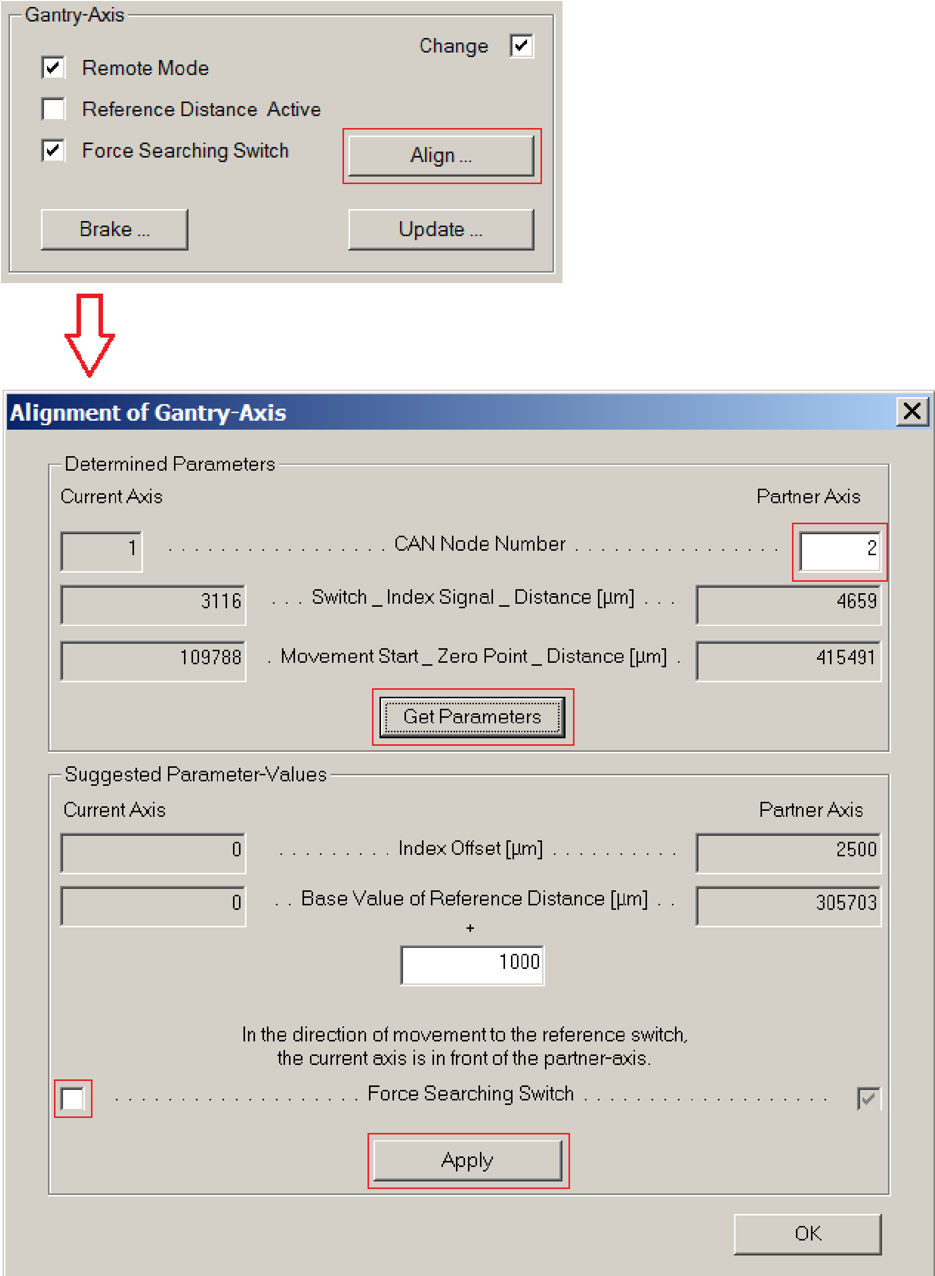


12. You must now disconnect the CAN-bus from the CANSet. The separation can be done either by creating a new initialisation file (CANSet\File\New) or by exiting CANSet. However, you must not forget to deactivate the commissioning mode and save the initialisation file before doing so.



When disconnecting from the CAN-bus, CANSet removes the cnc control from the PC. Therefore, all motors are de-energised. With a vertical axis, the axes cannot sag because you have already reactivated the brakes in step 11.

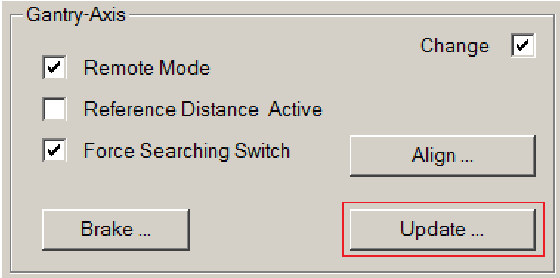
13. The ACSetup instances for the master and slave axes are still active. You must connect them to the CAN-bus via the menu "CAN Settings" (ACSetup\Connection\CAN-Settings). Please note not to activate the declaration mode for all ACSetup instances. The tabs “Reference Run” of the two axes must each be activated via "ACSetup\Commissioning\Random Order". With an axis, the dialogue “Alignment of Gantry-Axis” is called up via the "Align..." button.



Enter the CAN node number of the other axis in "Partner Axis". Press the "Get Parameters" button to get the stored data of the axes. This data is then used to calculate the base reference distances and the index offsets. The reference distances can still be varied as required.

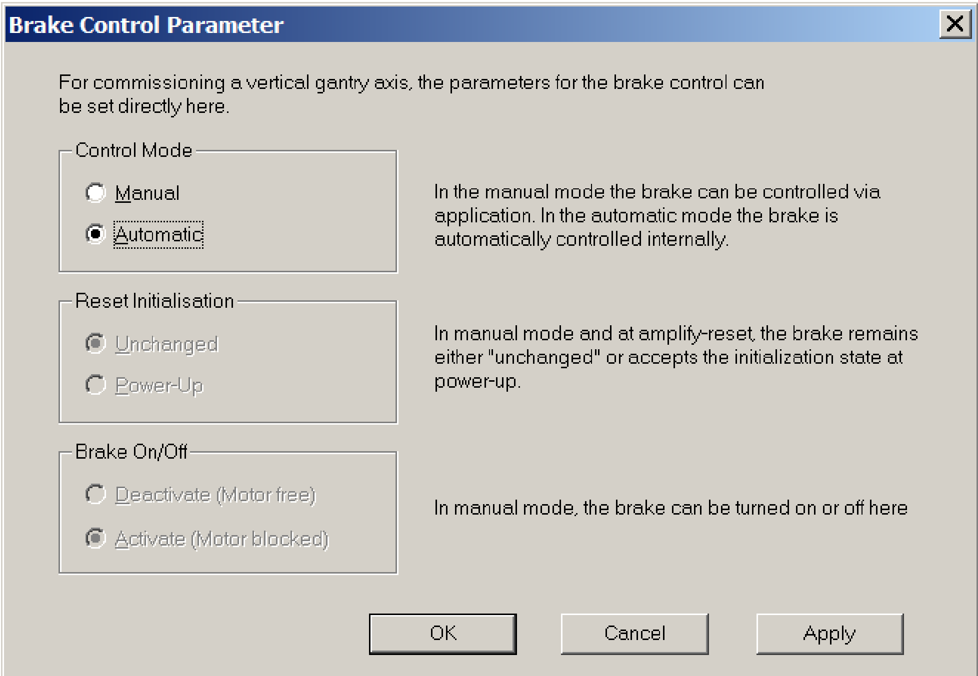
Depending on the mechanical design, the "Force Searching Switch" option can be activated for the axis whose switch is at the front. The base reference distance is the distance between the two switches and is therefore a useful aid for the decision. If you have a large base reference distance, you do not need to activate this option. The option is more likely to be made with a small base reference distance. For the axis with the switch at the rear, this option is automatically activated and can no longer be changed.

With the "Apply" button, the calculated parameters are not only transferred to the current axis but also to the partner axis. For the current axis, the parameter values only become visible after the alignment dialogue has ended. For the partner axis, the data is only displayed in ACSetup after the "Update" button has been clicked here. It is essential that the parameters on the partner axis are updated and checked. Otherwise the data will be lost.

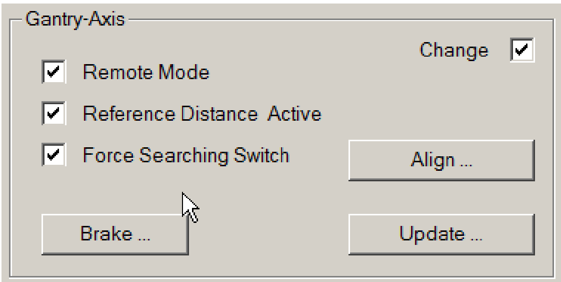


Note that determining of the index offset of an axis with the button "Determine Index Offset ..." is no longer necessary or useful.

14. In the case of a vertical gantry axis, you must use the "Brake..." button to open the "Brake setting" window in order to reset the brake settings that you changed in step 5 to the old values. In most cases you have the automatic mode for the brake as shown in the picture.



15. Activate the "Reference Distance Active" option for the two axes and accept the data.



16. If the option "Force Searching Switch" is not activated in both axes, it is recommended to set the two objects "Homing Switch Search Path (udu)" (Object 0x6510 – subindex 0x16) and "Homing Switch Leave Path (udu) "(Object 0x6510 - subindex 0x17) to prevent an endless movement in case of defective reference switchs (see section “Path limitation of homing” on page 122 and section “Drive Data” on page 129).

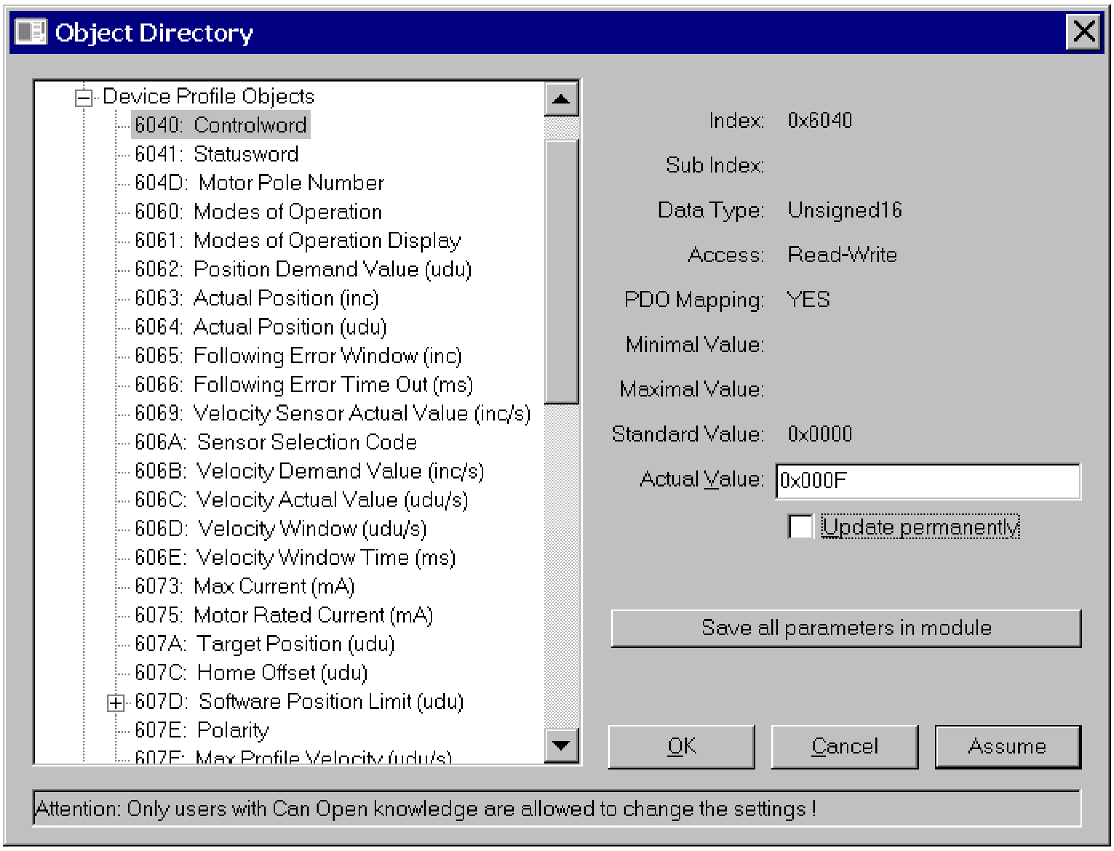
Object Directory dialogue box

The Object Directory dialogue box shows a tree view of all parameters (CanOpen objects) of the drive module. Via this structure, you can directly access all objects and read and / or write the parameter according to the respective characteristics. A description of all objects can be found in the chapter “Object Directory” on page 100. Only users experienced in CanOpen standards may use this dialogue box in order to set parameters. All other users shall realise the setting of parameters via the “Commissioning” dialogue box (see “Commands of the Commissioning menu” on page 48).

The tree view on the left side is subdivided into Communication (DS301), Device Profile (DS402) and Manufacturer-Specific Objects. On the right side, the characteristics and the content of the currently selected object are displayed. By means of the <Assume> button, you can realise changes in the current parameters. <OK> also confirms the current change and closes the dialogue box.

The “**Update permanently**” option has the effect that the value of the currently selected parameters is read about 5 times per second from the drive module (in case of an enabled online mode).

Via the “**Save all parameters in module**” button, you can permanently store the currently set values (set of parameters) in the drive module (the storing of individual parameters is not possible).



Firmware update

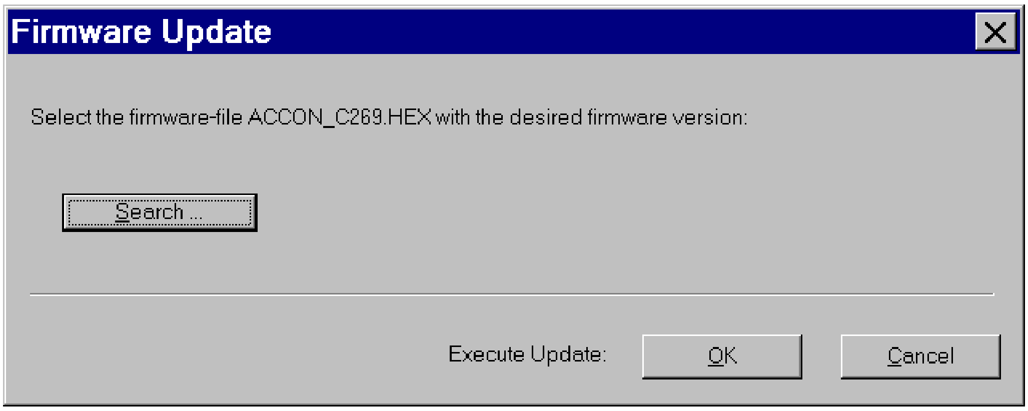
The software of the drive module is stored in a rewritable Flash memory. Thus, the programme can be loaded easily from the outside into the module, without memory modules having to be replaced. The loading of a new software version can only be realised via the serial interface.

Apart from the programme, the configurable set of parameters is also in the Flash memory. The memory sector is composed of four segments, the fourth segment being used to store the current parameters. A segment-wise deletion leads to the fact that only the first three memory banks are newly written during the normal update, i.e. the set of parameters is maintained. If new parameters are added due to the update, these are configured with default values.

Contrary to the normal update, the update via the bootstrap loader deletes the complete Flash memory and the stored parameter settings are lost. This possibility to load the programme also works, if no executable software exists in the module, e.g. after a failed normal update. To note, here it is only possible to work with a baud rate of 19200 baud

It is recommendable in any case to safe the set of parameters as a DCF file before carrying out an update.

The programme which is to be loaded into the module is read in from a hex file (e.g. ACCON_C269.HEX). This file must be selected via a corresponding dialogue box. Afterwards you can launch the update process.



CanOpen protocol

Overview

The communication in CanOpen networks is based upon CAN data packages the content (max. 8 bytes) and target address of which are used by the CanOpen protocol.

The main part of the communication is handled via the communications objects SDO (Service Data Object) and PDO (Process Data Object).

Each CanOpen participant has a reserve of variables and parameters that are arranged in an object directory with defined addresses and that can be read or written via the network.

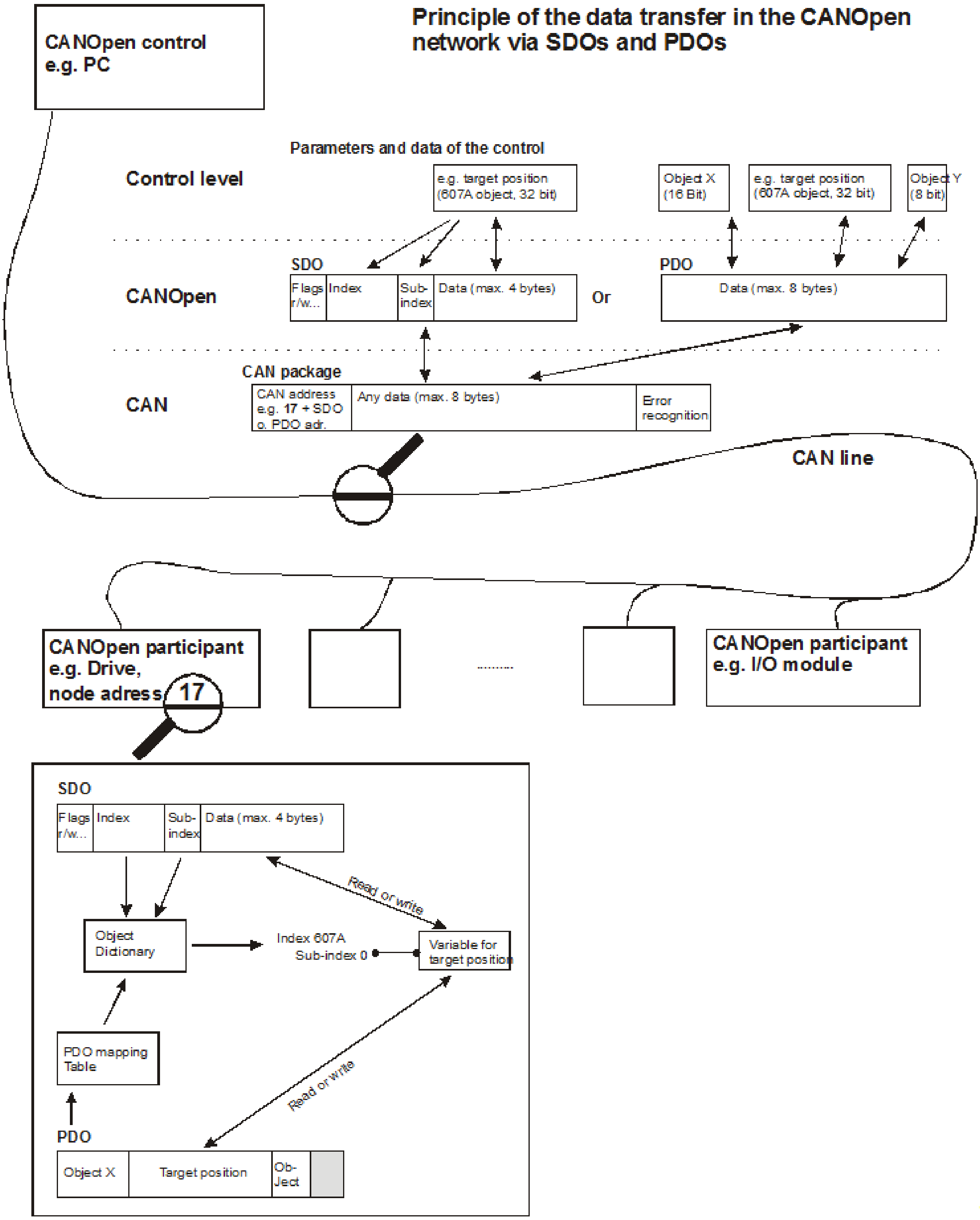
A CanOpen participant can be subdivided into three functional blocks:

Communication	Object directory	Application
Sending and receiving of communication objects. SDO, PDO, SYNC, etc.	Administration of communications parameters. Interface to internal variables and parameters.	Application programme, e.g. control of a drive, I/O administration. Implementation of the device profile.

Apart from the two objects for the data transfer, there are still further communications objects, e.g. for the synchronisation or error messages. All in all, the following CanOpen objects are supported:

Communications object	Short description
SDO	Universal communications channel for the reading and writing of all objects contained in an objects directory. Slower than PDO since the object address has always also to be transferred and since a feedback via a second SDO has to be realised.
PDO	Communications channel for the exchange of process data. Fast transfer, since the data are sent without protocol overhead.
EMCY	Emergency object for the transfer of error messages.
SYNC	The synchronisation object allows a synchronous operation of several bus participants.
NODE GUARDING	Monitoring of the bus participants through the exchange of cyclical messages.
NMT OBJECT	Control of the basic states of all bus participants.

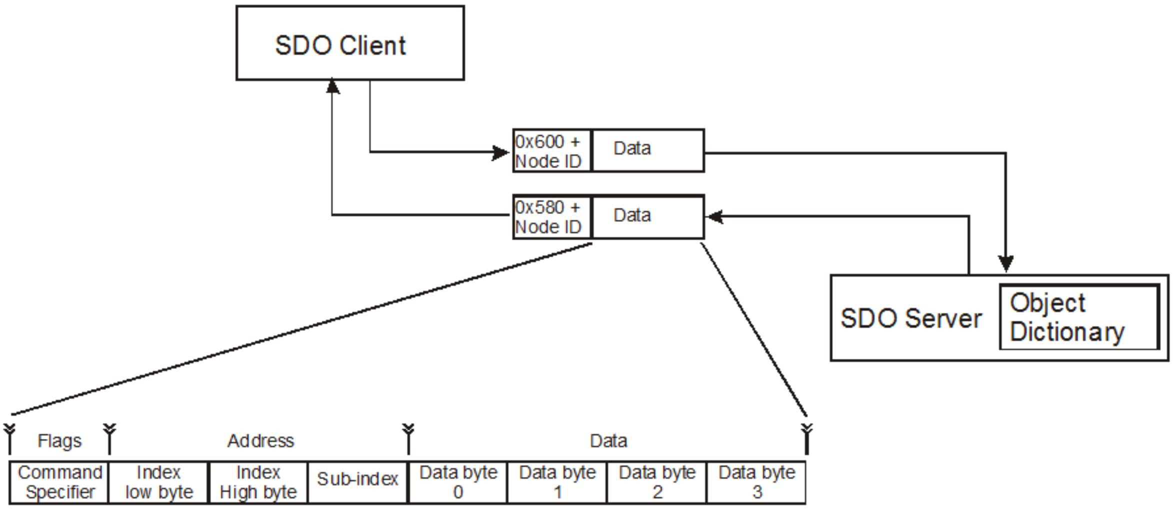
The following figure tries to display the communication with a CanOpen device via SDO or PDO.



SDO

The Service Data Object (SDO) makes it possible to access the object directory of a CanOpen device. An SDO always uses two CAN objects with different IDs, since this protocol is always confirmed. An SDO creates a communications channel between two CanOpen participants. The device the object directory of which is accessed is the server of this SDO. The COB-ID of the sent Object (0x600 + Node-ID) and the COB-ID of the returned object (0x580 + Node-ID) contain the node ID of the SDO server.

Each CanOpen device should dispose of a default SDO. The identifier of the Can objects related to this default SDO result from a defined Can address plus the node number (node ID) of the device.



The content of the first byte of an SDO (Command Specifier) controls the communication and defines e.g., if an object is read or written.

An SDO message always consists of 8 bytes, irrespective of the fact, how many data bytes are really transferred. The unused data bytes can contain any values and have to be ignored. The number of data bytes can be given in the Command Specifier; it is, however, also defined by the data type of the object to be transferred.

Basically, there are two different types of an SDO transfer:

- parameters with a length of 1-4 bytes are transferred with an SDO telegram (Expedited Transfer).
- data with a length of more than 4 bytes are transferred in several successive SDO telegrams (Normal Transfer).

In case of the IMD20, IMD40 drive module, all parameters can be transferred in the framework of the “Expedited Transfer”, with the exception of the objects 1008_h (device name), 1009_h (hardware version), $100A_h$ (software version) and 2081_h (trace data). The objects given do not have to be used in the normal operation.

Overview of the possible values of the command specifiers in case of the "Expedited Transfer"":

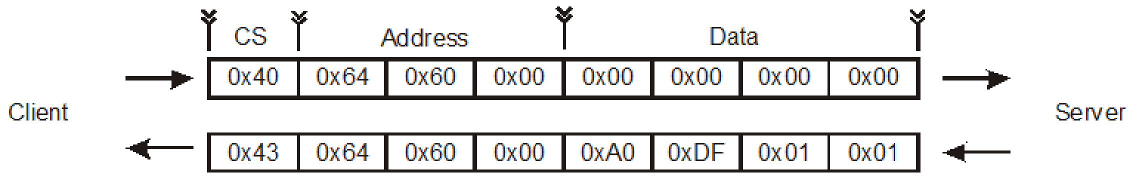
<table>
<tr><th colspan="3">Requirement (Client)</th><th colspan="3">Response (Server)</th></tr>
<tr><td rowspan="6">Read object (upload request)</td><td colspan="2">Command specifier</td><td rowspan="6">Content of the object (upload response)</td><td>Command specifier</td><td>Number of data bytes</td></tr>
<tr><td colspan="2" rowspan="5">0x40</td><td>0x42</td><td>Not specified</td></tr>
<tr><td>0x4F</td><td>1</td></tr>
<tr><td>0x4B</td><td>2</td></tr>
<tr><td>0x47</td><td>3</td></tr>
<tr><td>0x43</td><td>4</td></tr>
<tr><td rowspan="6">Write object (download request)</td><td>Command specifier</td><td>Number of data bytes</td><td rowspan="6">Confirmation (download response)</td><td colspan="2">Command specifier</td></tr>
<tr><td>0x22</td><td>No data</td><td colspan="2" rowspan="5">0x60</td></tr>
<tr><td>0x2F</td><td>1</td></tr>
<tr><td>0x2B</td><td>2</td></tr>
<tr><td>0x27</td><td>3</td></tr>
<tr><td>0x23</td><td>4</td></tr>
<tr><td rowspan="2">Interruption of the SDO communication</td><td colspan="5">Command specifier</td></tr>
<tr><td colspan="5">0x80
Data bytes include an error code indicating the reason for the interruption.</td></tr>
</table>

The SDO transfer can be interrupted by a participant for different reasons. The following error codes can be indicated by the drive module.

SDO interruption error code	Meaning
0x05030000	Toggle Bit was not changed
0x05040000	SDO protocol timeout exceeded
0x05040001	Command specifier invalid or unknown
0x06010001	Read access to write-only object
0x06010002	Write access to read-only object
0x06020000	Object not existing in the object directory
0x06040041	Mapping for this object is not admissible
0x06040043	Incompatibility of a parameter
0x06060000	Hardware error
0x06070012	Data type is not correct. The service parameter is too long
0x06090011	Sub-index not existing
0x06090030	Range of values of the parameter exceeded
0x06090031	Parameter value too high
0x06090032	Parameter value too small
0x06090042	Number and length of the parameters to be mapped exceeds the PDO length
0x08000000	General error
0x08000022	Parameter cannot be written or stored due to the current device status (mode of operation, etc.)

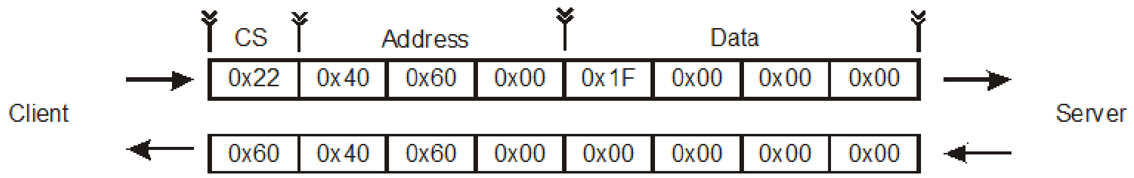
Example 1:

Query of the actual position (object 6064_h). As a response, the value 16900000_d = $0101DFA0_h$ is sent back.



Example 2:

Set the control word (object 6040_h) to the value: $001F_h$.



PDO

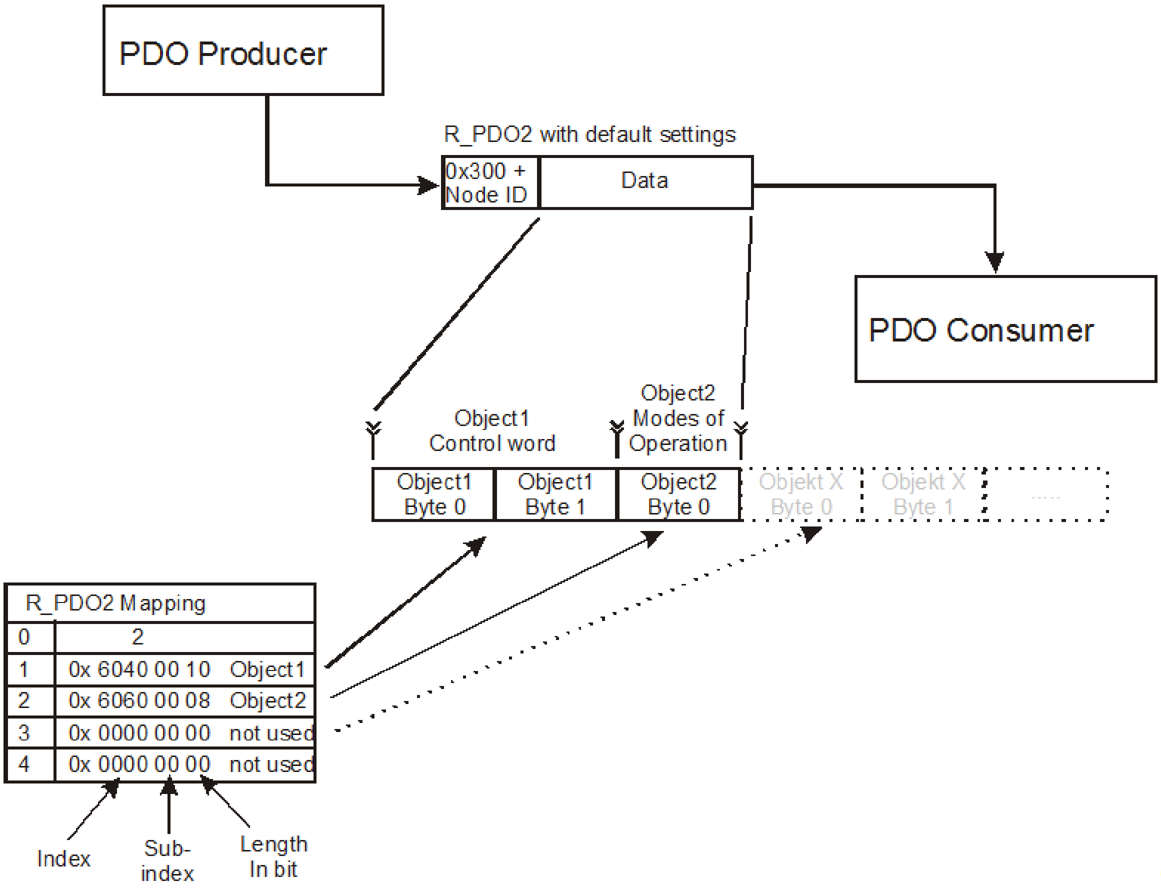
A Process Data Object (PDO) is a Can message with a defined identifier containing one or several objects with address information. The recipient (several recipients possible) knows the arrangement of the objects in this Can frame due to the Mapping table for this PDO in its object directory. The table is either preset firmly or written before the sending of the PDOs.

The receipt of a PDO is not confirmed by the recipients. The length of a PDO (1-8 bytes) depends on the number and length of the contained objects.

Mapping

A CanOpen device can dispose of several Receive PDOs and Transmit PDOs. The Mapping for the first R_PDO is contained in the object 1600h; for the next R_PDOs in the objects 1601h, 1602h, etc., the T_PDO mapping parameters as of Index 1A00h are given in the object directory.

In the figure, the default arrangement of the objects in the Receive PDO2 of the drive module is illustrated.



When the PDO arrives, the PDO-consumer checks the mapping parameter object 0x1601 belonging to the Receive-PDO2.

- The subindex 0x00 has the value 2. Therefore, the PDO consumer knows that it only has to take over the data for two of his objects
- The subindex 0x01 points to the object 0x6040_0x00 with the length of 16 bits. The PDO consumer then takes the first two PDO data bytes into its object “Controlword” (0x6040_0x00).
- The subindex 0x02 points to the object 0x6060_0x00 with the length of 8 bits. The PDO consumer then takes the third PDO byte into its object "Modes of Operation" (0x6060_0x00).

For the PDO producer, the mapping works in the same way, but only in the reverse order. That is, when requesting to send, the PDO producer looks at the mapping parameter object that belongs to the PDO. The 0x00 subindex is the number of objects to be transferred. The entries from the subindex 0x01 are processed sequentially and define from which objects the data is to be transferred.

It is interesting to know that the mapping tables of PDO producer and PDO consumer do not have to be identical. In other words, you can map an object from PDO producer to a object with a different index and subindex in the PDO consumer. In the PDO consumer, you can dismember the object data. In other words, you can only take a part of the data from the individual PDO producer object. It's just a matter of mapping table.

The mapping can be changed or extended at all times. For all PDOs of the module, it is possible to map up to 4 objects.

(See “PDO / Mapping parameters” on page 104).

Procedure for the setting of the mapping:

The setting of the mapping is carried out via SDO.

The entry under sub-index 0 in the mapping table defines the number of active objects in the PDO. Before the mapping can be changed, sub-index 0 **must** be set to 0 in order to deactivate the mapping.

Afterwards, the mapping entries can be described as of sub-index 1.

At the end, the corresponding number of objects is once again entered in the sub-index 0. If an object cannot be mapped or if the length of the PDO is exceeded, the SDO transfer is interrupted with a corresponding error message.

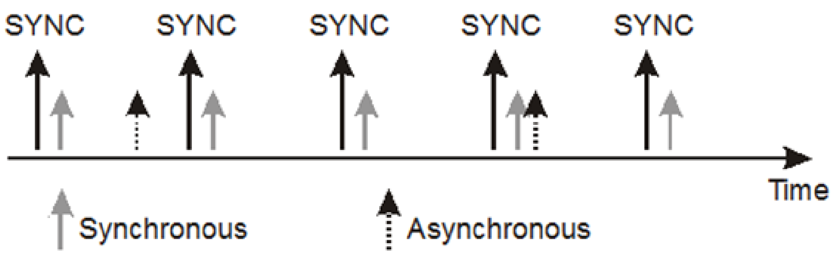
Types of transfer

The transfer of PDOs can be triggered by three different events:

- by an internal event (e.g. change of a status) or an internal timer.
- by the query of the PDO via a remote request (RTR Can object).
- by the receipt of a SYNC message.

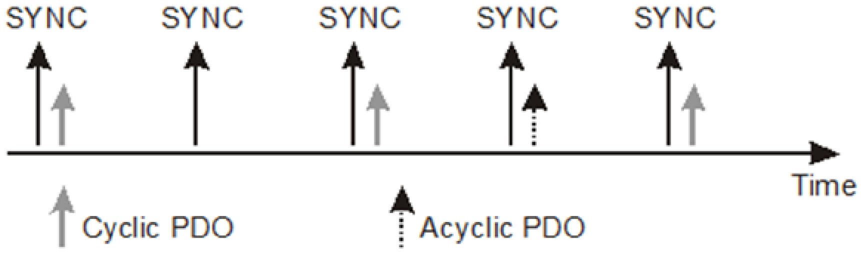
Furthermore, two types of transfer are distinguished:

- synchronous transfer. Synchronous PDOs are transferred directly after the SYNC object.
- asynchronous transfer. Asynchronous PDOs can be transferred at any given point in time.



The transfer of synchronous PDOs can again be subdivided into two types:

- Cyclic PDOs are transferred periodically according to an adjustable number of SYNC impulses (1-240).
- Acyclic PDOs are released by an internal event. They are transferred synchronously with the SYNC signal, but not periodically.



The setting of the type of transfer of PDOs is made via PDO communications parameters that are given in the object directory for R_PDOs as of index 1400h and for T_PDOs as of index 1800h. The following table shows an overview of the types of transfer and allocation to the parameter transmission type.

Type-no.	Cyclic	Acyclic	Synch-ronous	Asynch-ronous	RTR only	Meaning
0		x	x			TPDO: Transfer to SYNC but not periodically RPDO: Transfer to SYNC but not periodically
1-240	x		x			TPDO: Transfer after each x-SYNC. RPDO: Transfer of the data after each x-SYNC.
241-251						TPDO: Not used RPDO: Not used
252			x		x	TPDO: Objects in the PDO are updated upon each SYNC but are sent only upon request (RTR). RPDO: Not used.
253				x	x	TPDO: Objects in the PDO are updated and sent upon receipt of the RTR. RPDO: Not used
254				x		TPDO: Not used RPDO: Immediate confirmation upon PDO-receipt
255				x		TPDO: Event controlled PDO. The transfer is triggered by a change of the mapped parameter or by an adjustable timer. RPDO: Immediate confirmation upon PDO-receipt triggered by a change of the mapped parameter or an adjustable timer.

SYNC

The SYNC signal provides a common cycle which can be used by all connected participants e.g. for the PDO transfer. The SYNC object is sent by a CanOpen participant and received by all devices participating in a synchronous operation.

The identifier of the SYNC object is by default set to 0x80 and has thus a very high priority within the Can network. No data bytes are transferred with the SYNC.

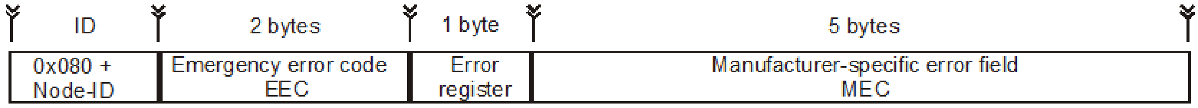
The SYNC signal plays an important role for the drive module in the Interpolation mode of operation. The superordinate control must communicate to the module beforehand the time interval of the SYNC signal; for this, the value (in µs) in the object "Communication Cycle Period" 1006_h has to be entered.

EMCY

Emergency messages are triggered by internal errors in the device and sent with a high priority to the Can bus. A superordinate control can thus react very quickly to an abnormal behaviour of individual components.

By default, the emergency message is characterised by the identifier 0x80 + node number.

With the emergency object, 8 data bytes are transferred for the error description. The structure of an emergency message looks as follows:



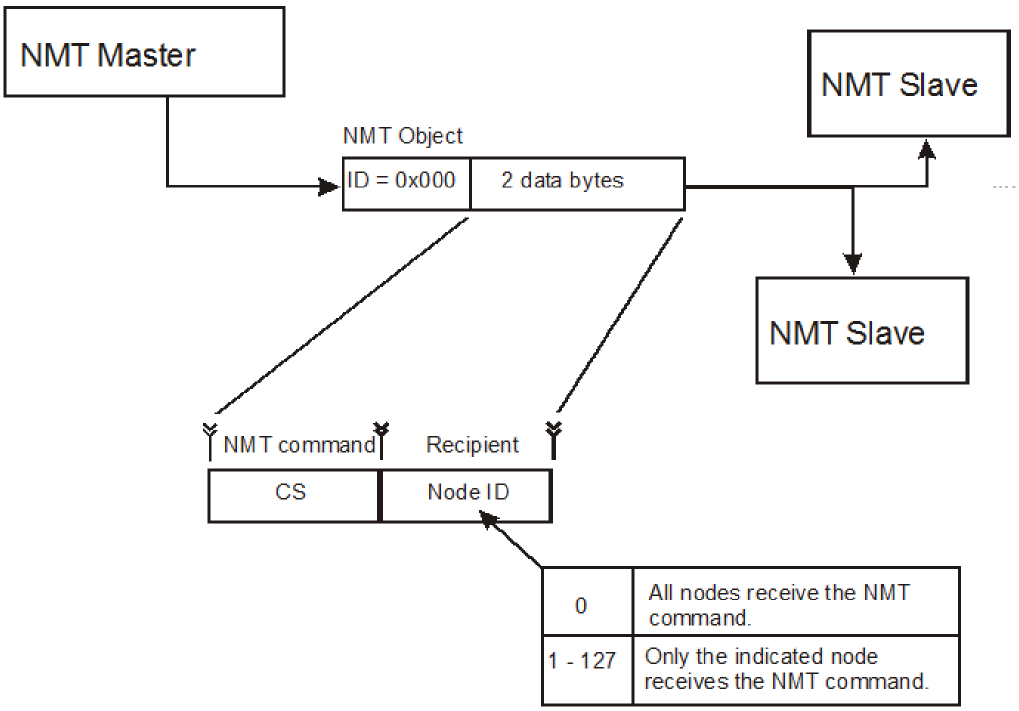
The first two bytes contain the error code of the fault that occurred. In the 3rd byte, the error register (object 1001_h) is transferred which only includes a rough classification of the type of error. The last 5 bytes may contain manufacturer-specific error information.

(See “Fault states” on page.45)

Network management - NMT

The network management administers the communication’s basic functions of the participants in the CanOpen network. Here, a master-slave configuration is taken as a basis, in which an NMT master controls and regulates the state of all other participants (NMT slaves).

Status changes of individual or all NMT slaves (Start, Stop, Reset...) are triggered by an NMT object with the Can identifier 0x000. This object is sent by the NMT master and always possesses two data bytes. The first byte contains a command code determining the state of the NMT slave. The second data byte defines, if all slaves are addressed at the same time or if individual nodes only are used.

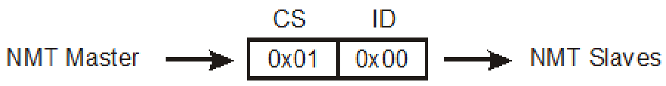


Command code (CS) of the NMT object

CS	Meaning	
0x01	“Start Remote Node“	Activate the Operational state
0x02	“Stop Remote Node“	Activate the Stopped state
0x80	“Enter Pre-Operational“	Activate the Pre-Operational state
0x81	“Reset Node“	Reset of all parameters and restart
0x82	“Reset Communication“	Reset of the communications parameters and restart

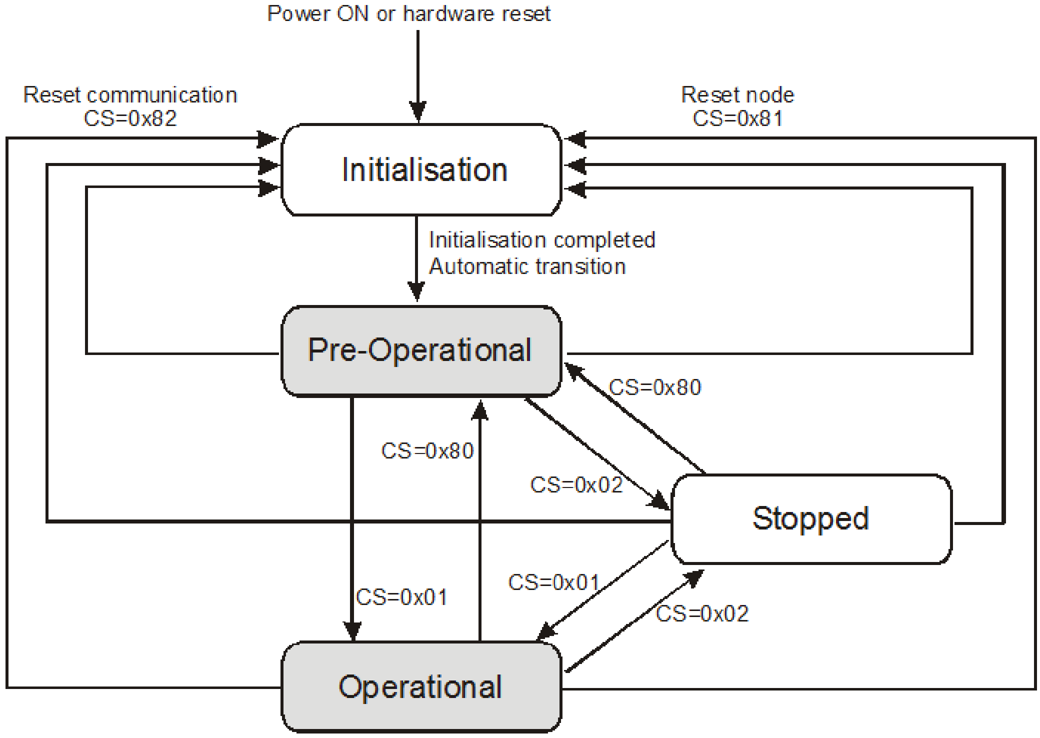
Example:

Bring all nodes into the Operational state



Status diagram

Every CanOpen participant disposes of an internal state machine which consists of four states and controls the communication behaviour of the node. The status transitions are triggered by internal events or by the receipt of NMT objects.



In the "Pre-Operational" state it is not possible to send or receive PDOs. In the "Stopped" state, no communication whatsoever is possible with the node, except for the Guarding.

Effects of the states on the processing of communications objects:

	Initialisation	Pre-Operational	Operational	Stopped
PDO			X	
SDO		X	X	
SYNC object		X	X	
EMCY object		X	X	
Boot-up object	X			
NMT objects inc. Node Guarding		X	X	X

Boot-Up object

After a successful initialisation, the module sends a boot-up object which notifies the NMT master of the fact that the node is available and has achieved the Pre-Operational state. The boot-up object uses the identifier of the guarding object and contains a data byte with the firm value 0.

ID	Byte 0
0x700 + Node-ID	0x00

Monitoring – Guarding

The guarding includes the monitoring of all NMT slaves by the master (node guarding) as well as the automatic self-interruption of the slaves in case of a failure of the NMT master (life guarding) .

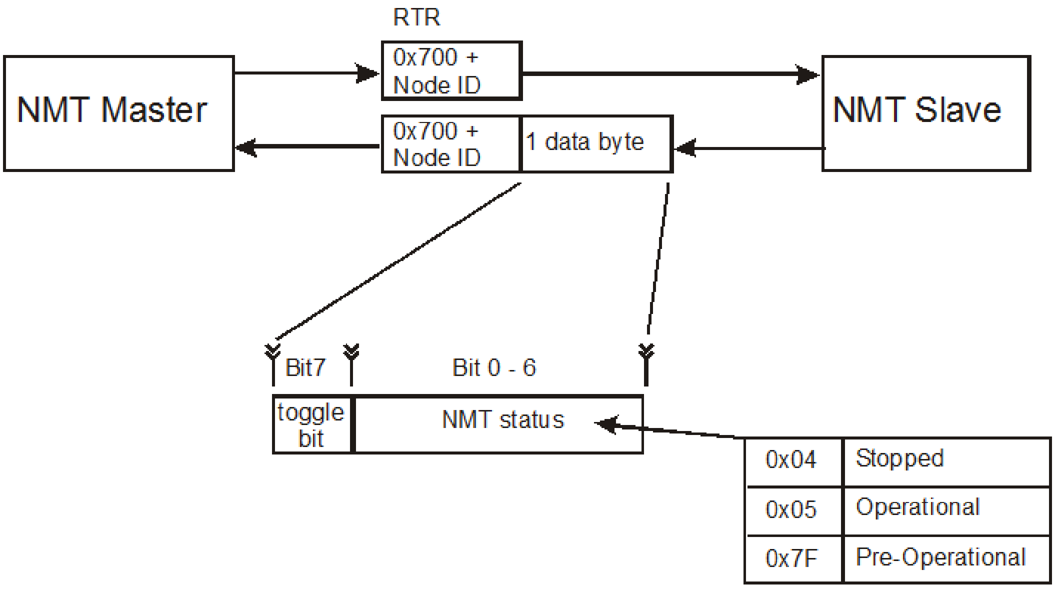
For this purpose, the master sends in regular time intervals (guard time), which can be set separately for every node, a request telegram (RTR – remote transmission request) for the guarding object of the respective slave.

The NMT slave replies within the life time (= guard time x life time factor) with the guarding object, which contains – in one data byte - the current NMT state and a toggle bit.

Upon the first query, the toggle bit has a value of 0 and changes its value (0, 1, 0, 1…) with the following guarding telegrams. If, within the life time, no response with the correct toggle bit value is sent back or if the NMT state does not match the expected state, the master has to assume an error.

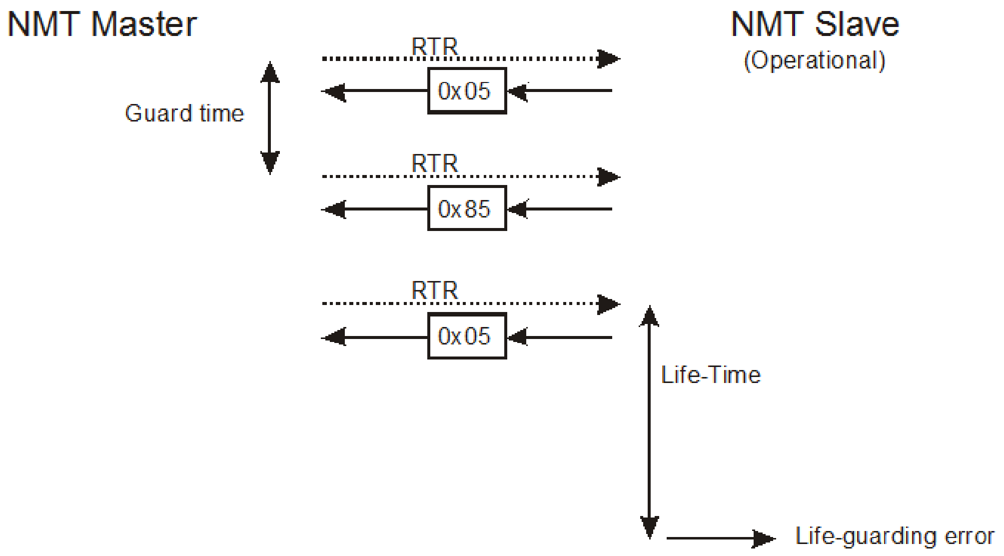
The guarding for a node is activated with the first request of the guarding object by the master and can only be switched off again through a reset of the node.

The NMT slave monitors the incoming queries of the master and switches to the fault state, if the telegrams for the life-time period fail to appear.



Example:

The module is in the Operational state. After the master does no longer sent a guarding request, the module changes to the fault state, when the life time elapsed.



Object Directory

The object directory contains all parameters and variables of the module that can be read or written via the Can network. The entries in the directory are characterised by a 16-bit index and a 8-bit sub-index.

Communications parameter

The range of 1000h to 1FFFh is reserved for the communication parameters of a CanOpen device. The communications part of the drive module software is structured according to the specification DS301 V4.0. In the following, all parameters are listed and described that are used for this and that control the communications behaviour.

General communications parameters

Device type

Index	**Sub**	**Name**	**Type**	**Attrib**	**Map**	**Default value**	**Meaning**
1000	00	Device type	Unsigned 32	RO	N	0x00020192	Device type

The device type indicates which device profile is used and which kind of device this is (e.g. servo drive = 02_h). Bit 0-15: device profile 402 = $192_{h.}$

Error register

Index	**Sub**	**Name**	**Type**	**Attrib**	**Map**	**Default value**	**Meaning**
1001	00	Error register	Unsigned 8	RO	Y	0x00	Error register

The error register contains a rough classification of an error that occurred. It is also sent as part of the emergency message. A more precise designation of the error is found in the error memory (1003_h).

Bit 7	6	5	4	3	2	1	0
-	-	Dev. prof.-specific	Commu-nication	-	Voltage	Current	general error

Manufacturer-specific status register

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1002	00	Manufacturer-specific status register	Unsigned 32	RO	Y	0x00	Manufacturer-specific status register

Presently only the bit with the lowest value is used in connection with the electronic handwheel. This bit toggles with each successful writing in the handwheel-object “Inputs” (Index $205D_h$ and Subindex $0C_h$ - see Handwheel on page 154).

Error memory

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1003		Pre-defined error field	RECORD				Error memory (list of the last errors that occurred).
1003	00	Number of errors	Unsigned 8	RW	N	0	Number of the errors occurred.
1003	01	Standard error field	Unsigned 32	RO	N		Current error.
1003	02	Standard error field	Unsigned 32	RO	N		Last but one error.
1003	03	Standard error field	Unsigned 32	RO	N		Second from the last error.
1003	04	Standard error field	Unsigned 32	RO	N		Third from the last error.
1003	05	Standard error field	Unsigned 32	RO	N		A maximum of 5 errors is stored.

The error memory includes the errors that occurred in the device and where fed back via emergency object.

- The sub-index 0 contains the number of registered errors.
- Each new error is saved; the preceding errors are moved downwards in the list.
- The writing of an "0" on the sub-index 0 deletes the error memory.
- The 32-bit long error entries consist of the error code (EEC, see “EMCY” on page 96) in the lower 2 bytes and additional information in the upper 2 bytes (MEC).

Upper 16 bits	Lower 16 bits
Additional information	Error code

COB ID Sync message

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1005	00	COB ID Sync message	Unsigned 32	CONST	N	0x00000080	Identifier of the SYNC message.

The lower 11 bits contain the identifier of the Sync-message the module can receive. The highest bit has the value 1 and indicates the users that the final stages IMD20/40 are able to consume the Sync-message.

SYNC interval

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1006	00	Communication cycle period	Unsigned 32	RW	N	0	Length of the SYNC interval in microseconds.

Interval between two successive Sync telegrams. For the Interpolation mode of operation, the exact value in µsec has to be entered. In the other modes of operation, this parameter is not evaluated.

Device name

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1008	00	Manufacturer's device name	Visible string	RO	N		Device name.

Hardware version

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1009	00	Manufacturer’s hardware version	Visible string	RO	N		Hardware version number.

Software version

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
100A	00	Manufacturer’s software version	Visible String	RO	N		Software version number.

Node number

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
100B	00	Node ID	Unsigned 32	RO	N		Node number.

Guard time

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
100C	00	Guard time	Unsigned 16	RW	N	0	Interval between two guard telegrams in milliseconds.

(See "Monitoring – Guarding" on page 99).

Life time factor

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
100D	00	Life-time factor	Unsigned 8	RW	N	0	See below.

Life-time factor x guard time = time, until the module switches to the fault state, if guard telegrams fail to appear.

(See "Monitoring – Guarding" on page 99).

Store parameters

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1010		Store parameters	RECORD				Store parameters.
1010	00	Largest supported sub-index	Unsigned 8	RO	N	1	Number of types of memory.
1010	01	Save all parameters	Unsigned 32	RW	N	0x00000001	Store all parameters.

By writing the four characters 's' 'a' 'v' 'e' in the ASCII code (0x 65 76 61 73) on the sub-index 1, all storable parameters of the module are saved permanently and internally.

This comprises:

- Communication parameter: PDO parameters (except PDO-ID), PDO mapping, guarding parameters, Sync interval.
- All writable device profile parameters except for Control word, Interpolation Data and Interpolation-Actual-Buffer-Size.
- Manufacturer-specific parameters: offset values of the analogue inputs.

Load default parameters

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1011		Restore default parameters	RECORD				Load default parameters.
1011	00	Largest supported sub-index	Unsigned 8	CONST	N	1	Number of options.
1011	01	Restore all default parameters	Unsigned 32	RW	N	0x00000001	Load all parameters with default values.
1011	02	Restore communication default parameters	Unsigned 32	RW	N	0x00000001	Load communications parameters with default values.

By writing the four characters 'l' 'o' 'a' 'd' in the ASCII code (0x 64 61 6F 6C) on the sub-index 1, all parameters are set to their default values. By writing the signature on sub-index 2, only communications parameter with default values are loaded.

High-resolution time stamp

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1013	00	High-resolution time stamp	Unsigned 32	RW	Y	0	High-resolution time stamp (µs).

Synchronisation of the interpolation cycle in the interpolation mode 2.

COB ID Error message

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1014	00	COB ID emergency message	Unsigned 32	RO	N	0x080 + Node ID	Identifier of the error message.

Identity object

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1018		Identity object	RECORD				SDO parameters of the module.
1018	00	Number of entries	Unsigned 8	CONST	N	3	Number of entries.
1018	01	Vendor ID	Unsigned 32	RO	N	0x00000031	Registered manufacturer ID.
1018	02	Product code	Unsigned 32	RO	N	0x00DC8112	Product number.
1018	03	Revision number	Unsigned 32	RO	N		Version number.

SDO parameters

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1200		Server SDO parameters	RECORD				SDO parameters of the module.
1200	00	Number of entries	Unsigned 8	CONST	N	2	Number of entries.
1200	01	COB ID Client->Server (rx)	Unsigned 32	RO	N	0x600 + Node ID	Identifier of the Can object which is sent to the module.
1200	02	COB ID Server->Client (tx)	Unsigned 32	RO	N	0x580 + Node ID	Identifier of the Can object which is sent by the module.

(See “SDO" on page.91.)

PDO / Mapping parameters

The drive controller supports 4 Receive and 4 Transmit PDOs with variable mapping of up to 4 objects per PDO. The type of transfer can be any possible way admissible in CanOpen. The identifier of the PDOs can be changed, but not stored. All other parameters (type of transfer, mapping) can be stored and must therefore not be reconfigured every time a reset has been made. See “PDO” on page 93 for a description of the types of transfer and the mapping procedure.

Receive PDO1 – communications parameters

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1400		R_PDO1 parameters	RECORD				Comm. parameters.
1400	00	Number of entries	Unsigned 8	CONST	N	2	Number of entries.
1400	01	COB ID	Unsigned 32	RO	N	0x200+Node no.	Identifier of the RxPDO1.
1400	02	Transmission type	Unsigned 8	RW	N	1	Transfer type of the PDO.

Receive PDO2 – communications parameters

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1401		R_PDO2 parameters	RECORD				Comm. parameters.
1401	00	Number of entries	Unsigned 8	CONST	N	2	Number of entries.
1401	01	COB ID	Unsigned 32	RO	N	0x300+Node no.	Identifier of the RxPDO2.
1401	02	Transmission type	Unsigned 8	RW	N	1	Transfer type of the PDO.

Receive PDO3 – communications parameters

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1402		R_PDO3 parameters	RECORD				Comm. parameters of the 3rd receive PDO.
1402	00	Number of entries	Unsigned 8	CONST	N	2	Number of entries.
1402	01	COB ID	Unsigned 32	RO	N	0x400+Node no.	Identifier of the RxPDO3.
1402	02	Transmission type	Unsigned 8	RW	N	1	Transfer type of the PDO.

Receive PDO4 – communications parameters

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1403		R_PDO4 parameters	RECORD				Comm. parameters.
1403	00	Number of entries	Unsigned 8	CONST	N	2	Number of entries.
1403	01	COB ID	Unsigned 32	RO	N	0x500+Node no.	Identifier of the RxPDO4.
1403	02	Transmission type	Unsigned 8	RW	N	1	Transfer type of the PDO.

Receive PDO1 – mapping parameters

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1600		R_ PDO1 mapping	RECORD				Mapping parameters.
1600	00	No. of obj. in PDO	Unsigned 8	RW	N	1	Number of objects.
1600	01	Mapping for 1st obj.	Unsigned 32	RW	N	0x60400010	Mapping of the 1st object.
1600	02	Mapping for 2nd obj.	Unsigned 32	RW	N	0x00000000	Mapping of the 2nd object.
1600	03	Mapping for 3rd obj.	Unsigned 32	RW	N	0x00000000	Mapping of the 3rd object.
1600	04	Mapping for 4th Obj.	Unsigned 32	RW	N	0x00000000	Mapping of the 4th object.

Receive PDO2 – mapping parameters

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1601		R_ PDO2 mapping	RECORD				Mapping parameters.
1601	00	No. of obj. in PDO	Unsigned 8	RW	N	2	Number of objects.
1601	01	Mapping for 1st obj.	Unsigned 32	RW	N	0x60400010	Mapping of the 1st object.
1601	02	Mapping for 2nd obj.	Unsigned 32	RW	N	0x60600008	Mapping of the 2nd object.
1601	03	Mapping for 3rd obj.	Unsigned 32	RW	N	0x00000000	Mapping of the 3rd object.
1601	04	Mapping for 4th obj.	Unsigned 32	RW	N	0x00000000	Mapping of the 4th object.

Receive PDO3 - Mapping-Parameter

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1602		R_ PDO3 Mapping	RECORD				Mapping parameters.
1602	00	No. of obj. in PDO	Unsigned 8	RW	N	2	Number of objects.
1602	01	Mapping for 1st obj.	Unsigned 32	RW	N	0x60400010	Mapping of the 1st object.
1602	02	Mapping for 2nd obj.	Unsigned 32	RW	N	0x607A0020	Mapping of the 2nd object.
1602	03	Mapping for 3rd obj.	Unsigned 32	RW	N	0x00000000	Mapping of the 3rd object.
1602	04	Mapping for 4th obj.	Unsigned 32	RW	N	0x00000000	Mapping of the 4th object.

Receive PDO4 – mapping parameters

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1603		R_ PDO4 mapping	RECORD				Mapping parameters.
1603	00	No. of obj. in PDO	Unsigned 8	RW	N	2	Number of objects.
1603	01	Mapping for 1st obj.	Unsigned 32	RW	N	0x60400010	Mapping of the 1st object.
1603	02	Mapping for 2nd obj.	Unsigned 32	RW	N	0x60810020	Mapping of the 2nd object.
1603	03	Mapping for 3rd obj.	Unsigned 32	RW	N	0x00000000	Mapping of the 3rd object.
1603	04	Mapping for 4th obj.	Unsigned 32	RW	N	0x00000000	Mapping of the 4th object.

Transmit PDO1 – communication parameters

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1800		T_PDO1 parameters	RECORD				Comm. parameters.
1800	00	Number of entries	Unsigned 8	CONST	N	2	Number of entries.
1800	01	COB ID	Unsigned 32	RO	N	0x180+Node no.	Identifier of the TxPDO1.
1800	02	Transmission type	Unsigned 8	RW	N	1	Transfer type of the PDO.
1800	03	Inhibit time	Unsigned 16	RW	N	50	Minimal sending interval for TxPDO1 (in [100 µs]).
1800	04	Compatibility entry	Unsigned 8	RW	N	0	No function.
1800	05	Event timer	Unsigned 16	RW	N	0	Timer for cyclic sending (unit: [1 ms]).

Transmit PDO2 – communication parameters

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1801		T_PDO2 parameters	RECORD				Comm. parameters.
1801	00	Number of entries	Unsigned 8	CONST	N	2	Number of entries.
1801	01	COB ID	Unsigned 32	RO	N	0x280+Node no.	Identifier of the TxPDO2.
1801	02	Transmission type	Unsigned 8	RW	N	1	Transfer type of the PDO.
1801	03	Inhibit time	Unsigned 16	RW	N	50	Minimal sending interval for TxPDO2 (in [100 µs]).
1801	04	Compatibility entry	Unsigned 8	RW	N	0	No function.
1801	05	Event timer	Unsigned 16	RW	N	0	Timer for cyclic sending (unit: [1 ms]).

Transmit PDO3 – communication parameters

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1802		T_PDO3 parameters	RECORD				Comm. parameters.
1802	00	Number of entries	Unsigned 8	CONST	N	2	Number of entries.
1802	01	COB ID	Unsigned 32	RO	N	0x380+Node no.	Identifier of the TxPDO3.
1802	02	Transmission type	Unsigned 8	RW	N	1	Transfer type of the PDO.
1802	03	Inhibit time	Unsigned 16	RW	N	50	Minimal sending interval for TxPDO3 (in [100 µs]).
1802	04	Compatibility entry	Unsigned 8	RW	N	0	No function.
1802	05	Event timer	Unsigned 16	RW	N	0	Timer for cyclic sending (unit: [1 ms]).

Transmit PDO4 – communication parameters

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1803		T_PDO4 parameters	RECORD				Comm. parameters.
1803	00	Number of entries	Unsigned 8	CONST	N	2	Number of entries.
1803	01	COB ID	Unsigned 32	RO	N	0x480+Node no.	Identifier of the TxPDO4.
1803	02	Transmission type	Unsigned 8	RW	N	1	Transfer type of the PDO.
1803	03	Inhibit Time	Unsigned 16	RW	N	50	Minimal sending interval for TxPDO4 (in [100 µs]).
1803	04	Compatibility entry	Unsigned 8	RW	N	0	No function.
1803	05	Event timer	Unsigned 16	RW	N	0	Timer for cyclic sending (unit: [1 ms]).

Transmit PDO1 – Mapping parameters

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1A00		T_ PDO1 mapping	RECORD				Mapping parameters.
1A00	00	No. of obj. in PDO	Unsigned 8	RW	N	1	Number of objects.
1A00	01	Mapping for 1st obj.	Unsigned 32	RW	N	0x60410010	Mapping of the 1st object.
1A00	02	Mapping for 2nd obj.	Unsigned 32	RW	N	0x00000000	Mapping of the 2nd object.
1A00	03	Mapping for 3rd obj.	Unsigned 32	RW	N	0x00000000	Mapping of the 3rd object.
1A00	04	Mapping for 4th obj.	Unsigned 32	RW	N	0x00000000	Mapping of the 4th object.

Transmit PDO2 – Mapping parameters

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1A01		T_ PDO2 Mapping	RECORD				Mapping parameters.
1A01	00	No. of obj. in PDO	Unsigned 8	RW	N	2	Number of objects.
1A01	01	Mapping for 1st obj.	Unsigned 32	RW	N	0x60410010	Mapping of the 1st object.
1A01	02	Mapping for 2nd obj.	Unsigned 32	RW	N	0x60610008	Mapping of the 2nd object.
1A01	03	Mapping for 3rd obj.	Unsigned 32	RW	N	0x00000000	Mapping of the 3rd object.
1A01	04	Mapping for 4th obj.	Unsigned 32	RW	N	0x00000000	Mapping of the 4th object.

Transmit PDO3 – Mapping parameters

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1A02		T_ PDO3 mapping	RECORD				Mapping parameters of the 3rd Transmit PDO.
1A02	00	No. of obj. in PDO	Unsigned 8	RW	N	2	Number of objects.
1A02	01	Mapping for 1st obj.	Unsigned 32	RW	N	0x60410010	Mapping of the 1st object.
1A02	02	Mapping for 2nd obj.	Unsigned 32	RW	N	0x60640020	Mapping of the 2nd object.
1A02	03	Mapping for 3rd obj.	Unsigned 32	RW	N	0x00000000	Mapping of the 3rd object.
1A02	04	Mapping for 4th obj.	Unsigned 32	RW	N	0x00000000	Mapping of the 4th object.

Transmit PDO4 – Mapping parameters

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
1A03		T_ PDO4 Mapping	RECORD				Mapping parameters.
1A03	00	No. of obj. in PDO	Unsigned 8	RW	N	2	Number of objects.
1A03	01	Mapping for 1st obj.	Unsigned 32	RW	N	0x60410010	Mapping of the 1st object.
1A03	02	Mapping for 2nd obj.	Unsigned 32	RW	N	0x606C0020	Mapping of the 2nd object.
1A03	03	Mapping for 3rd obj.	Unsigned 32	RW	N	0x00000000	Mapping of the 3rd object.
1A03	04	Mapping for 4th obj.	Unsigned 32	RW	N	0x00000000	Mapping of the 4th object.

Device profile parameters

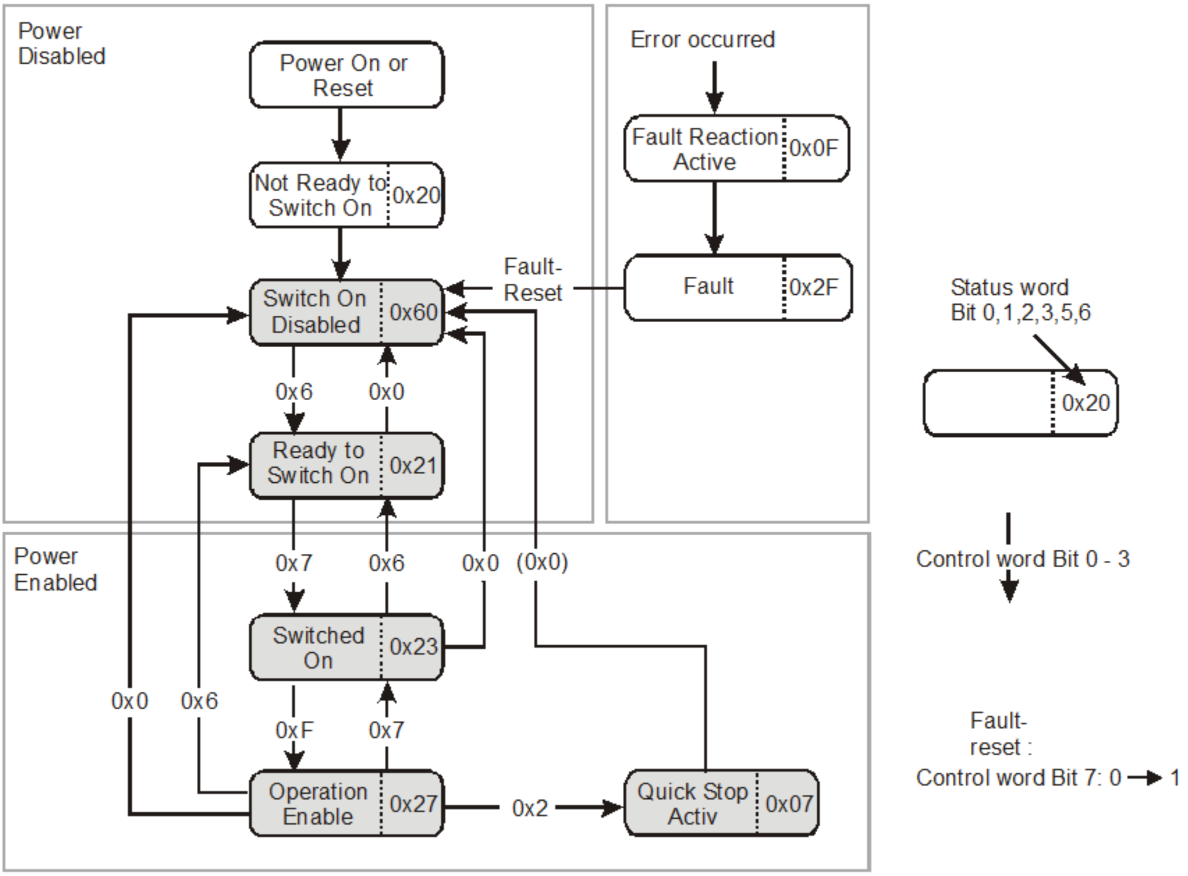
CanOpen comprises different devices (e.g. I/O modules, drives, encoders ...), so-called device profiles, in which pre-defined parameters and modes of operation are available for each device class. According to the functional scope of a device, more or less of these default objects can be implemented.

The drive module uses the DS402 V1.0 device module. In the following, all used parameters are listed and described according to functional groups. In this regard, all drive functions and the behaviour of the module are shown in the different modes of operation.

Device control

The drive module comprises a so-called state machine which defines which operating states may be applied and how the transition to other states can be triggered. This status diagram is controlled by a Control word or by an event (e.g. errors that occurred). Via the status word the current state can be queried.

State machine



Description of the individual states:

- **Not Ready to Switch On:** initialisation.
- **Switch On Disabled:** Initialisation completed, final stage blocked.

  Drive functions blocked. Brake in the automatic operation mode active.
- **Ready to Switch On:** Final stage blocked. Drive functions blocked.

  Brake in the automatic operation mode active.
- **Switched On:** Final stage released. Drive functions blocked.

Brake in the automatic operation mode active.

- **Operation Enable:** Final stage released. Drive functions enabled. Motor is live.
- **Quick Stop Active:** Quick Stop function (braking with maximum ramp) is realised. The motor is live. After the motor achieved its standstill, automatic transition to the Switch On Disabled state.
- **Fault Reaction Active:** An error occurred. The motor is braked with the Quick-Stop ramp, then transfer to the Fault state. In case of critical faults, the system automatically switches to the Fault state. Brake in the automatic mode not active.
- **Fault:** Final stage blocked. Drive functions blocked.

Control word

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6040	00	Control word	Unsigned 16	RW	Y	0x00	Control of all important drive functions.

Meaning of the bits in the control word

Bit	All modes of operation	Mode of operation: “Velocity control”	Mode of operation: “Profile position mode”	Mode of operation: “Homing mode”	Mode of operation: “Interpolation”
0-3	Control of the state machine.				
4	-	-	Apply new set of position.	Start reference run.	Release of interpolation mode.
5	-	-	The set of position immediately becomes valid.	-	-
6	-	-	0 = absolute / 1 = relative	-	-
7	Reset fault.				
8	Stop	Braking with normal ramp.	Interrupt positioning. (motion vector is maintained)	Interrupt reference run.	-
9-10	Not used.				
11	-	-	Cancel positioning. (motion vector is lost)	Cancel reference run.	-
12-15	Not used.				

Status word

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6041	00	Status word	Unsigned 16	RO	Y	0x00	Status of all important drive functions.

Meaning of the bits in the status word

Bit	All modes of operation	Mode of operation: “Velocity control”	Mode of operation: “Profile position mode”	Mode of operation: “Homing mode”	Mode of operation: “Interpolation”
0-3	Status of the state machine.				
4	Final stage blocked (no enabling signal at input 4).				
5-6	Status of the state machine.				
7	I^2t – current limitation is enabled.				
8	Main supply voltage is missing.				
9	Release of the controller via control word possible.				
10	Target value achieved.	Speed achieved.	Position achieved.	Position achieved.	-
11			Software limit switch active		
12	-	Speed = 0	Set of position applied.	Reference run completed.	Interpolation released.
13	-	-	Following error.	Error during reference run.	Following error.
14	-	-	Motion active.	Motion active.	Buffer full.
15	-	-	-	-	Interpolation stopped (see explication beneath).

In the reference run mode, bit 11 is set when the switch is approached. What kind of switch is meant here depends on the reference run method (see Section “Homing Mode – Reference run” on page 120).

- Reference run method: 1, 2, 3, 6, 17, 18, 19, 21

  If you carry out a reference run to a limit switch with/without index signal or to a reference switch with/without index signal, the limit switch or reference switch used is meant here. Bit 11 is set to indicate that the switch has tripped. Any index signal use is of no importance here.

- Reference run method: 32, 33

  If you carry out the reference run only on the index signal, a set bit 11 indicates that the index signal has occurred

- Reference run method: 34

  In reference run method 34, where the current position is set as the zero point, there is no switch and no index signal. Therefore bit 11 is not used here.

In the reference run mode, bit 15 indicates that the last reference run phase "Approach Reference Distance" is active. This means that the switch or the index signal has already been found. From the switch or the index signal, the axis is currently moving the reference distance to reach the final zero point. This bit is not set in reference run method 34.

A higher-level controller uses the two bits 11 and 15 in the reference run mode to coordinate the reference run of a gantry axis in remote mode (see section “Commissioning of reference run of a gantry axis” on page 77). If the remote mode of reference run is not active, these two bits do not matter, although they are set or reset normally during the reference run. The same applies to a single axis.

In the operation mode interpolation, the bit 15 is configured, when the interpolation is stopped due to the following reasons:

- The Sync-message has not arrived or

- The buffer of the interpolation data is empty or
- In the operation mode “Fast stop on inputs” the motion stops due to the detected bit sample. Important: the bit is only set after the standstill of the axis (see “Fast stop on inputs” on page 149).

Mode of operation

The setting of the mode of operation is realised via the 6060h object. This parameter can only be written. The query of the active mode of operation is carried out via the 6061h object.

The following modes of operation can be set in the module via these parameters:

Mode of operation	Parameter value
Moving away from the limit switch	-3
Velocity control (analogue input)	-2
Positioning (profile position mode)	1
Velocity control (profile velocity mode)	3
Reference run (homing mode)	6
Interpolation (interpolated position mode)	7

Setting of Modes of operation

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6060	00	Modes of operation	Signed 8	WO	Y	1	Setting of the mode of operation.

Display of the mode of operation

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6061	00	Modes of operation display	Signed 8	RO	Y	1	Display of the current mode of operation.

Current control

The following objects refer to the current settings and the regulation of the motor current. During the commissioning, these parameters have to be set according to the motor used and the desired behaviour. For this, see "Motor and Transmission dialogue box" on page 62.

Maximum current

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6073	00	Maximum current	Unsigned 16	RW	N	12000	Maximum output current in mA.

The maximum current can amount to values between 1,000 and 25,000 (mA). The maximum current is limited by 25000 (mA) at IMD20 and is limited by 8000 (mA) at IMD40.

Rated current

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6075	00	Motor's rated current	Unsigned 32	RW	N	3000	Rated current in mA.

The admissible settings for the rated current are between 0 to 10,000 mA. The rated current is limited by 25000 (mA) at IMD20 and is limited by 8000 (mA) at IMD40.

I2t – protection time

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6510	06	I2t – current limitation time	Unsigned 16	RW	N	500	I²t – current limitation time in ms.

Values between 10 and 60000 (ms) can be set.

Current controller parameter

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
60F6		Torque Control Parameter	RECORD				Parameter of the current controller
60F6	00	Number of Entries	Unsigned 8	CONST	N	2	Number of entries
60F6	01	Kp	Unsigned 16	RW	N	8	Proportional amplification.
60F6	02	Ki	Unsigned 16	RW	N	30	Integral amplification.
60F6	03	Fast Sample (Dummy)	Unsigned 8	RW	N	0	Defines the limitation time for the current control 0: Standard 1: Fast
60F6	04	Iq_limit	Unsigned 16	RW	N	1428	Limitation of output of Iq current controller
60F6	05	Id_limit	Unsigned 16	RW	N	71	Limitation of output of Id current controller

(See "Current Controller dialogue box" on page.60.)

Profile Velocity Mode – velocity control with ramp profile

The objects given here are important for the velocity control mode of operation. Furthermore, the **Acceleration** parameter (6083_h) is used in this mode of operation.

The two parameters **Target velocity** and **Acceleration** define the speed profile which is implemented by the drive. As soon as the target speed parameter changes its value, the motor is accelerated with the given acceleration or braked, until the new target speed value is achieved. In the **Status word**, Bit10 (Target Reached) is set, if the actual velocity remains in the **Velocity Window** for the **Velocity Window Time**.

The position control works with encoder-increment as position unit. Internally a 32-Bit-position counter is used, whose counter level in the object is on 'Actual Position (inc)" (6063_h). During the endless turning the position counter will overflow at a certain moment, a fact that does not have any influence in this operation mode on the operation. But do not forget that the position during the overflow changes rapidly

- from -0x7FFFFFFF to 0x7FFFFFFF in case of the negative rotation direction or
- from 0x7FFFFFFF to -0x7FFFFFFF in case of the positive rotation direction.

In the object “Actual Position (udu)“ the actual position is in the user unit. From the internal position counter, the object “Actual Position (udu)“ is converted with the aid of the calculation factor (object “Position Factor“ - 6093_h). Due to the conversion, two apparently defective situations may arise.

- With a conversion factor less than 1, the object “Actual Position (udu)“ reaches the overflow faster than the object “Actual Position (inc)“. In this case the object “Actual Position (udu)“ will remain on one of the two values +/-0x7FFFFFF, whereas the object “Actual Position (inc)“ goes on counting upwards or downwards until the overflow. After the overflow from the object “Actual Position (inc)“ the object “Actual Position (udu)“will change again.
- With a conversion factor bigger than 1, the object “Actual Position (inc)“ reaches the overflow before the object “Actual Position (udu)“. Since the object “Actual Position (udu)“ has been calculated from the object “Actual Position (inc)“, the object “Actual Position (udu)“ will also change fast during the overflow from the object “Actual Position (inc)“. However, the fast change of the object “Actual Position (udu)“ is not situated at +/-0x7FFFFFFF, but at values whose absolute values are smaller than 0x7FFFFFFF.

However, this is not an error, only a problem of the conversion and the 32-Bit-number limitation.

Target velocity

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
60FF	00	Target velocity	Signed 32	RW	Y	0	Target velocity in the profile velocity mode in user-defined units / s.

Actual velocity

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
606C	00	Velocity: actual value (user-defined units / s)	Signed 32	RO	Y	0	Actual velocity in user-defined units / s.

Actual demand velocity value

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
606B	00	Velocity: target value (Increment/s)	Signed 32	RO	Y	0	Actual target velocity value in Increment/s.

Actual velocity sensor value

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6069	00	Velocity sensor: actual value	Signed 32	RO	Y	0	Actual velocity in Increment/s.

Velocity sensor selection

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
606A	00	Sensor selection: Code	Signed 16	RO	N	0	Type of the velocity sensor (0 = velocity is determined on the basis of the position encoder).

Velocity window

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
606D	00	Velocity window	Unsigned 16	RW	N	100	Velocity window for the achievement of the target velocity in user-defined units / s.

Velocity window time

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
606E	00	Velocity window time	Unsigned 16	RW	N	1	Time frame for the achievement of the target velocity in milliseconds.

Velocity controller parameters

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
60F9		Velocity control parameters	RECORD				Parameters of the velocity controller.
60F9	00	Number of entries	Unsigned 8	CONST	N	6	Number of entries.
60F9	01	kp	Unsigned 16	RW	N	50	Proportional amplification.
60F9	02	ki	Unsigned 16	RW	N	5	Integral amplification.
60F9	03	kd	Unsigned 16	RW	N	1000	Differential amplification.
60F9	04	td	Unsigned 8	RW	N	0	Scan time of the D proportion in units of the controller scan time.
60F9	05	e_limit	Unsigned 16	RW	N	200	Following limitation.
60F9	06	hard_limit	Unsigned 8	RW	N	0	Method of the following limitation.

For the setting of the velocity controller, see "Velocity Controller dialogue box" on page 65. For the optimisation of the following limitation value, see “Can Interpolation dialogue box” on page 73.

Profile Position Mode – position control with ramp profile

In the Profile Position Mode, a motion segment can be allocated to the drive module which the controller automatically realises. During the processing of a segment, the parameters for the next movement can already be set, so that individual segments can be lined up seamlessly.

The setting of the individual movement segments is realised via the parameters **Target position, Segment velocity, Final velocity** and **Acceleration**, the same values being used for acceleration and braking ramps.

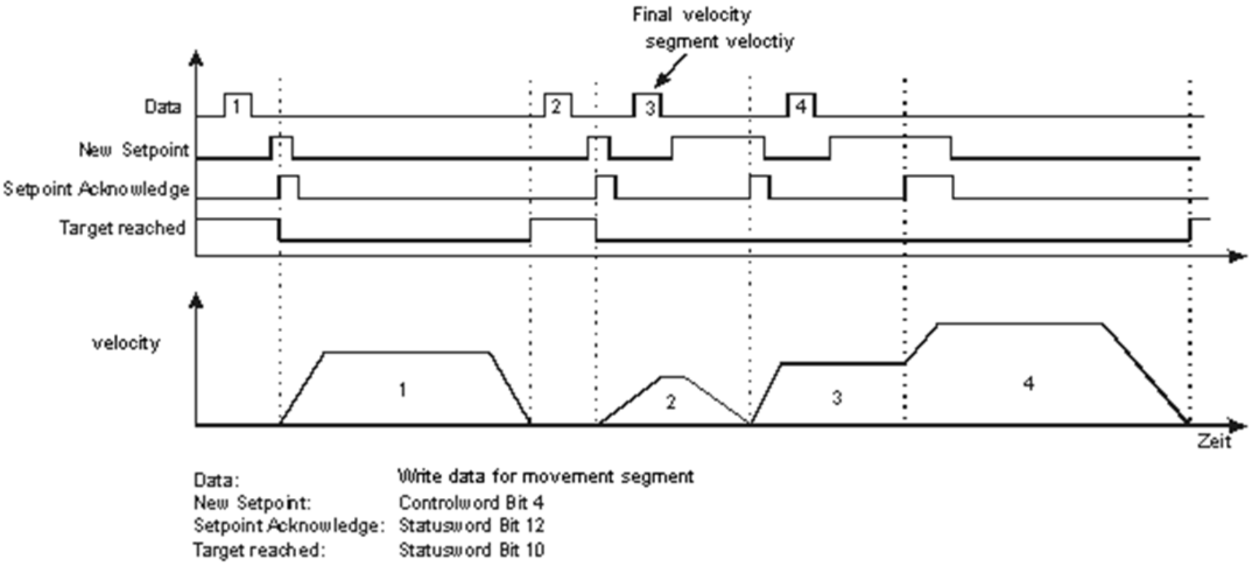
Via **Control word** and **Status word** a handshake is made during the transfer of the segments.

In the following illustration, 4 movement segments are executed after each other.

The data for the second segment are transferred only, after the first segment was completed. After the transfer of the motion parameter, the respective Bit 4 (New Setpoint) is set in Control word in order to tell the drive to apply the new movement parameters and to launch the movement. The controller confirms the acknowledgement with the Bit 12 in the Status word (Setpoint Acknowledge). If Bit 4 is reset in the control word, the drive replies with the revocation of the Setpoint Acknowledge and signalises thus that it is prepared to transfer further data.

After the second movement segment was launched, the master immediately begins with the transfer of data for the third movement segment and sets Bit 4 in the control word. The drive module applies the data at the end of the current segment and simultaneously starts the new movement.

Likewise, Segment 4 is directly put after Segment 3; the drive, however, does not brake, since in Segment 3 the final velocity is identical to the segment velocity.



If, in addition to Bit 4, Bit 5 is set in the control word (Change Set Immediately), the current movement is interrupted and the execution of a new segment is immediately started with. Here, the system accelerates or brakes down to the new segment velocity according to the ramp set.

With the Bit 6 in the Control word, there exists the option of setting the position of the new segment as absolute or relative. A recurrent relative movement can at some stage lead to an overflow of the internal 32-bit position counter. The module highlights the overflow with the error code 44. To avoid the overflow, you can

- either run the reference run. This resets all relevant positions.
- or shift the reference point. The object 2056h “Change Home Position” can be used for this (see “Reference point” on page 144).

Target position

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
607A	00	Target position	Signed 32	RW	Y	0	Target position of a movement segment in user-defined units.

Profile velocity

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6081	00	Profile velocity	Unsigned 32	RW	N	10000	Velocity during the movement segment in user-defined units / s.

End velocity

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6082	00	End velocity	Unsigned 32	RW	N	0	Velocity at the end of the movement segment in user-defined units / s.

Profile acceleration

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6083	00	Profile acceleration	Unsigned 32	RW	N	100000	Acceleration in user-defined units / s².

Position demand value

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
60FC	00	Position demand value (increment)	Signed 32	RO	Y	0	Actual position target value (increment).

Actual position (Incr.)

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6063	00	Actual Position (increment)	Signed 32	RO	Y	0	Actual position value in increments.

Actual position (udu)

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6064	00	Actual position (user-defined units)	Signed 32	RO	Y	0	Actual position value in user-defined units.

Following error position window

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6065	00	Following error window	Unsigned 32	RW	Y	1000	Position window for the monitoring of the following error [Incr.].

The monitoring of the following error is realised by means of the two parameters “Following error position window” and "Following error time window". The following error is indicated in the status word (Bit 13), if the actual position deviates from the target value for the "Following error time window" by the "Following error position window”.

Following error time window

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6066	00	Following error timeout	Unsigned 16	RW	Y	10	Time window for the monitoring of the following error [milliseconds].

Following error actual value

Index	Sub	Name	Typ	Attrib	Map	Default value	Meaning
60F4	00	Following Error Aktual Value	Signed 32	RO	Y	0	Actual position following error in user defined unit

The actual position following error (desired position – actual position) is showed only in the following operations modes (see operation mode on page 112):

- profile velocity mode
- profile position mode
- interpolated position mode
- homing mode
- limit switch leaving

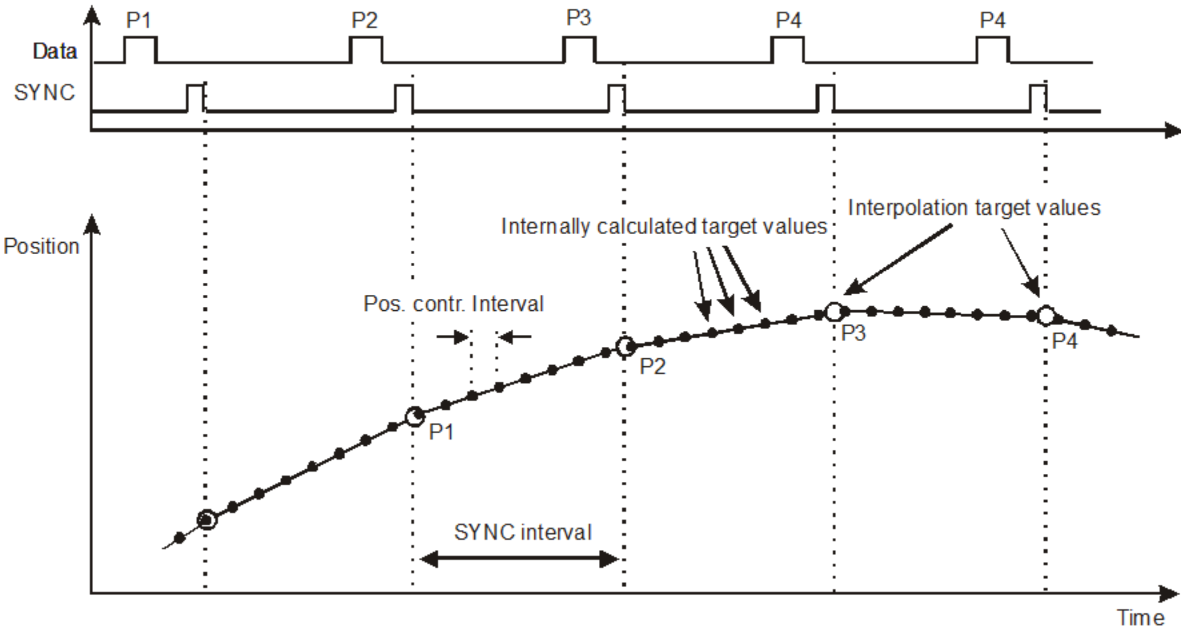
Position controller parameters

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
60FB		Position control parameters	RECORD				Parameters of the position controller.
60FB	00	Number of entries	Unsigned 8	CONST	N	6	Number of entries.
60FB	01	kp	Unsigned 16	RW	N	10	Proportional amplification.
60FB	02	Ki(dummy)	Unsigned 16	RW	N	0	Integral amplification.
60FB	03	kd	Unsigned 16	RW	N	100	Differential amplification.
60FB	04	td	Unsigned 8	RW	N	0	Scan time of the D proportion in units of the controller scan time.
60FB	05	kv	Unsigned 16	RW	N	100	Velocity amplification factor kv.
60FB	06	Ff	Unsigned 8	RW	N	0	Feed forward factor.

The parameterisation of the position controller can be carried out by means of the ACSetup programme (see “Position Controller dialogue box” on page 71).

Interpolated Position Mode – Position control with interpolation

The interpolation principle is easy to apply. At first, the “Interpolation Submode Select” ($60C0_h$) parameter is set to equal 0. Via the parameter “Interpolation Data” ($60C1_h$ – sub-index 1), the superordinate control hands over a new target position value (in user-defined units) to the drive controller in precisely defined time intervals (Sync time). The drive generates target values for its position controller so as to make sure that the given target position value is achieved in linear at the end of the time interval.



In order to launch the interpolation, the Sync signal has to be active and Bit 4 has to be set in the control word.

The length of the SYNC interval in µs has to be entered in the "Communication Cycle Period" (1006_h) object so as to make sure that the drive calculates the interim values correctly.

If the SYNC signal fails or comes in too late (0.5-1 ms), the interpolation process is stopped and the Bit 15 is set in the status word. For this, the last target position is maintained.

Selection of the interpolation method

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
60C0	00	Interpolation sub-mode select	Signed 16	RW	N	0	0: selection of the interpolation process.

Data record for interpolation

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
60C1		Interpolation data record	RECORD				Data record for interpolation.
60C1	00	Number of entries	Unsigned 8	CONST	N	1	Number of entries.
60C1	01	Interpolation data	Signed 32	RW	Y	0	Interpolation data (absolute value target position in user-defined-unit).

Interpolation interval

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
60C2		Interpolation time period	RECORD				Interpolation cycle in µs.
60C2	00	Number of entries	Unsigned 8	CONST	N	2	Number of entries.
60C2	01	ip time units	Unsigned 16	RW	N	3000	Scan time for the interpolation.
60C2	02	ip time index	Interger 8	RW	N	-6	Time unit -6 corresponds to µs (1 µs = 10^{-6} s).

Configuration of the interpolation buffer

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
60C4		Interpolation data configuration	RECORD				Configuration of the interpolation buffer.
60C4	00	Number of entries	Unsigned 8	CONST	N	2	Number of entries.
60C4	01	Max. buffer size	Unsigned 8	RW	N	31	Maximum size of the target value buffer.
60C4	02	Actual size	Unsigned 8	RW	N	15	Adjustable size of the target value buffer.

Homing Mode – Reference run

The objects listed here affect the execution of a drive module's reference run. The reference run is required for the positioning module in order to define the zero point of the axis.

(See "Reference Run dialogue box" on page 76).

The type as well as the velocities and acceleration during the search for the zero point can be set.

For the execution of the reference run, the module has to be switched to the Homing Mode of operation. Via the control word, the reference run is launched and, with the help of the status word, the actual reference run status can be queried:

Control word Bit 4	Meaning
0	Reference run not active.
0 -> 1	Launch reference run.
1	Reference run active.
1 -> 0	Interrupt reference run.

Status word Bit 12	Status word Bit 13	Meaning
0	0	Reference run not yet completed.
0	1	Reference run successfully completed.
1	0	Error occurred during reference run.

Homing method

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6098	00	Homing method	Signed 8	RW	N	17	Type of reference run.

The following possibilities can be selected:

Homing method	Direction	Target	Reference point for zero position
1	negative	limit switch	Next index signal after the limit switch during the moving out of the limit switch.
2	positive	limit switch	Next index signal after the limit switch during the moving out of the limit switch.
3	positive	reference switch	Next index signal after the limit switch during the moving out of the limit switch.
6	negative	reference switch	Next index signal after the limit switch during the moving out of the limit switch.
17	negative	limit switch	limit switch
18	positive	limit switch	limit switch
19	positive	reference switch	reference switch
21	negative	reference switch	reference switch
32	negative	index signal	index signal
33	positive	index signal	index signal
34	no run	current position	current position

Homing speeds

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6099		Homing speeds	RECORD				Speeds during the reference run.
6099	00	Number of entries	Unsigned 8	CONST	N	2	Number of entries.
6099	01	Speed during search for switch	Unsigned 32	RW	N	10000	Speed during the movement to the switch in user-defined units / s.
6099	02	Speed during search for zero	Unsigned 32	RW	N	1000	Speed during the movement out of the switch in user-defined units / s.

Homing acceleration

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
609A	00	Homing acceleration	Unsigned 32	RW	N	1000000	Acceleration during the reference run in user-defined units / s².

Home offset

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
607C	00	Homing offset	Signed 32	RW	N	100	Reference offset in user-defined units.

The reference offset is the distance from the reference point of the zero position (e.g. switching point to the limit switch) to the zero position.

The home offset is the distance from the reference point of the cero position (for example the switching point of the limit switch or the index signal of the encoder) to the cero position. In case of a positive value, the axis will go further away from the switching point. If the value is negative, the axis moves in the direction of the switching point. A negative home offset is only useful when the switching point is the index signal.

Remote control of homing

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6510	14	Homing Remoteable	Unsigned 8	RW	N	0	remote-mode of homing 0 : not active 1 : cctive
6510	15	Home Offset Enable	Unsigned 8	RW	N	1	home offset enable/disable 0 : disable 1 : enable

The "Homing Remoteable" parameter allows you to switch the remote mode of the reference run on or off for a gantry axis. You can use the "Home Offset Enable" parameter to activate or deactivate movement of the reference distance. It is a tool for starting up a gantry axis in remote mode (see section "Commissioning of reference run of a gantry axis" on page 77).

Path limitation of homing

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6510	16	Homing Switch Search Path (udu)	Unsigned 32	RW	N	0	Maximum way to search the homing switch 0 : no monitoring otherwise : search path
6510	17	Homing Switch Leave Path (udu)	Unsigned 32	RW	N	0	Maximum way to leave the homing switch 0 : no monitoring otherwise : leave path

In the first step of the homing, the axis is driven to search for the homing switch. The activation of the switch leads to the end of the first step and the motion reversal. In this second homing step, the axis drives out of the switch. A defective switch in these two steps can cause a change in the switching states of switch to go undetected. The homing trip then becomes an endless movement or at least a movement up to the hard mechanical stop. To prevent it, you can use the maximum motion path (object "Homing Switch Search Path (udu)" – sub-index 0x16) for searching the switch and the maximum movement path (object "Homing Switch Leave Path (udu)" – Subindex 0x17) for driving out the switch.

When searching for the switch, the path starts with the starting point of the homing journey and ends

- when the switch is activated (single- or gantry-axis in the standard mode of the homing – see section "Commissioning a gantry axis in standard mode" on page 79).
- when reversing the movement (gantry-axis with homing in remote mode – see section "Commissioning a gantry axis in the remote mode" on page 81). Please note here that the

homing movement only reverses when the switch, which is in the current movement direction further located, becomes active.

Because the homing can start from any point, you need to define this search path in such a way that the reference switches can always be found within this search route. Otherwise, you'll get the error code 43 if the switches are unfortunately outside the search route. At a value of 0 in this object, the search path is not controlled.

When driving out of the reference switch, the path begins with the reversal point of the movement and ends

- when the switch becomes inactive (single- or gantry-axis in the standard mode of homing – see section "Commissioning a gantry axis in standard mode" on page 79).
- when the switch, which is in the current movement direction further located, becomes inactive (gantry-axis with the homing in remote mode – see section "Commissioning a gantry axis in the remote mode" on page 81).

The inactivation of the switch must take place within the path of motion that you set in the "Homing Switch Leave Path (udu)" object. Otherwise you will get the error 43. At a value of 0 in this object, the search path is not controlled. For a homing with index signal (homing method 1, 2, 3, 6 – see section the "Homing method" on page 121), this monitoring ends before driving the index offset. The index signal is indeed monitored. It does not have to do with these two objects. Within 1.5 revolutions after driving the index offset, the index signal must occur. Otherwise, the homing trip will be canceled.

Controlling movement lengths during the homing is a sensible thing. A value 0 in the corresponding objects switches off the respective monitoring during the homing trip. A value unequal 0 activates monitoring. There are, however, two exceptions to consider here.

- when homing to "current position" (homing method 1 – see section "Homing method" on page 121), these two objects are ignored.
- When homing to index signal (homing method 32 and 33 – see section "Homing method" on page 121), these two objects are also ignored. However, the monitoring of the homing takes place anyway. If the index signal does not occur after 1.5 revolutions, the reference drive is canceled.

Moving the axis out of a limit switch – Leaving limit switch

If a limit switch of the axis is triggered, the drive module switches to the fault state and the final stage cannot be released, until the limit switch is active.

The “Moving out of a limit switch” mode of operation has been designed in order to move the axis out of a limit switch (see “Mode of operation” on page 112). For this, output 2 is set with the activation of this mode of operation; it can be used for the shunting of the limit switches of this axis. Afterwards, the module can be switched to the Operation Enable state; by setting the Bit 4 in the control word, the movement out of the limit switch is launched.

Direction of movement	negative in case of an activated positive limit switch
	positive in case of an negative positive limit switch
Velocity	reference run velocity (out of the switch)
Acceleration	reference run acceleration
Target position	switching point of the limit switch + reference point offset

Control word Bit 4	Meaning
0	No movement.
1	The movement out of the limit switch is carried out, if a limit switch is activated.

Factor Group – Conversion factors

The objects in the Factor Group are required for the conversion of the position, velocity and acceleration value from user-defined units (e.g. µm) into internal units (increments).

The only object which really is required for the conversion is the conversion factor (6093_h). All length, speed and acceleration data that are given in user-defined units are multiplied by this factor. For this, the units for the velocity and acceleration are “user-defined unit / second” or “user-defined unit / second²").

The conversion factor is calculated from the other objects according to the formula described under “Motor and Transmission dialogue box” on page 62.

The parameters dimensions index and unit index do not affect the conversion factors, but only serve for information purposes about the units. The ACSetup setting programme can present the right units e.g. for all length and velocity data.

Units index

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6089	00	Position notation index	Signed 8	RW	Y	0	Index for positioning unit.

By means of the units index, the unit to be used can be set. The following values can be chosen:

Units index	Meaning
0	Not specified.
-3	Milli (metre)
-6	Micro (metre)
75	Arc seconds
76	Arc minutes
77	Degree

Dimensions index

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
608A	00	Position dimension index	Unsigned 8	RW	Y	1	Index for the dimensions of the position.

The dimensions index indicates which physical dimension is to be used. The following values can be chosen:

Dimensions index	Meaning
0	Not specified (1)
1	Length
12	Angular measurement

(1) Here, “increments” is used as user-defined unit. A superordinate control must then convert the unit of length or angular measurement into increments or vice versa by means of the 0x6093 object. Thus, the control or drive module can correctly analyse the corresponding data.

Incremental encoder resolution

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
608F		Position encoder resolution	RECORD				Resolution of the incremental encoder.
608F	00	Number of entries	Unsigned 8	CONST	N	2	Number of entries.
608F	01	Encoder increments	Unsigned 32	RW	N	4000	Number of encoder impulses (4 times the value of the line number).
608F	02	Motor revolutions	Unsigned 32	RO	N	1	Per number of motor revolutions.

As the encoder resolution, 4 times the value of the line number of the encoder has to be set, since the resolution is increased by the 4-edge evaluation. All internal calculations refer to the quadruple resolution. The number of motor revolutions is firmly set to 1.

Transmission ratio

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6091		Transmission ratio	RECORD				Transmission ratio.
6091	00	Number of entries	Unsigned 8	CONST	N	2	Number of entries.
6091	01	Motor revolutions	Unsigned 32	RW	N	1	Revolutions at the transmission input.
6091	02	Shaft revolutions	Unsigned 32	RW	N	1	Revolutions at the transmission output.

If a transmission exists between the motor and the drive axis, the transmission ratio factor can be set in this object.

Feed forward constant

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6092		Feed forward constant	RECORD				Feed forward constant.
6092	00	Number of entries	Unsigned 8	CONST	N	2	Number of Entries.
6092	01	Feed	Unsigned 32	RW	N	5000	Feed.
6092	02	Shaft revolutions	Unsigned 32	RO	N	1	Per revolution.

The forward feed constant defines how many user-defined units (e.g. µm) are covered per drive axis revolution.

Conversion factor

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
6093		Position factor	RECORD				Conversion factor for position units (is also used for velocity and acceleration).
6093	00	Number of entries	Unsigned 8	CONST	N	2	Number of Entries.
6093	01	Numerator	Unsigned 32	RW	N	4	Numerator.
6093	02	Divisor	Unsigned 32	RW	N	5	Divisor.

Motor parameters

Here are all the objects that contain the motor parameters

Number of motor poles

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
604D	00	Motor Pole Number	Unsigned 8	RW	N	4	Number of motor poles

Motor Data

Index	Sub	Name	Typ	Attrib	Map	Default value	Meaning
6410		Motor Data	RECORD				General parameters of the connected motor
6410	00	Number of Entries	Unsigned 8	CONST	N	12	
6410	01	Hall_CBA_001 Position	Unsigned 16	RW	N	0	Position [Inkr.] for CBA_Halls_Condition 001
6410	02	Hall_CBA_010 Position	Unsigned 16	RW	N	0	Position [Inkr.] for CBA_Halls_Condition 010
6410	03	Hall_CBA_011 Position	Unsigned 16	RW	N	0	Position [Inkr.] for CBA_Halls_Condition 011
6410	04	Hall_CBA_100 Position	Unsigned 16	RW	N	0	Position [Inkr.] for CBA_Halls_Condition 100
6410	05	Hall_CBA_101 Position	Unsigned 16	RW	N	0	Position [Inkr.] for CBA_Halls_Condition 101
6410	06	Hall_CBA_110 Position	Unsigned 16	RW	N	0	Position [Inkr.] for CBA_Halls_Condition 110
6410	07	Encoder Index Position	Unsigned 16	RW	N	0	Position [Inkr.] of encoder index signal
6410	08	Encoder Index Emulating	Unsigned 8	RW	N	0	Activation of emulation of encoder index signal
6410	09	Voltage Constant (V/Krpm)	Unsigned 16	RW	N	50	Voltage constant (Volt / (1000 rpm))
6410	0A	Rated Speed (rpm)	Unsigned 32	RW	N	3000	Rated speed (rpm)
6410	0B	Winding Inductance (mikro Henry)	Unsigned 32	RW	N	1000	Phase inductance in micro Henry
6410	0C	Winding Resistance (mOhm)	Unsigned 32	RW	N	1000	Phase resistor in mOhm
6410	0D	Motor Type	Unsigned 8	RW	N	0	0 = rotation motor 1 = linear motor
6410	0E	Pitch	Unsigned 32	RW	N	31008	Pitch of the linear motor
6410	0F	Emulating Encoder Index Safety Distance	Unsigned 16	RW	N	250	Emulating Encoder Index Safety Distance
6410	10	Motor Temperatur Sensor	Unsigned 8	RW	N	0	1 = Motor temperature sensor active 0 = Motor temperature sensor not active
6410	11	Invert Motor Temperature Input	Unsigned 8	RW	N	0	1 = Motor temperature sensor input inverted 0 = Motor temperature sensor input not inverted
6410	12	Motor Temperature Input	Unsigned 8	RO	N	0	State of motor temperature sensor input
6410	13	Hall Check	Unsigned 8	RW	N	1	Hall inputs monitoring 0 = disabled 1 = enabled

The motor specific parameters stand in this object. These parameters can be taken out of the motor data sheet of the manufacturer. The three hall sensors according to the 3 motor phases delivers directly after switch-on the information in which of the six magnetic field conditions the motor is located. With this raw position information the motor can turn one revolution, until the first index signal comes. At this point of time the motor can be commutate precisely. At linear drives it is often the case, that the used length measuring system delivers no index signal. For this case the option "encoder index emulation" must be used to generate the index signal out of the hall signals. In the subindexes from 01 to 06 the positions of the hall conditions must be input. The position of the index signal stands in subindex 07. In case that the hall positions and index position cannot be find in the motor data sheet it is possible to measure them. But this is a very complicated. Communicate with us. We will help you to determine the hall sensor and index signal positions. In the subindex 09 ... 0C the user must input the values of the motor data sheet. The used units are especially taken into consideration.

Power stage parameter

By adapting the objects mentioned here you can adjust the operating characteristics of the power amplifiers to your application.

Axis direction

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
607E	00	Polarity	Unsigned 8	RW	N	0x00	Axis direction (0x00: positive, 0xC0: negative).

The Axis direction parameter makes it possible to invert the rotation direction of the motor at identical target values.

Specific parameters of the final power stage

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
60F7		Power stage parameters	RECORD				Specific parameters of the final power stage.
60F7	00	Number of entries	Unsigned 8	CONST	N	1	Number of Entries.
60F7	01	Limit switch active: disable current	Unsigned 8	RW	N	0	Behaviour of the current in case of an activated limit switch.

Normally, the current is switched off by the hardware, if a limit switch is activated. By means of the 0x2054 object (“Limit switch shunt"), the safety circuit can be shunted in order to be able to switch on the current again. The motor can then be moved in both directions. There are, however, applications in which it is desired that the movement in the active direction of the limit switch is blocked. In this case, the parameter “Limit switch active: disable current” must be set to 1; otherwise movements in both directions are still admissible.

Parameter value	Meaning in case of an activated limit switch
0	The movement in both directions is possible.
1	Only the movement out of the active limit switch is admissible.

Drive Data

Index	Sub	Name	Typ	Attrib	Map	Default value	Meaning
6510		Drive Data	RECORD				General parameters of the drive module
6510	00	Number of Entries	Unsigned 8	CONST	N	3	
6510	01	Digital Input1 Configuration	Unsigned 8	RW	N	0	Configuration of the digital input 1
6510	02	Digital Input2 Configuration	Unsigned 8	RW	N	1	Configuration of the digital input 2
6510	03	Digital Input3 Configuration (dummy)	Unsigned 8	RW	N	2	Configuration of the digital input 3
6510	04	Digital Input4 Configuration	Unsigned 8	RW	N	3	Configuration of the digital input 1
6510	05	Invert Digital Inputs	Unsigned 8	RW	N	0x00	Evaluation of the digital inputs (high- / low-active)
6510	06	IIt-Currentlimit Time	Unsigned 16	RW	N	500	I²t-current limitation time
6510	07	Mode of Set Bracke	Unsigned 8	RW	N	0	Mode of operation for the control of the brake 0: manual 1: automatic
6510	08	Init Value of Digital Outputs	Unsigned 8	RW	N	0x00	Initialisation values of the digital outputs
6510	09	Invert Digital Outputs	Unsigned 8	RW	N	0x01	Evaluation of the digital outputs (high- / low-active)
6510	0A	Hardware Type	Unsigned 8	RO	N	0x00	Hardware type IMD20 or IMD40 0 = Imd20, 1=Imd40
6510	0B	Standstill Tolerance Window	Unsigned 32	RW	N	10000	Position tolerance of the standstill supervision
6510	0C	Actual Standstill State	Unsigned 8	RO	N	0	Actual Standstill State
6510	0D	Home Index Offset	Unsigned 32	RW	N	0	Home Index Offset (User defined units)
6510	0E	Phase Current Hard Limit	Unsigned 8	RW	N	100	Motor phase current limitation in % von 13 A at IMD40 or 25A at IMD20
6510	0F	DC Bus Voltage Low Limit	Unsigned 16	RW	N	180	DC Bus Voltage Low Limit in volt
6510	10	DC Bus Voltage Over Limit	Unsigned 16	RW	N	380	DC Bus Voltage Over Limit in volt
6510	11	Max Current Extension	Unsigned 8	RW	N	0	Maximum possible motor current when using IMD40 0 : 8000mA 1 : 13000mA

(Drive-Data – continuation)

Index	Sub	Name	Typ	Attrib	Map	Defaultwert	Bedeutung
6510	12	Digital Input5 Configuration	Unsigned 8	RW	N	2	Cofiguration of digital input 5 0 : motor temperatur 1 : free use 2 : unused
6510	13	Reset value of Digital Outputs	Unsigned 8	RW	N	0x01	Initial value at software reset 0 : unchanged 1 : like when switching on
6510	14	Homing Remoteable	Unsigned 8	RW	N	0	remote-mode of homing 0 : not active 1 : cctive
6510	15	Home Offset Enable	Unsigned 8	RW	N	1	home offset enable/disable 0 : disable 1 : enable
6510	16	Homing Switch Search Path (udu)	Unsigned 32	RW	N	0	Maximum way to search the homing switch 0 : no monitoring otherwise : search path
6510	17	Homing Switch Leave Path (udu)	Unsigned 32	RW	N	0	Maximum way to leave the homing switch 0 : no monitoring otherwise : leave path

Via the parameter "Digital InputX Configuration" it is defined on which bit (0 ... 3) in the “Digital Inputs“ parameter, input X is illustrated.

The parameter "Invert digital inputs" defines, if the inputs are to be evaluated low-actively or high-actively.

Invert digital inputs		Meaning
Bit 0 … 3	0	Input 1 ... 4 is high-active (i.e. no signal inversion).
	1	Input 1 ... 4 is low-active (i.e. signal inversion).

The meaning of the bits is internally allocated as follows:

Mode of operation	Parameter value
Bit 0	Negative limit switch
Bit 1	Positive limit switch
Bit 3	Enabling signal

The “Mode of brake control” determines how the brake is controlled via output 3. It has to be taken into account that the 0x60FE “Digital outputs” object is currently used only for output 3. This, however, can still be changed in the future.

In the manual mode, the brake has to be set or reset via the “Digital Outputs” parameter by means of SDO. The first bit of the “Init. value of digital outputs” object can be defined as the initial value of output 3.

In the automatic mode, the “Digital Output” parameter is blocked for a change from outside. The “Init. value of digital outputs” parameter does not have a meaning in this mode. The brake control is carried out internally. In the dead status, the brake is automatically set and in the live status, the brake is also automatically reset.

Parameter value	Mode of the brake control
0	Manual
1	Automatic

The "Invert digital outputs" parameter defines, if output 3 is to be evaluated low-actively or high-actively. This parameter is blocked for the automatic mode.

Invert Digital Outputs		Meaning
Bit 0	0	Output 3 is high-active (i.e. no signal inversion): Value 0 or 1 to Bit 0 from the object 0x60FE → value 0 or 1 at output 3)
	1	Output 3 is low-active (i.e. signal inversion): Value 0 or 1 to Bit 0 from the object 0x60FE → value 1 or 0 at output 3)

The different parameters of the brake control can very easily be defined and tested with the ACSetup programme in the “Brake dialogue box” on page 59.

The standstill monitoring was implemented to monitor the movement condition of the motor. If the motor changes from the condition "movement" into the condition "silence", the actual position is recorded. A accordant signal is given out to notify that the motor is in condition "silence". If the motor leaves the tolerance range of the previous recorded position another signal is given out to notify that the motor is in condition "movement". This tolerance range can be defined in the parameter "Standstill Tolerance Window". The standstill monitoring works in all movement operation modes.

With the parameter "Phase Current Hard Limit" the user can define a certain percentage of the maximal hardware limit phase current. During the operation all motor phase currents are measured permanently and are compared with this limit value. An exceeding of this value leads to deactivation of the controlling until the motor phase currents decrease under the limit value. Take into consideration that the IMD20/IMD40 is not switched off during this measure. It is only a security measure.

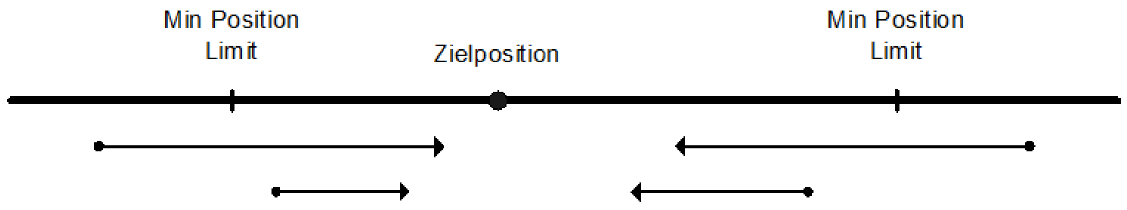
Limits of movement

Software limit switches

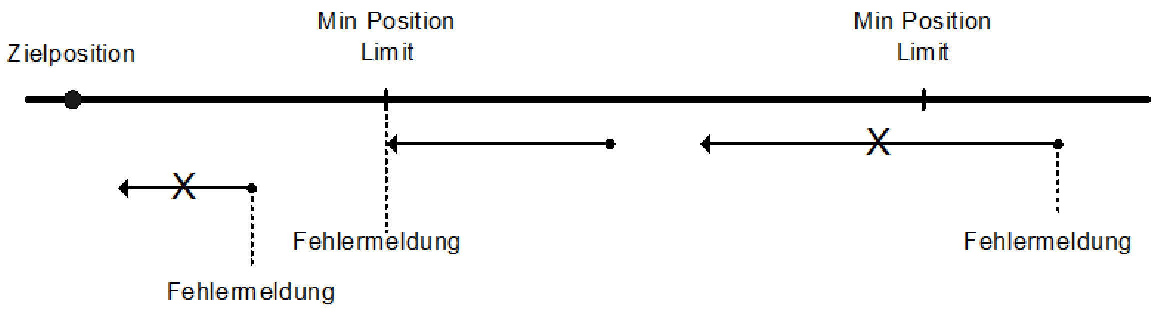
Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
607D		Software Position Limits (udu)	RECORD				Freelly definable movement range
607D	00	Number of Entries	Unsigned 8	CONST	N	2	Number of entries
607D	01	Min Position Limit	Signed 32	RW	N	-0x7FFFFFFF	Movement limit [user unit] in the negative direction
607D	02	Max Position Limit	Signed 32	RW	N	0x7FFFFFFF	Movement limit [user-unit] in the positive direction

The software limit switches are only monitored in the operation mode “Profile Position Mode“ and only when the monitoring is activated by the setting of the object 2058. During an active movement in the mode “Profile Position Mode“, setting the software limit switches is not allowed. The following points must be followed in relation to the movement range.

- Each movement to a target position within the movement range is allowed.



- A target position outside the movement range only allows movements from a position within the movement range up to the relevant limit. If the limit is reached, the bit 11 of Statusword is set. Other movements are not permitted. The bit 11 is set here immediately at the start of the movement. The following image shows only one example with target positions to the left of the movement range. The same applies of course to target positions to the right of the movement range.



- Unlike in the past, if a software limit switch is reached, the power element is not switched off and the output stage does not go into error state. The motor remains unpowered at the limit. Bit 11 of the Statusword is set to notify the software limit switch violation.
- When setting the software limit switches, the output stage does not check if the values are reasonable or not. I.e. the user can e.g. set the negative limit switch to the right of the positive limit switch. The error is not noticed until the start of a movement. In other words, the software limit switches are only monitored during an active movement.

Maximum speed

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
607F	00	Max. profile speed in user defined unit / s	Unsigned 32	RW	N	100000	Maximum speed in user defined unit / s.

The maximum speed internally restricts all velocities in the modes of operation “Profile velocity mode” and “Profile position mode”. Furthermore, it indicates the range which is available as target value range in the mode of operation “Velocity controller with analogue input” (-10 ... +10 V = -Vmax ,.. +Vmax)

Maximum acceleration

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
60C5	00	Max. Acceleration in user defined unit /s²	Unsigned 32	RW	N	10000000	Maximum acceleration in user defined unit /s.

The maximum acceleration internally restricts all acceleration and braking ramps in the modes of operation "Profile Velocity Mode" and "Profile Position Mode", even if the acceleration parameter (6083_h) has a greater value.

Digital inputs and outputs

Digital inputs

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
60FD	00	Digital inputs	Unsigned 8	RO	Y	0x00	Current status of the digital inputs.

The first four bits contain the current status of the four digital inputs. The allocation of the inputs to the bits as well as the active range of the inputs is set by means of the 6510_h 01_h - 05_h parameters.

The meaning of the bits is allocated as follows:

Bit number	Meaning
Bit 0	Negative limit switch
Bit 1	Positive limit switch
Bit 3	Enabling signal

Digital outputs

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
60FE	00	Digital outputs	Unsigned 8	RW	Y	0x00	Actual state of the digital outputs.

With the help of the first bit of the object 0x60FE, output 3 can be set or reset for the brake control. Currently, only output 3 is shown on the object "Digital outputs".

The meaning of the bits is internally allocated as follows:

Bit number	Meaning
Bit 0	Output 3 for the brake control
Bit 1 ... 7	Not yet used

Manufacturer-specific parameters

In the range of 2000_h to $5FFF_h$ of the object directory, there are those parameters that are not predefined by a CanOpen specification.

Can transmission speed

With these objects, you have the possibility to set different transmission speed of the CAN bus software.

CAN Baud Rate

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
2001		Can Baud Rate	RECORD				Displaying and changing of the der CAN-Bus baud rate in the modul
2001	00	Number of Entries	Unsigned 8	CONST	N	3	Number of entries
2001	01	Actual Baud Rate	Unsigned 8	RO	N		Current Baud Rate Code
2001	02	Actual High Speed Submode	Unsigned 8	RO	N		Current Baud Rate Code in High Speed Submode
2001	03	New High Speed Submode	Unsigned 8	RW	N	0	New Baud Rate Code in High Speed Submode

If the DIP switch 5 (responsible for the baud rate) is in position "0" (Low Speed Mode) the can bus has a baud rate about 20 kBd. In the position "1" of the DIP switch 5 (High Speed Mode) the user can choose with the help of the parameter "New High Speed Submode" a different transfer baud rate. The new baud rate is only active after a "switch-off" and "switch-on" of the module.

With a double click on the object 2001h-03 "New High Speed Submode" a new baud rate for the can bus can be chosen by an input dialog.

The following baud rate code is used in the object Can Baud Rate:

Baud Rate Code	CAN-Bus Baud Rate
0	1000 kBit/s
1	800 kBit/s (not supported)
2	500 kBit/s
3	250 kBit/s
4	125 kBit/s
5	Reserviert (not supported)
6	50 kBit/s
7	20 kBit/s
8	10 kBit/s (not supported)

The shorter the baud rate is, the longer can be the can bus cable and the more sensitive compared to disturbance is the data transfer on the can bus.

You cannot chose the baud rate as short as you want. In interpolation mode with four motors a baud rate of at least 125KBd is necessary. If a gantry axis is in operation, a baud rate of at least 250kBd is necessary.

Relation between Baud rate and wire length of CAN bus:

Baud Rate (kBit/s)	20	50	125	250	500	1000
Wire length (m)	2500	1000	500	250	100	25

Write protect the set parameters

Password

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
2042		Password	RECORD				Change of password
2042	00	Number of entries	Unsigned 8	CONST	N	3	Number of Entries.
2042	01	Current password	Unsigned 32	WO	N		Current password
2042	02	Modification mode	Unsigned 8	RO	N		0 → Wrong password 1 → Correct password
2042	03	New password	Unsigned 32	WO	N		New password

The “Password” object offers the possibility to protect the set parameters from unauthorised changes. It has to be taken into account that the protection mechanism does not work on the IMD20, IMD40 level. An application software has to use the possibilities offered by this object in order to protect the parameters within the IMD20, IMD40 module.

During the initialisation phase, the “Modification mode” parameter is always set to 0. The writing of the “Current password” parameter with the correct password sets the “Modification mode” parameter to 1. In this case, a new password can be defined via the "New password" parameter. A wrong password sets back the "Modification mode” parameter to 0. The setting of a new password is thus prohibited. Apart from the setting of a new password, an application software can control by means of the “Modification mode” parameter, if the password entered is correct or not, and respond accordingly. After a firmware update via the bootstrap loader, the module always has the default password “ISEL”.

(See “Password” on page 48).

Synchronous control - Gantry Axis

Using these objects, the the gantry axis of Can cnc control is implemented.

Synchronous control

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
2043		Synchronous control	RECORD				Control of the gantry axis.
2043	00	Number of entries	Unsigned 8	CONST	N	3	Number of entries.
2043	01	Synchronous mode	Unsigned 8	RW	N	0	0 → No gantry axis 1 → Slave axis 2 → Master axis
2043	02	Slave minimal node ID	Unsigned 8	RW	N	127	Minimum CAN ID no. of the slave axis.
2043	03	Slave maximal node ID	Unsigned 8	RW	N	127	Maximum CAN ID no. of the slave axis.
2043	04	Slave error	Unsigned 8	RW	N	0	0 → Fault-free slave 1 → Faulty slave
2043	05	Master status word	Unsigned 16	RW	Y	0x0027	Master axis status word.
2043	06	Master: current position	Signed 32	RW	Y	0	Current position of the master axis.
2043	07	Following error window	Unsigned 32	RW	N	10000	Position window for the monitoring of the gantry axis.
2043	08	Following error	Signed 32	RO	Y	0	Current deviation of position between the master and the slave axis.
2043	09	Auxiliary variable	signed 32	RW	N	0	Auxiliary variable for the synchronisation process.
2043	0A	Max. following error	signed 32	RW	N	0	Max. deviation of position between master and slave.
2043	0B	Coupling factor	unsigned 8	RW	N	4	Coupling factor with master axis.
2043	0C	Homing Follow Error Check (Dummy)	Unsigned 8	RW	N	1	Homing Follow Error Check (Dummy)
2043	0D	Homing Following Error Windows (udu)	Unsigned 32	RW	N	10000	Positionsfenster für die Überwachung der Gantry-Achse während der Referenzfahrt
2043	0E	Homing Search For Switch	Unsigned 8	RW	N	1	Referenzschalter-Suchen 0 : Aus 1 : Ein
2043	0F	Homing Reverse Enable	Unsigned 8	RW	Y	1	Umkehr der Referenzfahrt 0 : Sperren 1 : Freilassen
2043	10	Homing Zero Enable	Unsigned 8	RW	Y	1	Anhalten im Nullpunkt am Referenzfahrt-Ende 0 : Ja 1 : Nein
2043	11	Homing Zero Position (udu)	Interger 32	RO	N	0	Abstand [udu] zwischen Startpunkt und Endpunkt der Referenzfahrt
2043	12	Distance Switch To Index (inc)	Unsigned 32	RO	N	0	Abstand [inc] zwischen Schalter und Indexsignal

With the help of the synchronous control objects given above, the ISEL CNC control realises the operation of the gantry axis. The “Synchronous Mode” object is set during the CNC operation by the control, if applicable. Currently, up to 2 gantry axes are allowed per machine. The CAN node ID of the slave axes are stored in the “Slave minimal node ID” and “Slave maximal node ID” for the monitoring of errors. The “Slave Error” object is only used by one master axis. If the related slave axis is faulty, this object is set in the master. In a slave axis, this object is of no importance. During the gantry, the slave axis receives the status as well as the current position of the master axis in the objects “Master: status word” und “Master: current position”. By means of these two objects, the slave axis recognises, if the master axis is faulty or not and if the deviation of position between the master and the slave exceeds the limit value in the “Following error window” object or not. The current value of the deviation of position between the master and slave can be seen in the “Following error” object. The “Auxiliary variable” sub-index is intended for internal use only. The “Max. following error” sub-index of the slave axis is an auxiliary means for the commissioning. The user can launch the ACSetup in the declaration mode and this variable indicates the maximum deviation of position between the master and the slave. The seven objects mentioned last are used only by one slave axis. In the master operation, they are not taken into account.

With the exception of the two objects “Following error window” and “Coupling factor”, all objects of the synchronous control are commissioned during operation by the ISEL CNC control. In the “Can Interpolation” dialogue box of the slave axis, the position error of a gantry axis can be defined as a value for the “Following error window” object and the coupling factor of the slave as the value of the “Coupling factor” object. It has to be taken into account that these two objects of a slave axis, and not of a master axis, have to be set.

When a higher-level controller sends the command "execute reference run", the output stage executes the reference run automatically. The various phases of reference run, such as "moving into the switch", "move out of the switch", "search for the index signal",... are performed internally in the output stage. Then the higher-level controller receives the feedback signal "reference run completed" from the output stage. This independent procedure is optimal for a single axis. But with a gantry axis, where both axes must always run synchronously, the two independent reference runs can lead to serious problems in the synchronisation of the two axes. To solve this problem, we have introduced the 3 objects "Homing Search For Switch", "Homing Reverse Enable" and "Homing Zero Enable". With these objects, a higher-level controller can then coordinate the reference runs on both axes in the so-called remote mode. The exact procedure of a reference run in remote mode can be found in the section “Commissioning of reference run of a gantry axis” on page 77.

A reference run normally starts with a movement in the direction of the reference switch to search for it. The value 1 in the "Homing Search For Switch" object tells the output stage to search for the switch, which is normally the case. With the value 0 in this object, you can tell the output stage that the switch does not necessarily have to be found. Then how does it fit together? This object is used for the reference run of a gantry axis. Both the master axis and the slave axis have a reference switch. It is not necessary to find both switches when driving to the switch. It is enough to find the switch at the back in the direction of movement. The position of the switch at the front is determined when moving out. The special advantage is where the two switches are located very far apart. If the gantry axis accidentally takes up a position between the two switches when starting the reference run, the search for the two switches can never be successful. Therefore, it makes sense to use this object to restrict the search to the switch at the back.

Normally, the reference run reverses the movement immediately after the switch has been found. If the switches on the two axes are found at different times, this can lead to a crash. The value 0 in the "Homing Reverse Enable" object can be used to delay the reversal of the reference run until the switches on both axes have been found. By writing the value 1 into this object of the two output stages, the reversal of the master reference run and the reversal of the slave reference run can be started at the same time.

After the motor has moved out of the switch or the index signal has been found, the reference distance is approached. And at the end of the reference distance zeroing takes place. This is the zero point and the motor stops here. And similar to above, the same problem can occur with a gantry axis. To prevent this, you can use the "Homing Zero Enable" object. A value of 0 prevents one motor from stopping at the zero point while the other motor is still searching for the index signal, for example. A higher-level controller with the value 1 in this object only allows resetting on both axes if the two axes have already started moving to the respective reference distances.

With the reference run start position as reference point, the output stage remembers the position of the reference switch in the "Homing Zero Position" object during reference run and possibly also the position of an index signal in the "Distance Switch To Index" object. During commissioning, the ACSetup programme can then use the "Homing Zero Position" object from the two axes to calculate the reference distances (Object 607C). The "Distance Switch To Index" object can then be used to calculate the index offset (Object 6510_0D).

Online distance control

These objects are used for distance control in real time.

Online distance control

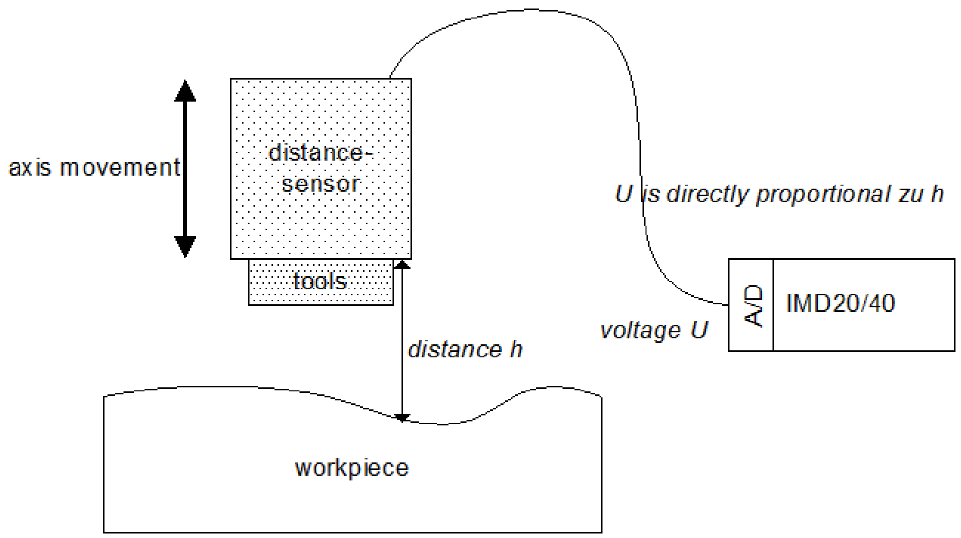
Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
2055		Distance Regulator	RECORD				Online distance control
2055	00	Number of Entries	Unsigned 8	CONST	N	22	Number of entries
2055	01	Regulator Mode	Unsigned 8	RW	N	0	0: distance control on a single axis 1: main axis of the distance control in a gantry axis 2: Minor axis of the distance control in a gantry axis
2055	02	A/D Resolution	Unsigned 8	RO	N	11	Resolution of the A/D converter
2055	03	A/D Absolute Range (mV)	Unsigned 16	RO	N	10000	±Voltage range of the A/D converter (mV)
2055	04	A/D Actual Input (digit)	Signed 16	RO	Y	0	Current input value of the A/D-converter (digit)
2055	05	A/D Desired Input (digit)	Signed 16	RW	N	0	Target value for the input of the A/D- converter (digit)
2055	06	A/D Desired Input Deadband (digit)	Unsigned 16	RW	N	20	Zero range around the target value of the input from the A/D-converter (digit)
2055	07	A/D Max Negative Deviation (digit)	Unsigned 16	RW	N	200	Maximum deviation in the negative direction (digit)
2055	08	A/D Max Positive Deviation (digit)	Unsigned 16	RW	N	200	Maximum deviation in the positive direction (digit)
2055	09	Distance Factor Numerator (udu)	Interger 16	RW	N	1	Numerator of the conversion factor between the length of movement [udu] and A/D input voltage [digit]
2055	0A	Distance Factor Divisor (digit)	Interger 16	RW	N	1	Divisor of the conversion factor between the length of movement [udu] and A/D input voltage [digit]
2055	0B	Regulator Time (100µs)	Unsigned 16	RW	N	100	Settling time in [100µs] for the maximum deviation
2055	0C	A/D Conversion Control	Unsigned 8	RW	Y	0	0: Switching off the A/D-converter 1: Switching on the A/D-connverter
2055	0D	A/D Conversion Status	Unsigned 8	RO	Y	0	0: A/D-converter not active 1: A/D-converter active
2055	0E	Total Deviation (udu)	Unsigned 32	RW	N	127	Maximum deviation to be controlled

(Online distance control – continuation)

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
2055	0F	Scaled Master Deviation Unit	Unsigned 8	RW	N	1	Unit of the distance deviation to be transferred from the main to the minor axis in a gantry axis 0: inc (increment) 1: udu (user defined unit)
2055	10	Master Deviation Scale Factor	Unsigned 16	RW	N	1	Scaling factor for the distance deviation to be transferred from the main to the minor axis in a gantry axis
2055	11	Scaled Master Deviation Byte	Interger 8	RW	Y	0	Scaled distance deviation of the main axis rounded off to 1 byte (8 Bits)
2055	12	Scaled Master Deviation Word	Interger 16	RW	Y	0	Scaled distance deviation of the main axis rounded off to 1 Word (16 Bits)
2055	13	Scaled Master Deviation DWord	Interger 32	RW	Y	0	Scaled distance deviation of the main axis rounded off to 1 Double Word (32 Bits)
2055	14	Regulator Control	Unsigned 8	RW	Y	0	0: Switching off the distance control 1: Switching on the distance control
2055	15	Regulator Status	Unsigned 8	RO	Y	0	Status of the distance control 0: Inactive 1: Active
2055	16	Auxiliary Variable	Interger 8	RW	Y	0	Auxiliary variable

In many applications such as laser cutting, engraving, ... a constant distance between the tool and the component surface is decisive in terms of the machining quality. If the irregularities in the component surface already exist before processing and are also unchanged, they can be metrologically recorded prior to processing and then corrected during the machining process. But if the irregularities arise during machining, the measurements, including the correction, are necessary in real time. With the object "distance regulator" it is possible to regulate the distance between the components and set the component to a target value. In this case, it doesn't matter, when and how the irregularities occur.

This object uses the A/D converter on the module. In the analogue mode with ±10-Volt input signal, you can logically enough not use this object. In both modes with CAN bus or RS232, the A/D converter remains unused. The A/D converter is the input interface for a distance sensor, which measures the distance to be regulated and delivers a proportional analogue signal. If the distance is to be held to a target value and if the voltage equivalent to the target value point is known, the distance deviation can be calculated from the voltage deviation. Then the distance to the component can be adjusted by the output stage. The correction will take place even during a movement. I.e. the correction value is then automatically added to the actual movement data.



The online distance control requires that many parameters have to be set beforehand using the different sub-objects. Below, the meaning of the individual sub-objects and the approach when setting, are explained precisely.

In normal cases, the object "regulator mode" has the value 0. It means that the axle is a single axis and not a gantry axis. In this case, the module reads its own A/D converter and calculates the position deviation from this and corrects the distance. The values 1 and 2 are intended for a gantry axis. With a gantry axis, you can use either the A/D converter of the master axis or that of the slave axis. The axis, whose A/D converter is used, is then the main axis and the other axis is the minor axis. The main axis does not necessarily have to be the master axis of the gantry axis. In the main axis of a gantry axis the correction of the deviation is similar to that of a single-axis. The minor axis, whose A/D converter is not used, does not determine the distance deviation itself. It is given the deviation. With the objects yet to be explained " Scaled Master Deviation Byte " or "Scaled Master Deviation Word" or "Scaled Master Deviation DWord" the main axis provides the deviation determined outwards. Also through these objects you can provide the minor axis with the deviation. The minor axis reads the deviation and corrects its position accordingly. You can of course also configure in such a way that both the master and the slave axis calculate the deviations from the input values of the A/D converter and correct their positions independently of each other. In this case, the object "regulator mode" has the value 0 in both axes and you have to connect the distance sensor to the two output stages. In the Isel Can control, due to the internal software structure for a gantry axis, it is established that the master axis is also the main axis. The slave axis is the minor axis. In other words, the output stage of the master axis is the input interface of the distance sensor. It is emphasised here that the number of axes with the active online distance control is not restricted. The online distance control can be activated for any number of axes.

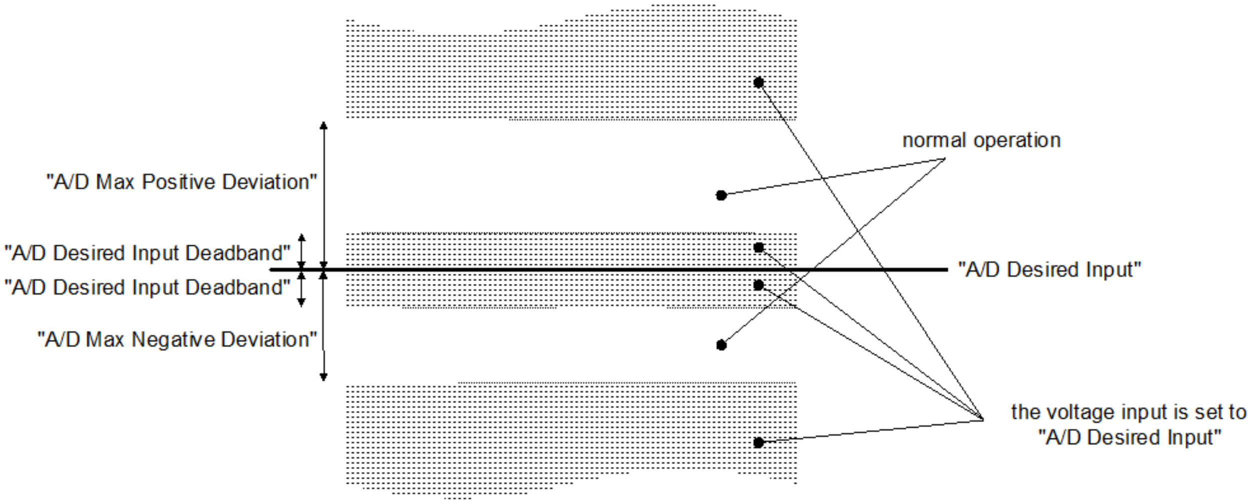
The object “A/D Resolution“ is a Read-Only object, the content of which delivers the resolution of the A/D converter. Currently you always receive the value 11 here. I.e. the A/D converter on the module has an 11-Bit resolution for the entire measuring area (negative … 0 … positive).

The object “A/D Absolute Range“ is a Read-Only object that gives the voltage range according to the amount in [mV] both in the negative and in the positive direction. The A/D converter currently has the measuring range ±10 Volt. Therefore the object returns the value 10000.

The object “A/D Actual Input“ is a Read-Only object, in which the current input value of the A/D converter or the voltage of the distance sensors stands. You should note that here you get the voltage in [digit], a resolution step of the A/D converter. In other words, in an 11-Bit-A/D converter, the value returned here is in the range (-1024 … 0 … 1024).

In the object “A/D Desired Input“, the target value in [digit] stands for the output voltage of the distance sensor. The primary control must set this target value. On the basis of this target value and the actual value of the A/D converter, the output stage can internally with the aid of the conversion factor from the two objects ”Distance Factor Numerator“ and ”Distance Factor Divisor“, calculate the distance deviation in length units. The position is then corrected on the basis of this deviation.

The object “A/D Desired Input Deadband“ defines the tolerance range around the target value (object ”A/D Desired Input“). As long as the voltage deviation is in this tolerance range, the actual value is set the same as the target value. This object can be used to influence the sensitivity of the distance control (see Image). Constant voltage fluctuation does not necessarily lead to a continuous adjustment.



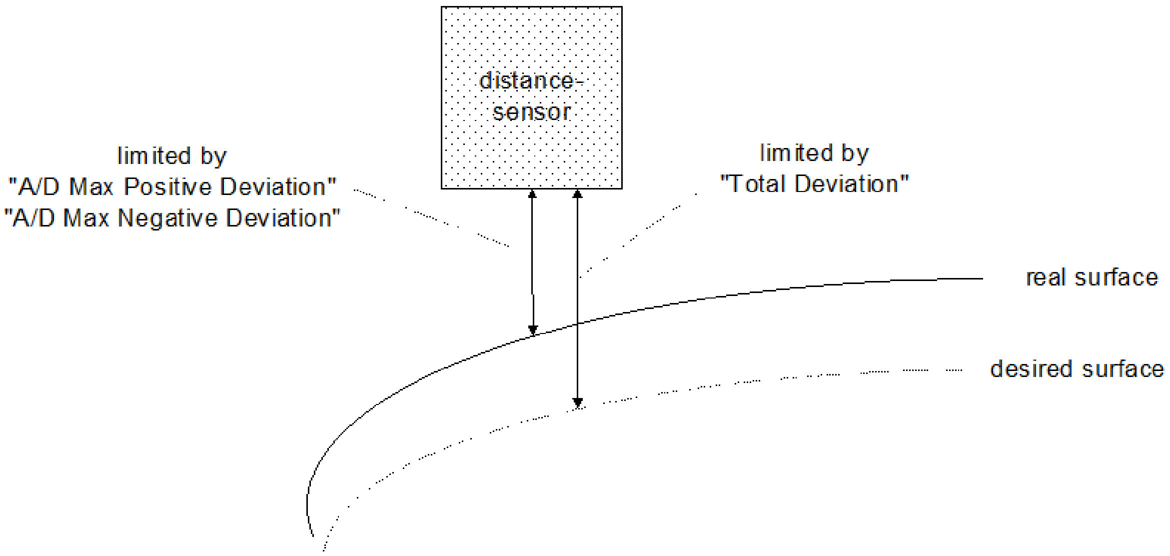
With the two objects “A/D Max Negative Deviation“ and “A/D Max Positive Deviation“, one defines the upper and the lower limit for the voltage deviation. If the deviation is outside this range, the actual value is set the same as the target value. In other words, the deviation is not evaluated internally (see Image). Using these two objects, you have the possibility of avoiding any sudden change in voltage. A sudden change in voltage may for example be caused by hardware errors, unfavourable component surface shape, … . In such cases, a strict position correction can be very dangerous.

You can input the conversion factor of the distance sensor via the two objects ”Distance Factor Numerator“ and ”Distance Factor Divisor“. The output stage uses this factor to calculate the position from the voltage.

In the object “Regulator Time”, you can establish how quickly a distance deviation should be adjusted. A deviation that corresponds to the larger value of the two objects ”A/D Max Negative Deviation“ and ”A/D Max Positive Deviation“, should be adjusted within the time defined here. A distance control that is too fast, can lead to coarse movements in the axis. Please note that the time defined here has the unit [100 µs].

With the object “A/D Conversion Control“, the A/D converter on the output stage can be switched on or off. The A/D converter cannot be controlled via this object if the output stage is in the analogue operating mode. The A/D converter can be rewad at any time via the object “A/D Actual Input“. Due to various delays, the A/D converter is not immediately ready for operation after switching on. It is therefore absolutely necessary to call up the status of the A/D converter via the object "A/D Conversion Status", before you start any other activities. The active A/D converter is essential for the distance control but you can also use the A/D converter for other purposes. The A/D converter may at any time be read via the object "A/D actual input".

The object “Total Deviation“ establishes how big the distance deviation to the target surface of the component is allowed to be as a maximum. Apart from another unit, there is another difference from the two objects ”A/D Max Negative Deviation“ and ”A/D Max Positive Deviation“. These two objects limit the current distance deviation to the real surface of the component. The object “Total Deviation“ limits the maximum deviation to the target surface. This limit is particularly important for a Gantry axis, where the distance deviation is transferred from one axis to the other via the CAN-Bus. A limit keeps the amount of data to be transferred small. This of course raises the question of how the distance sensor knows the target surface. Through direct measuring, the distance sensor only knows the real surface. It does not know the target surface. Only the output stage knows the target surface, as it has the movement data.



The next 5 objects are designed for distance control in a Gantry axis. The object “Scaled Master Deviation Unit“ establishes, in which unit the distance deviation to be transferred from the main axis to the minor axis should stand. Zero means that the distance deviation has the unit [inc]. In a unit defined by the user, this object has the value 1. The object “Master Deviation Scale Factor“ gives you the option of scaling the distance deviation to be transferred. This object is designed for the reduction of the data to be transferred via the bus. The greater the scaling factor, the lower the precision of the distance deviation. Both of these objects have to be set both in the main axis and in the minor axis. It goes without saying that the values of the same objects in the two axes must be the same. The main axis uses the two objects to calculate the den scaled value of the distance deviation. The minor axis uses these two objects to get back to the correct position deviation from the scaled value coming via the CAN-Bus. Although the main axis does not need to, its position deviation to be corrected in favour of the synchronicity in the gantry axis, is also gained from the internally calculated scaled value to have the same imprecision as that of the minor axis. Via the 3 objects “Scaled Master Deviation Byte“, “Scaled Master Deviation Word“ and “Master Deviation DWord”, the main axis provides the internally calculated distance deviation outwards and the minor axis also receives the deviation to be corrected via these 3 objects. Please note that only the first two objects named receive the scaled distance deviation. The third object “Master Deviation DWord“ is not subject to any scaling but for this object the unit set in the object “Scaled Master Deviation Unit“ still applies. A higher level control can itself set which for these 3 objects is used for the transfer of the distance deviation. The ISEL-CAN control only uses the object “Scaled Master Deviation Byte“. I.e. the scaled distance deviation is then in the range -127 … 0 … +127. One special feature of the value range must still be mentioned. The main axis uses

- the value -128 =0x80 in the first object
- the value – 32768 = 0x8000 in the second object and
- the value -2147483648 = 0x80000000 in the third object,

to inform the minor axis that the distance control in the main axis is no longer active. I.e. these values represent a control command. They are not a distance deviation.

The distance control is activated with the object “Regulator Control“. The switching on of the distance control is then only successful if the following conditions are fulfilled:

- In a single axis or in the main axis of a gantry axis, the A/D converter is already successfully activated beforehand. In the minor axis of a gantry axis, the operation of the A/D converter is not required.

- The operating mode “Interpolation“ is active (see object 6061 – “Modes of Operation Display“). I.e. the distance control only works in the operating mode “Interpolation“.

Please note that in a gantry axis, you have to activate the distance control both in the main axis and in the minor axis. The distance control is also deactivated with this object. An active distance control can also become inactive without this object in the following cases.

- The operating mode “Interpolation“ is switched off.
- The A/D converter of the single axis or of the main axis of a gantry axis is switched off.
- In the minor axis of a gantry axis, if the distance control of the main axis is no longer active.

To query whether the distance control is successfully switched on or off, or whether the distance control is still active or not, you use the Read-Only object “Regulator Status ".

The last object "Auxiliary variable" is an auxiliary variable for the data transfer between the major and the minor axis of a gantry axis.

As you can see, the distance control is a very complex and complicated story and using it in a higher level control is not exactly easy either. In our ISEI-CAN control we provide you with various Dll functions for the Windows operating system, to facilitate the use of the distance control. If it is possible, you should use these functions.

Reference point shifting – Homing point shifting

With these objects, you can move the zero point of the axis arbitrarily without having to do the reference move.

Shifting the homing point

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
2056		Change Home Position	RECORD				Shifting the reference point
2056	00	Number of Entries	Unsigned 8	CONST	N	2	Number of entries
2056	01	Change Control	Unsigned 8	RW	N	0	Flank 0→1: shift is carried out .
2056	02	New Home Relative Position (inc)	Signed 32	RW	N	0	Relative position of the new reference point in [inc]

In the operating mode "Profile Position Mode" a recurring movement with relative positions may cause the internal 32-bit position counter to overflow at some point. In practice, however, it only happens in a rotation axis. In a linear axis, one has enough room with a 32-bit position counter in normal cases. The position counter overflowing leads to the error 44. To prevent overflow, you have the option of regularly reducing the contents of the 32-bit position counter. The object "New Home Relative Position" first of all defines the desired shift. With the object "Change Control", you then perform the shift. Please note that the flank 0→1 is required here to start the action. The desired shift is not displayed in [user unit] but in [increment]. However, you can at any time easily make the conversion between [user] and [increment] using the conversion factor (object 6093 "Position Factor"). After a successful shift of the reference point, all relevant items in the module are reduced exactly by the desired amount. Changing the counter content logically leads to a shift in the reference point and all items read from the module of course also change. In a relative movement however, it does not matter. In a rotational axis, you can choose the shift so cleverly to have the shifted reference point always physically in the same position. Control software that reads out and displays the current positions, can convert the position read from the module to the extent position of the rotational axis and display it. As a result, the position shift is no longer noticeable.

The moving of the reference point is only permitted when the new positions, caused by the moving, are still within the authorised 32-Bit-numbers. In the three motion modes “Profile Position Mode“, “Interpolated Position Mode“ and “Profile Velocity Mode“ the motion can be carried out. In the first two modes no motion segment may be active. In the mode “Profile Velocity Mode“ the shifting is not necessary, because an overflow does not have any impact on the operation (see Profile Velocity Mode – velocity control with ramp profile on page 113). But the shifting can be very useful when you want to avoid the problem with the overflow and the problem with the conversion between the user unit and the increment, or when you want to limit the position indication in certain numbers.

In order not to have to directly activate these objects, you also have the option of using the Dll functions of the Isel-Can control. However, this only applies to the CAN connection

Encoder monitoring

Using this object, the user can check the functioning of the incremental position encoder.

Monitoring the encoder

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
2057		Position Encoder Supervise	RECORD				Monitoring the position encoder
2057	00	Number of Entries	Unsigned 8	CONST	N	6	Number of entries
2057	01	Index Supervise Enable	Unsigned 8	RW	N	1	0: No monitoring of the index signal. 1: Index signal is monitored.
2057	02	Index Check Area (rev)	Unsigned 16	RW	N	2	Monitoring range in rotation
2057	03	Actual Index Counter	Unsigned 32	RO	N		Current number of arriving index signals
2057	04	Position Supervise Enable	Unsigned 8	RW	N	0	0: No monitoring of the encoder signals. 1: Encoder signals are monitored.
2057	05	Position Discrepancy Limit (inc)	Unsigned 16	RW	N	100	Maximum permissible position discrepancy
2057	06	Actual Position Discrepancy (inc)	Unsigned 16	RO	N		Current position discrepancy

The monitoring of the encoder index signal is an important thing. The output stage uses the hall values (set in the dialogue box "Motor and Transmission" and stored in object 6410_h) for the commutation of the 3-phase motors until the index signal arrives. After that, the index signal is used for the commutation. Please note that the index signal must only arrive once after switching on and then the output stage independently further calculates the data for the commutation. In normal cases, the commutation with hall sensors only lasts for a maximum one revolution because the encoder delivers one index signal per revolution. Trapezoidal currents are emitted during the hall sensor commutation. It leads then to a strong torque fluctuation and poor quality in the controller. Heavy acceleration can also cause a high current. Sinusoidal currents are emitted in the commutation with the index signal. Here, there is hardly any torque fluctuation and high control quality is achieved here. The power requirement also remains low.

Faulty hall sensors are detected immediately by the output stage after switch-on and further operation is not possible. However, an incorrect index signal usually goes unnoticed, because the motor continues with the hall commutation. With the object "Index Supervise Enable" equal to 1, you may request that the output stage detect the incorrect index signal. The number of revolutions of the motor is defined in the object “Index Check Area" defines. The output stage waits for the index signal, until the set number of motor revolutions is exceeded. The motor continues to run unnoticed if there is an index signal. If the index signal fails to arrive, the output stage then has the error 19 (meaning: index signal error - see “Fault states” on page 45). With the object "Index Supervise Enable" equal to 0, you turn off the monitoring of the index signal. While the error 19 can no longer occur here, it may still occur somewhere else. E.g. in a reference run with index signal, there is a logical check to see whether or not the index signal is available. If the index signal does not arrive or is faulty, the output stage then has the error 19.

Regardless of the monitoring of the index signal, you still have the option here of monitoring the step loss. There are several reasons for a step loss, such as errors on the encoder cables or too high movement speed. If the speed is too high, the output stage cannot handle the high frequency of the encoder signals. This error is more common than you think. The step loss monitoring is based on the fact that the encoder provides a certain number of increments between two index signals. A deviation is an indication of a step loss. The tolerance range is defined in the object " Position Discrepancy Limit ". As long as the deviation is in this range, there is no error message. If the tolerance range has been left, the output stage has the error 42 (see “Fault states” on page 45). The object “Actual Position Discrepancy" returns the current deviation. Normally, the value is always 0 and the monitoring of the step loss is only optional. You have to turn this option on or off via the object "Position Supervise Enable". The step loss can only be monitored with an active monitoring of the index signal.

Actual operating states

Here are different objects showing current internal states of the power amplifier and the motor.

Active communication interface

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
2050	00	Active comm. interface	Unsigned 8	RO	N		Currently set target value channel

This parameter indicates which target value channel is currently activated; said channel is fixed by means of the DIP switch on the front side and queried during boot-up or reset.

Set bit	Designation	Meaning
Bit 0	CAN	Via CAN, data can be read.
Bit 1	RS232	Via RS232, data can be written.
Bit 2	Analogue input (current controller)	Control word and mode of operation cannot be changed. Control of the state machine via release signal.
Bit 3	Analogue input (velocity controller)	See current controller.

Error byte

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
2052	00	Error byte	Unsigned 8	RO	Y		Display of the current error code.

In this parameter, the internal error code is indicated, when an error occurred (Fault state). If case of several errors, the error code with the highest priority (smallest value) is given (see “Fault states” on page 45).

Actual motor current

Index	Sub	Name	Typ	Attrib	Map	Defaultwert	Meaning
2070		Actual Motor Current	RECORD				Actual motor current (mA)
2070	00	Number of Entries	Unsigned 8	CONST	N	6	Number of entries
2070	01	Iq	Signed 16	RO	N	0	Actual current Iq
2070	02	Id	Signed 16	RO	N	0	Actual current Id
2070	03	Iu	Signed 16	RO	N	0	Actual current Iu
2070	04	Iv	Signed 16	RO	N	0	Actual current Iv
2070	05	lw	Signed 16	RO	N	0	Actual current lw
2070	06	Irms	Unsigned 16	RO	Y	0	Actual root mean square Irms of current

Actual analogue input value

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
2071	00	Actual analogue input	Signed 16	RO	Y	0	Current analogue target value [mV] -10000 .. +10000

DC Bus Voltage

Index	Sub	Name	Typ	Attrib	Map	Defaultwert	Meaning
2072	00	Number of Entries	Unsigned 8	CONST	N	2	
2072	01	IMD20 Actual Active Level Status	Unsigned 8	RO	N	0	At IMD20: 1= DC Bus Voltage over 30V, 0 = DC Bus Voltage under 30V
2072	02	IMD40 Actual Voltage	Unsigned 16	RO	N	0	At IMD40: actual DC Bus Voltage

Motor controller

Our power stage uses a special controller, which controls the 3-phase motor. Using these objects, the internal registers of the controller can be read or write. Writing the internal register is a very critical thing. It should be done only by experienced personnel and even in test mode. Before writing the controller register, the object 205B has to be used to unlock the write protection.

Motor controller write register

Index	Sub	Name	Typ	Attrib	Map	Defaultwert	Meaning
2059		Motor Controller Write Register	RECORD				Writing in registers of the motor controller
2059	00	Number of Entries	Unsigned 8	CONST	N	2	Number of entries
2059	01	Register Offset	Unsigned 8	RW	N	0	Offset write register
2059	02	Register Size	Unsigned 8	RW	N	1	Register length in byte = 1,2,3,4
2059	03	Data to Register	Unsigned 32	WO	N	0	Data to register

With this object the user can write data into all write register of the internal motor controller. In dependence of "Register Size" the first byte or the first 2, 3, or all 4 data bytes of "Data to Register" are written with the address in "Register Offset".

Motor controller read register

Index	Sub	Name	Typ	Attrib	Map	Defaultwert	Meaning
205A		Motor Controller Read Register	RECORD				Reading form registers of the motor controller
205A	00	Number of Entries	Unsigned 8	CONST	N	2	Number of entries
205A	01	Register Offset	Unsigned 8	RW	N	0	Offset read register
205A	02	Register Size	Unsigned 8	RW	N	1	Register lenght in byte = 1,2,3,4
205A	03	Data from Register	Unsigned 32	RO	N	0	Data from register

With this object the user can read all read register of the internal motor controller. "Data from Register" can be loaded with the data of the read register in dependence of "Register Size" with the first byte or with the first 2, 3 or all 4 bytes.

Motor controller access control

Index	Sub	Name	Typ	Attrib	Map	Defaultwert	Meaning
205B	00	Motor Controller Access Enable	Unsigned 8	RW	N		Enable or disable access to motor controller 0 → disable 1 → enable

With this object the user can control the access to the inside motor controller. This object is normally used for the test purpose during the development phase.

Fast stop on inputs

These objects are meant for the fast stop of the motion of one or more axes upon receiving a desired bit sample at an input port, latchin the real positions of the axis (latched). If one needs these positions, they can be read back.

Motion Stop On Input

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
205C		Motion Stop On Input	RECORD				Fast stop of a motion with active inputs
205C	00	Number of Entries	Unsigned 8	CONST	N	13	Number of entries
205C	01	Motion Stop Mode	Unsigned 8	RW	N	0	0: Operation mode not active 1: External inputs, stop and latching at syn-object 2: Internal inputs, stop and latching at syn.-object 3: Internal inputs, immediate stop and latching
205C	02	Stop Delay	Unsigned 8	RW	N	0	Delay time in [Syn.-interval] until the start of the stop
205C	03	Stop Acceleration	Unsigned 32	RW	N	100000	Brake acceleration [udu/s²] upon acceleration stop
205C	04	Latch Delay	Unsigned 8	RW	N	0	Delay time in [Syn.-interval] until the latching of the real position
205C	05	Latched Position	Integer 32	RO	N	0	Latched real position [udu]
205C	06	Amplify Input Port Number	Unsigned 8	RW	N	0	Number of the amplify-input port for the motion stop
205C	07	Input Glitch Filter	Unsigned 8	RW	N	0	suppression filter 0: Not active 1: Active
205C	08	Input Value	Unsigned 8	RW	Y	0x00	Current input bit pattern
205C	09	Filter Mask	Unsigned 8	RW	N	0x00	Filter mask
205C	0A	Reference Value	Unsigned 8	RW	N	0x00	Desired input bit pattern for the motion stop
205C	0B	Motion Stop Control	Unsigned 8	RW	Y	0	"Motion Stop On Input" 0 → 1: Start
205C	0C	Motion Stop Status	Unsigned 16	RO	Y	0x0000	Internal status of the operation mode "Motion Stop On Input"
205C	0D	Auxiliary Variable	Integer 8	RW	Y	0	Auxiliary variable

There are many applications for this function. A typical example is a measuring probes to measure edges or surfaces of the working pieces. Functionally the probes is a switch, connected to an input. By pressing the switch there is a level changing at the input, and the axis position is read for measurement purposes. The measurement probes must be stopped at the same time. A delayed motion stop causes a stronger deviation and finally the breaking of the measurement probes. The level change is at the input as a trigger signal, causing the start of all activities. With this example only one input serves as trigger signal. But this must not be always the case. It is possible that more or all inputs of an input port are included. The arrival of a certain bit sample at these inputs is then the trigger signal for this operation mode.

One thing still needs to be explained. The motion stop on inputs means that the levels of the final stage controlling the axis must be known. Normally the inputs are directly on the final stage. So, in other words, they are the inputs of the final stage. If there are more axes, the inputs of the axes can be connected easily by hardware. In this way the bit pattern is on all axes. But this is not the only possibility. In case of a decentral control with one bus as information transfer means the used input port can also be elsewhere, for example in an IO-module. The inputs are then sent by the bus to the final stages. It is also possible that the input port of one final stage is used. The other final stages get the information then from the bus. The information transfer through the bus has the advantage that the there are few hardware cabling works. The easy expandability is another advantage. With the CAN-bus with the CANOpen-Standard the data transfer can be done very easily with the PDO-mechanism (see PDO on page 93). In the CAN-module with the used inputs the inputs are packed in a PDO (mapping), and then put on the bus. All corresponding final stages, requiring the inputs, are programmed for the receipt of this PDO. The final stages have for this purpose an own object, receiving during the PDO-reception the inputs arriving with the PDO. This object is a virtual input port, whose bit pattern works as trigger signal. Apart from the PDO, you can also take SDO, to set the virtual input. The use of a serial interface, to set the virtual port, is also possible.

When several axes take part simultaneously in the action, it is possible that the virtual ports are not updated at the same time. This can be avoided with the use of the synchronisation object (see Types of transfer on page 95 and SYNC on page 96). Only upon arriving the synchronisation object, the virtual ports are read in the final stages and when the desired bit pattern has arrived, corresponding activities are started. The concurrence with all axes is guaranteed in this way. Another problem during the information transfer through the bus is the delay time. A short period of the synchronisation object can be useful here. But it is not possible to eliminate the delay time completely. If the time requirements are very high, the inputs must be directly hardware-cabled, as already indicated.

The objects listed in the chart are necessary to make the operation mode “Motion Stop On Input“ both effective and flexible. Now the different objects are described in a detailed way.

By writing the object “Motion Stop Mode“ the working way of this operation mode can be determined

- Object = 0 → Mode 0: The operation mode “„Motion Stop On Input“ is not active.
- Object = 1 → Mode 1: To stop the motion external inputs are used. The inputs come with a PDO in the object “Input Value“. This means that the object “Input Value“ is the virtual input. However, the confirmation of the inputs in the internal processing is only done when the next synchronisation object has arrived.
- Object = 2 → Mode 2: To stop the motion internal inputs on the final stage are used. However, the reading of the internal inputs is only done when the synchronisation object arrives. In this way it is possible to guarantee the concurrence with all axes.
- Object = 3 → Mode 3: To stop the motion internal inputs on the final stage are used. The inputs are updated periodically in intervals of 125 µs. Here the synchronisation object is not taken into consideration. Since the period of the synchronisation object can go from one to several milliseconds, the changes on the inputs are registered very fast. Here we give priority to quickness rather than to concurrence.

When determining a certain working mode for an axis, the users are free. They only must guarantee that the inputs can reach all axis.

In the case where inputs of a IO-CAN-module are used, all axes must be in mode 1. Through the bus the IO-module sends a PDO with the inputs. The object “Input Value“, indicated also further on, is the virtual input port. This object must be programmed to receive this PDO.

Both modes 2 and 3 are meant when an input port is used on the final stage. Through the hardware cabling the inputs of several axes can be connected with each other. The trigger signal is then directly available for the axes. A mix of both modes in the different axis is also possible. The object

“Input Value” is here the reflection of the real input port. Each time when the used input port is read, the object “Input Value” is updated. Contrary to mode 1, where this object is the virtual input port, this object is not used to stop the motion. The motion stop is triggered by the real input port. In both modes the aim and the significance of the object is somewhere else. This object can be mapped in a PDO and sent by the bus to the axes that are not connected with the trigger signal through hardware. These receipt-axes must be in mode 1 and must be programmed in such a way that its object “Input Value” gets the sent inputs during the PDO-reception. If an axis is connected through hardware with the trigger signal, but is in mode 1, its inputs will not be requested internally.

Important: the modes 1, 2, 3 of an axis can only be activated, when the axis is in the operation mode Interpolation (see Interpolated Position Mode – Position control with interpolation on page 118) or positioning (see Profile Position Mode – position control with ramp profile on page 115).

With the object “Stop Delay” the users can determine how many synchronisation intervals the axis must delay the motion stop upon arriving of the trigger signal. Trigger signal upon arriving. Why? one should ask. The following scenario will explain it. An axis in mode 2 or 3 reads its inputs and sends the input values through PDO to the axes in mode 1. And if it is a synchronous PDO, the trigger signal only arrives 1 synchronisation interval later. When the object “Stop Delay“ of the sending axis is put on 1 and the receiving-axis “Stop Delay“ on 0, both axes will start the motion stop simultaneously. In the Isel-CAN-CNC-control, with the Master of a Gantry-axis, we have set the object “Stop Delay” even on 2, when the Master is in mode 1. Here the virtual input port is sent with a delay of an additional synchronisation interval through the Master to the Slave. In this way, both the Master as the Slave can start the stop procedure at the same time, which is important for the synchronisation between the two axes.

With the start of the motion stop the axes move during the braking phase independent from each other, even if they were together in an interpolation motion directly before the motion stop. In that case the axes will be stopped at different times. When the axis velocities are known, the object “Stop Acceleration” can be set so that the axis motions are like a interpolation-motion. All axes stop then approximately at the same time. The braking has a linear ramp. Consequently, the calculation of the axis accelerations is not difficult. However, if you think that a motion like a interpolation is not important, the object “Stop Acceleration” with the corresponding maximal acceleration can be escribed. Then the axes are maximally slowed down and will stop fastest.

Similar as in the case of the object “Stop Delay”, with the object “Latch Delay” you can determine how many synchronisation intervals the axis must delay the latching of the real position upon receiving the trigger signal. In this way it is possible that the latching of the real position takes places at the same time with all axes. When the object “Latch Delay” is zero, the real position will be confirmed immediately upon arriving the first synchronisation object after receiving the trigger signal when the axis is in mode 1 or 2. In mode 3 the latching is done immediately when the trigger signal has arrived. The synchronisation object is not important here. In case of the very exact strobing mode 3 is very interesting, when all inputs of all axes are connected to the source of the trigger signal by means of hardware. In this case the trigger signal arrives at all axes at the same time.

If the real position has been latched once, it can be read back through the object “Latched Position”. If the real position has not been latched yet, there will be an error upon reading this object. A latched real position that has not been read yet, will be overwritten during the next motion stop. Important: the real position will only be latched once after each start with the object “Motion Stop Control“, after arriving of the trigger signal. The new latching is only possible again when the mode “Motion Stop On Input“ with the object “Motion Stop Control” is reset and set again.

In mode 2 and 3 the axis is working with its own inputs. If the final stage has several input ports, then the port can be determined whose inputs form the trigger signal with the aid of the object “Amplify Input Port Number“. The final stages IMD20/40 have only one port. Therefore, this object is always zero.

With an active object “Input Glitch Filter“ a low-pass filter implemented with software has been connected to suppress possible disruption on the inputs. The disadvantage of a low-pass filter is the signal delay. The filter is disconnected by default.

The object “Input Value“ has already been handled above with regard to the object “Motion Stop Mode“. It is a very important object for this operation mode. Important: this object is used differently in mode 1 and in mode 2 and 3. In mode 1 this object can be used as the virtual input. In mode 2 and 3 the object is the reflection of the input port, enabling the motion stop. This reflection can be sent through the PDO to the axis in mode 1, to set the object “Input Value” that is working here as the virtual input.

Independently whether the object “Input Value“ is used in mode 1 or the internal inputs are used in mode 2 or 3, the bit enabling the trigger signal must always be considered as complete bytes. Of course, it is possible that you only want to use one or few bits of a byte. In this case, the object “Filter Mask” will be used to shield the bits that are not used. A bit value of zero means that the bit is not being evaluated. A bit value of 1 indicates that the bit belongs to the bit pattern.

The final stag controls in mode 1 constantly the object “Input Value” or in mode 2 or 3 the internal inputs. Upon arriving the bit pattern determined in the object “Reference Value” the real position is latched and the motion stop is started.

The example of the final stage IMD20/40 in mode 2 or 3 can explain the combination between the three objects “Input Value“, “Filter Mask“ and “’Reference Value“. In the only input port (see Digital inputs on page 133) the final stage IMD20/40 only has one free bit (the input 5 at the 4th bit position. Do not forget that the bit numbering starts with 0). This bit is normally foreseen to connect the motor temperature sensor. If you want to use this bit as trigger signal, first you must configure the bit for the “free use” (see dialog on page Inputs dialogue box on page 75). Additionally, you must define whether level 0 or 1 is the active level. With the value 1 the active level is shown in the input. The inactive level is shown there with the value 0. To be able to evaluate only the 4th bit of the input, the object “Filter mask” gets the value 0x10. If the condition

[Hardware_Input-Port] bit_by_bit _AND_logic_operation [Object “Filter Mask“] =

[Object „Reference Value“]bit_by_bit _AND_logic_operation [Object “„Filter Mask“]

is fulfilled, the real position and the stop are started. In the mode 1 there is no hardware limitation. The object “Filter Mask” can be set independently from the fact which bits of the object “Input Value” arriving through PDO belong to the bit pattern. Here the condition

[Object „Input Value“] bit_by_bit _AND_logic_operation [Object “Filter Mask“] =

[Object “Reference Value“] bit_by_bit _AND_logic_operation [Object “Filter Mask“]

must be fulfilled, to latch the real position and to start the stop procedure.

Apart from the object “Latched Position” all objects of the subindexes 01_h, 02_h, … $0A_h$ must be set before activating the operation mode “Motion Stop On Input“ with the object “Motion Stop Control“. Now a motion can be started. During the active operation mode “Motion Stop On Input“ the final stag is requesting constantly information of the object “Input Value” and/or the hardware inputs. When the trigger signal arrives, the position is latched and the stop procedure is started. Afterwards the latched position can be read. A new use of the operation mode “Motion Stop On Input” is only possible if in advance with the object “Motion Stop Control“ the operation mode “„Motion Stop On Input“ is deactivated. Consequently, the operation mode “Motion Stop On Input“ can only be triggered by the flank 0 → 1 on the object “Motion Stop Control“. The activation of “Motion Stop On Input“ is subject to another limitation. The both operation modes “Motion Stop On Input“ and “Online distance regulation” may not be active at the same time.

The object “Motion Stop Status“ shows the internal status of the operation mode “Motion Stop On Input“. This object can be read back at any time. For the significance of the bits, see the following chart.

Bit	meaning
0	0 → Operation mode from Stop_On_Input_Mode has not been set yet Object “Motion Stop Mode“ is 0 1 → Operation mode from Stop_On_Input_Mode has been set Object “Motion Stop Mode“ is 1, 2, or 3 (see bit 2)
1	0 → Stop_On_Input_Mode is not active. Object „Motion Stop Control“ is 0 1 → Stop_On_Input_Mode is active Object „Motion Stop Control“ is 1
2	If bit 0 is not 0 0 → The bit pattern comes with a PDO through the bus. Object “Motion Stop Mode“ is 1 (Mode 1 active) 1 → The bit pattern comes through the hardware inputs Object “Motion Stop Mode“ is 2 or 3 (Mode 2 or 3 active)
3	If bit 0 and bit 2 are 1 0 → Hardware inputs are updated in a period of 125µs. Object “Motion Stop Mode“ is 3 1 → Hardware inputs are updated arriving the synchronisation object. Object “Motion Stop Mode“ is 2
4	0 → Object “Input Value“ has not been updated yet after activation of the operation mode “Motion Stop On Input“. Mode 1: PDO has not come yet Mode 2, 3: Inputs have not been read yet 1 → Object „Input Value“has already been updated at least once after activation of the operation mode “Motion Stop On Input“. Mode 1: PDO has already come Mode 2, 3: Inputs have already been read
5	0 → The desired bit pattern is presently not at the input 1 → The desired bit pattern is presently at the input
6	0 → The desired bit pattern has never come. 1 → The desired bit pattern has come already one.
7	0 → Real position has been latched 1 → Real position has not been latched yet
8	0 → The stop procedure has not been started yet 1 → The stop procedure has been started
9	0 → The axis is in motion 1 → The axis is not in motion
10 … 15	Unused bits and always 0

The last object “Auxiliary Variable“ is an auxiliary variable for the data transfer between the Master and the Slave of a Gantry-axis, and is only used with the CAN-CNC-control.

Upon using our CAN-CNC-control, you are working with the DII-functions at a higher level. You do not have direct contact with the objects that have been treated here. But studying the objects may be useful to understand the working of the operation mode “Motion Stop On Input“ deeper and more thoroughly. The use of our DII-functions simplifies your work, but is not an obligation. By setting the objects you can operate one or more axes also in this operation mode. The use of the CAN-bus is not either a necessary condition for this operation mode. Instead of the CAN-bus, you can also take the serial interface.

Electronical handwheel

For the configuration the machine axes can be moved manually or step-by-step by means of the keys of a keyboard or by means of axis adjustment keys of the operation panel of the machine. However, it is easier and safer with a handwheel. With a portable handwheel the machine operator is always immediately on site, with an overview of the configuration steps and controlling the infeed exactly and correctly.

All objects that have been described here are meant to integrate the Isel handwheel RCS07 in the CAN-CNC-control. Without the CAN-CNC-control the Isel handwheel does not work. Unfortunately, with these objects it is not possible to integrate another handwheel in the CAN-CNC-control. For the integration of the handwheel in an application software, there are DII-functions that operate logically also these objects to realise the functionality of a handwheel. For the use of the DII-functions, a study of this objects ist not necessarily, but they are useful to get a better understanding of the working of the handwheel. The interpolation axes of the CAN-CNC-control are determined in advance by means of the initialisation file. These axes can be operated readily with the handwheel. But if you want to connect additional handling axes with the handwheel, these objects must be studied. The manual for the handwheel RCS07 must also be used to get a faster and a better understanding of the structure and the working of the handwheel and the objects.

Handwheel

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
205D		Handwheel	RECORD				Handwheel
205D	00	Number of Entries	Unsigned 8	CONST	N	23	Number of entries
205D	01	Handwheel Mode	Unsigned 8	RW	N	0	0: unlink handwheel 1: link handwheel
205D	02	Multi Axis	Unsigned 8	RW	N	0	0: Single axis operation 1: multi axes operations
205D	03	Axis Coding	Unsigned 16	RW	N		Axis coding
205D	04	Max Velocity (udu/s)	Unsigned 32	RW	N	50000	Maximum velocity [udu/s]
205D	05	Acceleration (udu/s²)	Unsigned 32	RW	N	100000	Acceleration [udu²/s]
205D	06	Jog Velocity (hw_inc/s)	Unsigned 16	RW	N	20	Jog velocity [hw_inc/s]
205D	07	Step Width Small (udu/hw_inc)	Unsigned 16	RW	N	250	Step width [udu/hw_inc] for slow motion
205D	08	Step Width Normal (udu/hw_inc)	Unsigned 16	RW	N	1250	Step width [udu/hw_inc] for normal motion
205D	09	Step Width Large (udu/hw_inc)	Unsigned 16	RW	N	2500	Step width [udu/hw_inc] for fast motion
205D	0A	Keys Enable	Unsigned 16	RW	N	0x0FC7	Enable/Disable the handwheel keys Bit_value = 0 : Disable Bit_value = 1 : Enable
205D	0B	Keys Mode	Unsigned 16	RW	N	0x05FF	Mode of the handwheel key Bit_value = 0 : Key Bit_value = 1 : Switch
205D	0C	Inputs	Unsigned 16	RW	Y	0x0000	Current status of the handwheel inputs
205D	0D	Position (hw_inc)	Unsigned 16	RW	Y	0	Current counter of the handwheel [hw_inc]

(Handwheel - continuation)

Index	Sub	Name	Type	Attrib	Map	Default value	Significance
205D	0E	Position Timestamp (ms)	Unsigned 16	RO	Y	0	Current counter of the timer in [ms]
205D	0F	Init Outputs	Unsigned 16	RW	N	0x0000	Initialisation value for the handwheel outputs
205D	10	Outputs	Unsigned 16	RO	Y	0x0000	Current value for the handwheel outputs
205D	11	Latched Position (udu)	Integer 32	RO	N	0	Latched real position [udu]
205D	12	Handwheel Control	Unsigned 8	RW	Y	0	0: Deactivate handwheel 1: Activate handwheel
205D	13	Handwheel Status	Unsigned 8	RO	Y	0x00	Current handwheel status
205D	14	Handwheel Master Status	Unsigned 8	RW	Y	0x00	Current handwheel status of the Master of a Gantry-axis
205D	15	Standstill Control	Unsigned 8	WO	N	1	On/Off of the standstill control 0: Off 1: On
205D	16	Handwheel Supervise Control	Unsigned 8	WO	N	1	On/off of the handwheel supervise control 0: Off 1: On
205D	17	Auxiliary Variable	Integer 8	RW	Y	0	Auxiliary variable

The procedure of the handwheel is better understood when one is conscious of the fact that the handwheel and the axis are separated CAN -modules and communicate with each other through the CAN-bus and make one unit. A direct access to the handwheel is not required. The whole procedure is done through the axis, with the aid of the objects currently indicated. The handwheel controls the axis. Some objects must be set to influence on the behaviour of the axis upon access through the handwheel. Consequently, these objects differ from axis to axis. Some objects simulate the internal handwheel status. That is why these objects are identical in all axes. Some objects are available because they can be used to get access to the handwheel. These objects are available at all axes. But the CAN-CNC-control logically only requires one axis, to get access to the handwheel through its objects.

Upon writing 1 in the object “Handwheel Mode“ the axis is prepared for use with the work with the handwheel. In that case, the axis is in the handwheel-ready-operation-mode. However, it is not possible yet to move the axis with the handwheel. A value 0 in the object “Handwheel Mode“ means the disconnection of the handwheel. In that case there is no connection anymore between the control and the handwheel. In the handwheel-ready-operation-mode only the key for the latching of the real position as well as both function keys F1 and F2 can be used. All other keys are locked. The inputs and the counter of the handwheel-adjuster can be requested by means of both objects “Inputs” and “Position”. The object “Handwheel Mode“ causes an initialisation of the axis. That is why all objects with the subindexes 02_h, … $0B_h$ have already been written in advance. These objects cannot be written anymore in the handwheel- ready-operation-mode.

With the object “Multi Axit“ it is possible to determine that, upon pusch several handwheel axis keys simultaneously, the corresponding axes can be moved at the same time. Therefore, this object must have the value 1. In case of the value 0 only one axis is moved, namely the first axis in the order X, Y, Z, A, B, C.

Each handwheel key relates to an input of the handwheel input port. The handwheel sends in frequently the status of the input port through the bus to all final stages. With other words, all final stages are always informed which keys are presently occupied. With the object “Axis Coding” the assignment between handwheel axis keys and axis can be determined. Each final stag compares the frequently coming handwheel input port with its own object "Axis Coding" to find out whether the key that has been assigned to it is currently occupied or not.

The object "Maximal Velocity" limits the moving velocity of the axis, independent whether the axis is moved with the handwheel adjuster of with the jog-keys. The value of this object in the user defined unit is internally limited to the maximal axis velocity.

With the object "Acceleration" the acceleration of the axis during the moving with the handwheel is determined. The object in the user defined unit is internally set on the maximal axis acceleration.

The object "Jog Velocity" determines the velocities of the jog-motion. Important: the velocity must here be indicated in [Handwheel increments / s] although a jog-motion is done with the keys +/- and not with the handwheel adjsuter. The reason is the fact that the real axis velocity is the product of the value here indicated and one of the step widths (subindexes 07_h, 08_h, 09_h). The step widths have the unit [user defined unit / handwheel increment]. The calculated axis velocity then has at the end the unit [user defined unit / s].

The three objects "Step Width Small", "Step Width Normal" und "Step Width Large" determine with which length the axis is moving, if the handwheel adjuster is turning 1 increment. In case of a motion with the handwheel adjuster the motion length of the axis is the product of the rotation angle of the handwheel adjuster [Handwheel increment] and the active step widths in [user defined unit / handwheel increment]. In case of a jog-movement the object "Jog Velocity" must be taken instead of the rotation angle. With these three objects three velocities can be determined. The selection of a step width is done with the feed keys. If more than one key is pushed at the same time, the first key in the order Slow, Normal and Fast is active.

The objects "Max Velocity", "Acceleration" and "Jog Velocity" as well as the 3 step width objects are axis orientated. This means that they can be set differently from axis to axis to get an appropriate motion of each axis.

If you want to stop a certain handwheel key, the object "Keys Enable" is used. The assignment of the keys to the bits can be found in the manual of the handwheel. The bit value 0 means the off-mode for the key. In the off-mode, the key does not react upon pushing. The bit value 1 means that the key is being used. The fact that the keys must be disconnected is a rather frequent procedure. For example, when the system has only 3 axes, X, Y, Z, the other axis keys A, B, C must already be deactivated. Or, if you do not want to use the function keys F1 and F2, they can also be excluded.

A handwheel key can be in the key mode or in the switch mode. In the key mode the key is active as long as it is pressed. Upon losing the key, it gets immediately inactive. In the switch mode the key changes its status each time when being pressed. The user recognises the active status of a key by means of the lightened LED. When the LED is off, the key is not active With the object "Keys Mode" the key mode or the switch mode can be determined for each key. As indicated above, the assignment between the keys and the bits can be found in the manual of the handwheel. Generally, it is more useful that both keys +/- for the job motion and the key for the latching of the real position are in the key mode. The remaining keys are in the switch mode.

The CAN-CNC-control uses both objects "Keys Enable" and "Keys Mode" of an axis, to control the handwheel. If there are more then one axis, we recommend setting the object "Keys Enable" on the same position for all axis. This also applies on the object "'Keys Mode". In this way bad surprises can be avoided.

With all axis the actual status of the input port of the handwheel through the CAN-bus get in the object "Inputs" and the actual counter of the handwheel adjuster in the object "Position". With the information from these two objects the axis then derives the required activities for them. Important: in the handwheel counter there is the number of the increments of the handwheel adjuster and the handwheel adjuster is giving only 100 increments/revolution. If you want to determine the rotation angle of the handwheel adjuster the object "Position" must be read at least once for each rotation of the handwheel. Otherwise, the overflow error is generated.

In each final stag a 16-bit-timer with a resolution of 1 ms is working independently. The status of the timer is illustrated in the object "Position timestamp". Due to the 16-Bit-data width the maximal time interval is 65536 ms. If you want to measure a longer time with this timer, the contents of the timer must be read at least once within this maximum time interval to avoid overflow.

The three objects “Inputs,” “Position” and “Position Timestamp” can be read again at any time by a higher-level control as the CAN-CNC-control to use the data for its own purposes. All axes have these three objects. It does not matter from which axis the objects are read back.

By means of the object “Outputs” the handwheel LEDs can be enabled. With this object each key can be marked active (LED on) or inactive (LED off). The assignment of the LEDs to the different bits can be found in the manual of the handwheel. By means of the object “Init Outputs” the initial status of the keys by moving the handwheel into the ready-operation-mode can be determined. Important: first only the key for the latching of the real position as well as both function keys F1 and F2 can be initialised. The remaining handwheel keys are still locked. Only upon the first activation of the handwheel with the object “Handwheel Control“ the other keys are set with the corresponding bit-values of the object “Init Outputs“. Upon deactivating the handwheel with the object “Handwheel Control” the current key status is marked internally. Upon a new activation the saved status is written back, as long as the handwheel has not been disconnected yet with the object “Handwheel Mode”. The remaining keys are only set upon the first activation of the handwheel direct after placing the handwheel into the ready-operation-mode with the aid of the object “’Init Outputs”.

With the latching handwheel key the final stages can be requested to save their current positions in the corresponding object “Latched Position”, that can be read back at any time. The real position is not latched, if the axis is not in the handwheel-ready-operation-mod. And if the position has not been latched yet, there will be an error message when you try to read it. A latched position that has not been read yet will be overwritten during the next latch order.

With the object “Handwheel Control” the axis is put into the handwheel-operation-mode where the handwheel takes over the control of the axes. Aktivating of the handwheel-operation-mode of an axis is only possible when the axis is already in the handwheel-operation-ready-mode and in Profile Position Mode (see Profile Position Mode – position control with ramp profile on page 115). When the handwheel operation mode is active, no other motion mode like Interpolated Position Mode, Profile Velocity Mode, … can be connected. Here only the Profile Position Mode may be active. In the handwheel-operation-mode all keys can be used. Although it is not an obligation, it is useful that the handwheel-operation-ready-mode

and then the handwheel-operation-mode are activated at the same time with all axes. For deactivating, the same applies, but of course in reverse order.

The object “Handwheel Status“ shows the internal status of an axis regarding the use of the handwheel. This object can be read back at any time. The significance of the bits can be read in the following chart.

Bit	meaning
0	0 → Handwheel-operation-ready-mode is not active. 1 → Handwheel-operation-ready-mode is active.
1	0 → Handwheel-mode is not active. 1 → Handwheel-mode is active.
2	0 → Communication with the handwheel is free of errors. 1 → Communication with the handwheel is defective.
3	0 → The axis is not selected by the handwheel. 1 → The axis is selected by the handwheel.
4	0 → Real position is not latched. 1 → Real position is latched.
5	0 → Function key F1 is not active. 1 → Function key F1 is active.
6	0 → Function key F2 is not active. 1 → Function key F2 is active.
7	Not used bit and always 0.

In the object “Handwheel Master Status“from the Slave of a Gantry-axis, the object “Handwheel Status “of its Master is shown. With this object the Slave is constantly informed about the status of the Master and can react consequently.

The standstill control on the final stag IMD20/40 has two channels. The hardware channel can be disconnected by the operator during the use of the handwheel with the enabling key. The software channel can be deactivated with the value 0 in the object “Standstill Control“. With the value 1 the software channel is active again. This object can only be written when the handwheel-ready-operation-mode is active. Important: the disconnection of the standstill control is a hard intervention in the safety circuit of the system. Consequently, this step may only be taken when there is no danger.

During the operation with the handwheel each axis monitors with the aid of a WatchDog-signal constantly whether the handwheel is still working correctly or not. In some situations, where the axis changes from one operation mode to another, the internal status is frequently not clear. To avoid conflicts the monitoring of the handwheel is disconnected with the value 0 in the object “Handwheel Supervise Control“. The value 1 in this object reconnects the monitoring. This object can only be written in the handwheel-ready-operation-mode.

The last object “Auxiliary Variable“ is an auxiliary variable for the data transfer between the Master and the Slave of a Gantry-axis, and is only used in case of the CAN-CNC-control.

Ojects for commissioning

The commissioning of an axis consisting of motor, power amplifier and mechanics, is a very complex thing. With objects listed here commissioning of axis was implemented.

Offset analogue input

Index	Sub	Name	Typ	Attrib	Map	Default value	Meaning
2064		Calibrate Analog IO	RECORD				Offset calibration of analogue input
2064	00	Number of Entries	Unsigned 8	CONST	N	3	Number of entries
2064	01	Calibrate Trigger	Unsigned 8	WO	N	0	A value unequal zero starts the calibration
2064	02	Current Offset Phase V	Signed 16	RW	N	0	Current offset of phase V
2064	03	Current Offset Phase W	Signed 16	RW	N	0	Current offset of phase W
2064	04	Analog Input Offset	Signed 16	RW	N	0	Offset of reference input(* 32)
2064	05	Analog Input Zero Area	Signed 16	RW	N	0	Anlog input zero area
2064	06	Supervise Current Offset Phase U	Signed 16	RO	N	0	Supervise Current Offset Phase U
2064	07	Supervise Current Offset Phase V	Signed 16	RO	N	0	Supervise Current Offset Phase V
2064	08	Supervise Current Offset Phase W	Signed 16	RO	N	0	Supervise Current Offset Phase W

This object can be used to realise the offset adjustment of the internal A/D converter. The writing of a 1 on the sub-index 1 starts the automatic adjustment. For this, the module has to be in the "Switched On" (or "Operation Enable") state (see “Device control”).

Under the sub-indices 2 and 3, the current offset values can be queried. These are given in units of 32 times the internal A/D converter resolution (11 bit, -1023 ... +1023). It is also possible to directly write the offset values.

The following objects are used to start and configure the test signals.

Current control test signal

Index	Sub	Name	Typ	Attrib	Map	Defaultwert	Meaning
2090		Test-Input for Encoder and Current Control	RECORD				Test signal for encoder and current controller
2090	00	Number of Entries	Unsigned 8	CONST	N	3	Number of entries
2090	01	Test-Input Duration	Unsigned 8	RW	N	25	Duration of test signal
2090	02	Iq-Test-Input Value	Unsigned 16	RW	N	100	Amplitude of the test signal for the current Iq
2090	03	Id-Test-Input Value	Unsigned 16	RW	N	100	Amplitude of the test signal for the current Id

Velocity control test signal

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
2091		Velocity / Position control: test input	RECORD				Test signal for velocity and position controller.
2091	00	Number of entries	Unsigned 8	CONST	N	4	Number of entries.
2091	01	Test: input duration	Unsigned 16	RW	N	500	Duration of the test signal.
2091	02	Test: input velocity	Signed 32	RW	N	10000	Amplitude of the test signal (target velocity [Incr./s])(1).
2091	03	Test: input acceleration	Unsigned 32	RW	N	10000000	Ramp slope of the test signal (Incr./s²).
2091	04	Max Move Length	Signed 32	RW	N	50000	Maximal Move Length (inc)

(1) Attention: If the “Configure test input” object (index 20A0 and sub-index 01) amounts to the value of 6, the maximum speed is determined. In this case, the amplitude of the test signal is the maximum path length to be moved in [Incr.]. This object restricts the range of movement of the axis during the determination process.

Start test signal

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
20A0		Test input	RECORD				Choose test signal and start.
20A0	00	Number of Entries	Unsigned 8	CONST	N	2	Number of entries.
20A0	01	Configure Test-Input	Unsigned 8	RW	N	0	Selection of the test signal.
20A0	02	Start Test-Input	Unsigned 8	WO	N	0	The writing of a 1 launches the test signal.
20A0	03	Test-Input Status	Unsigned 8	RO	N	0	Test input status

The sub-index 2 helps to carry out the following settings:

Configure test input	Internal mode of operation
0	Back to normal mode of operation
1	Determine maximal acceleration
2	Test signal velocity control
3	Test signal position control
4	Encoder test positive direction
5	Encoder test negative direction
6	Determine maximum velocity
7	Velocity control with ramp profile
8	Following error
9	Speed amplification factor kv
10	Test signal current control

Trace Data

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
2081	00	Trace data	Domain	RO	N		Data field for internal registrations of courses of current and velocities etc.

General parameters

Here, those objects are grouped that cannot be allocated to a certain group, since they affect several modes of operation or functional groups.

Maximum jerk

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
2040	00	Max. jerk	Unsigned 32	RW	N	10000000	Max. jerk in user-defined unit / s^3.

The maximum jerk limits the vibration behaviour of the mechanical axis. The jerk value can be read in and used by the CNC control. Within the final power stage, this parameter is not used. The CNC control of ISEL uses this parameter. The smaller the jerk value, the better the vibration behaviour of the axis. The acceleration or braking process, however, takes longer accordingly.

Axis type

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
2041	00	Axis type	Unsigned 8	RW	N	0	0 Linear axis. 1 Rotary axis.

The CNC control of ISEL uses this parameter for the display. Otherwise, this parameter does not have any particular application purpose.

Fault reset

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
2051	00	Fault reset	Unsigned 8	WO	N		Fault reset or reset.

The writing of a 1 releases a fault reset (alternatively to control word Bit7 0→1). The writing of a 2 releases a complete reset of the module.

Limit switch Bridge

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
2054	00	Limit switch Bridge	Unsigned 8	RW	N	0	0: No shunting 1: Shunting of the limit switches

In case of an activated limit switch, the main supply voltage is switched off for safety reasons. A shunting of the limit switch deactivates the safety circuit. Thus, the supply voltage can be switched on. Afterwards, the axis can be moved out of the active limit switch. Concerning this subject, see “Moving the axis out of a limit switch” on page 123.

Enable/Disable the software limit switches

Index	Sub	Name	Type	Attrib	Map	Default value	Meaning
2058	00	Software Position Limit Enable	Unsigned 8	RW	N	0	0: No monitoring 1: Monitoring active

Setting the software limit switches via the object 607D is not enough to start the monitoring of the software limit switches (see “Software limit switches” on page 131). The monitoring must still be activated via this object. This function must not be used during an active movement in the operating mode “Profile Position Mode“. When this function is called up, software limit switches are not checked. In other words, there may be invalid values here. The error is not noticed until the start of a movement.

EDS / DCF files

The drive module includes a so-called EDS file (Electronic Data Sheet) which contains a description of all objects of a module. This file has a form which is defined in CanOpen so as to make sure that it can be read in by different configuration tools. The ACSetup programme also uses this EDS file in order to create a list of existing objects.

In order to store the current settings of the objects, the DCF files (Device Configuration File) are created. They also contain the object descriptions from the EDS file, but also offer an additional entry, i.e. the current value of the object.

At the beginning of the file, general data on the device and the CanOpen characteristics are given. Then, the object descriptions are given which have the following structure:

e.g. "Reference run velocity” object (6099_h sub-index 1)

File entry	Meaning
[6099sub1]	Index and sub-index
ParameterName=Speed During Search For Switch	Name of the parameter
ObjectType=0x7	Object type (0x7 = variable)

DataType=0x0007	Data type (0x0007 = UNSIGNED32)
AccessType=RW	Access type (RW = read and write admissible)
DefaultValue=10000	Default value
PDOMapping=0	Mapping admissible? (0 = no Mapping)
ParameterValue=15000	Current value, **given only in the DCF file**

Appendix

Connection of Isel EC motor to the IMD20, IMD40

Encoder connection

Cable colour coding *(Encoder on Motor)*	**Pinout SUBD15** *(IMD20,IMD40)*	**Pin designation** *(IMD20, IMD40)*
White	9	HALL_B_IN
White/Black	--	
Green	14	HALL_C_IN
Green/Black	--	
Yellow	1	HALL_A_IN
Yellow/Black	--	
Orange	11	ENC_Z
Qrange/Black	3	/ENC_Z
Grey	13	ENC_A
Grey/Black	5	/ENC_A
Brown	12	ENC_B
Brown/Black	4	/ENC_B
Red	2	VCC_Encoder
Black	10	DGND

Motor connection

Please note:

- *Isel low-voltage motor to IMD20*
- *Isel high-voltage motor to IMD40*
- *There are two cable colour codings of Isel motor*

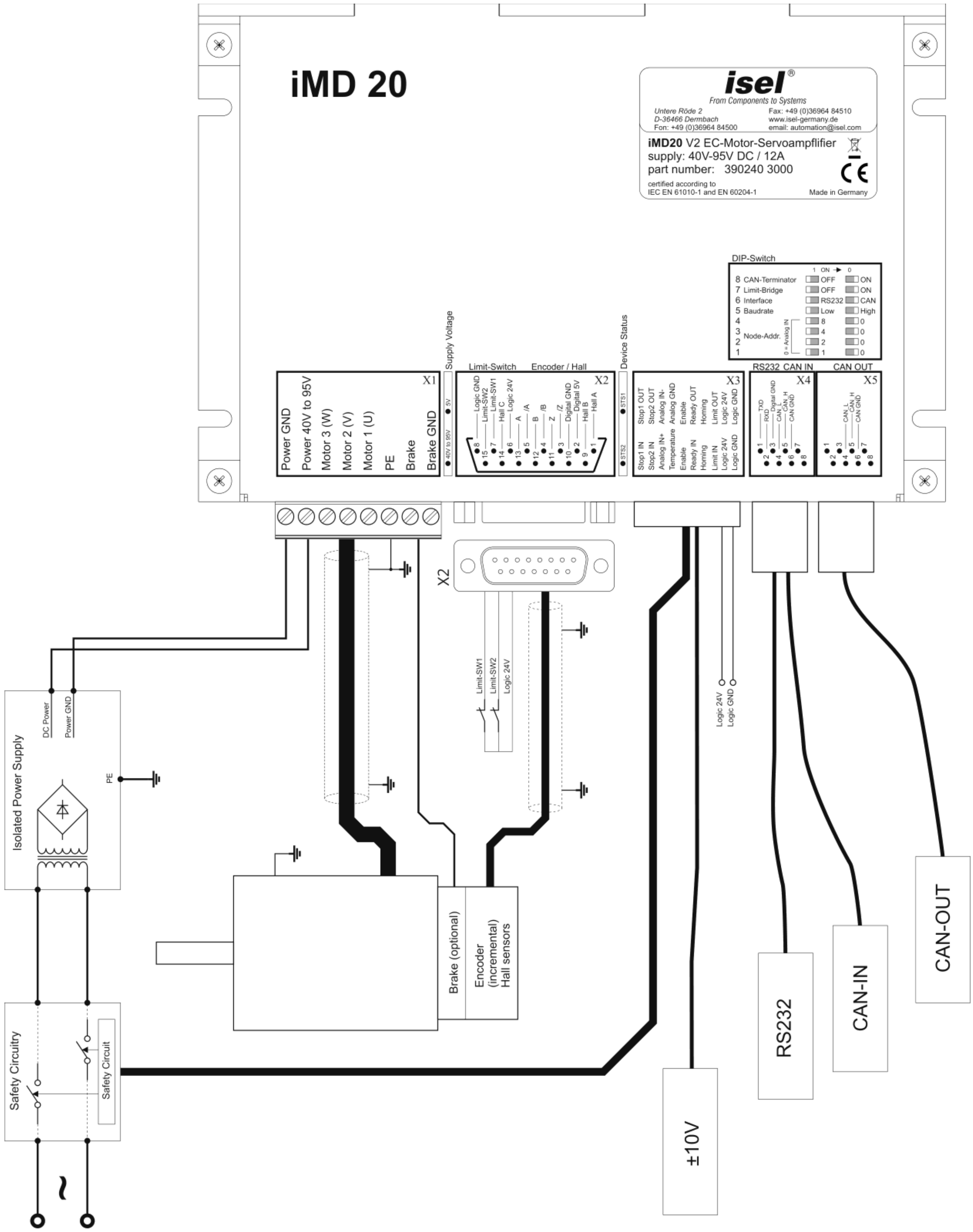
Motor cable colours: Yellow/Blue/Green

Motor cable	**Connection at IMD20**	**Connection at IMD40**
Yellow	Motor 1 (U)	Motor U
Blue	Motor 2 (V)	Motor V
Green	Motor 3 (W)	Motor W

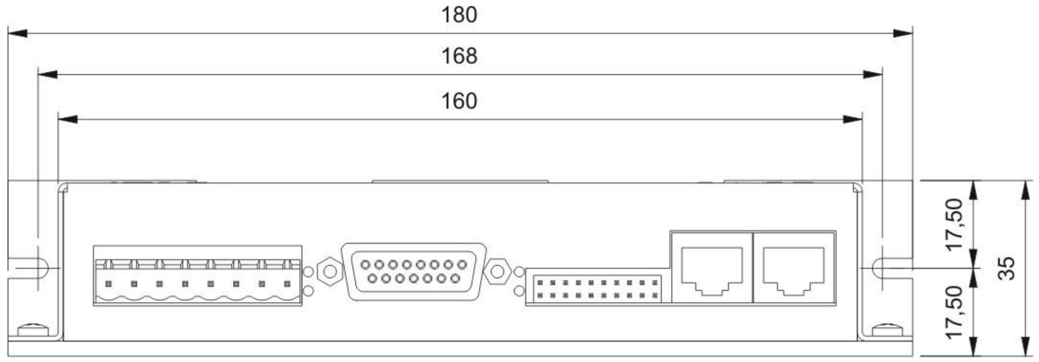
Motor cable colours: Yellow/Black/Blue

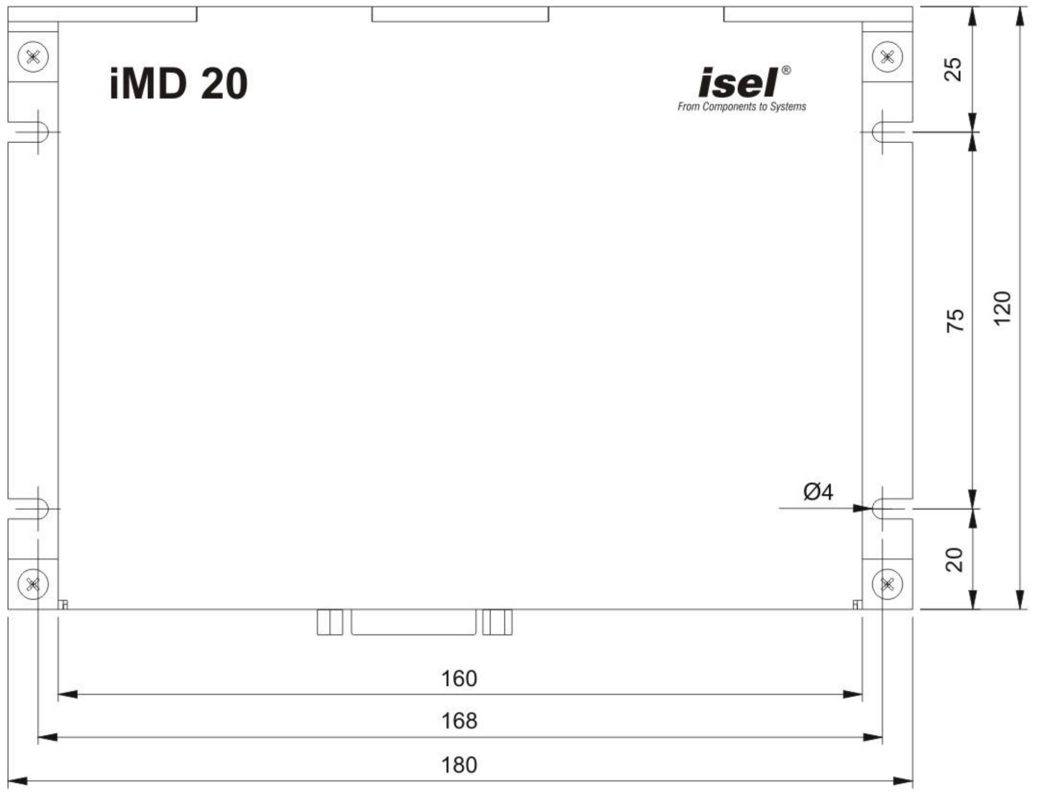
Motor cable	**Connection at IMD20**	**Connection at IMD40**
Yellow	Motor 1 (U)	Motor U
Black	Motor 2 (V)	Motor V
Blue	Motor 3 (W)	Motor W

IMD20 connection

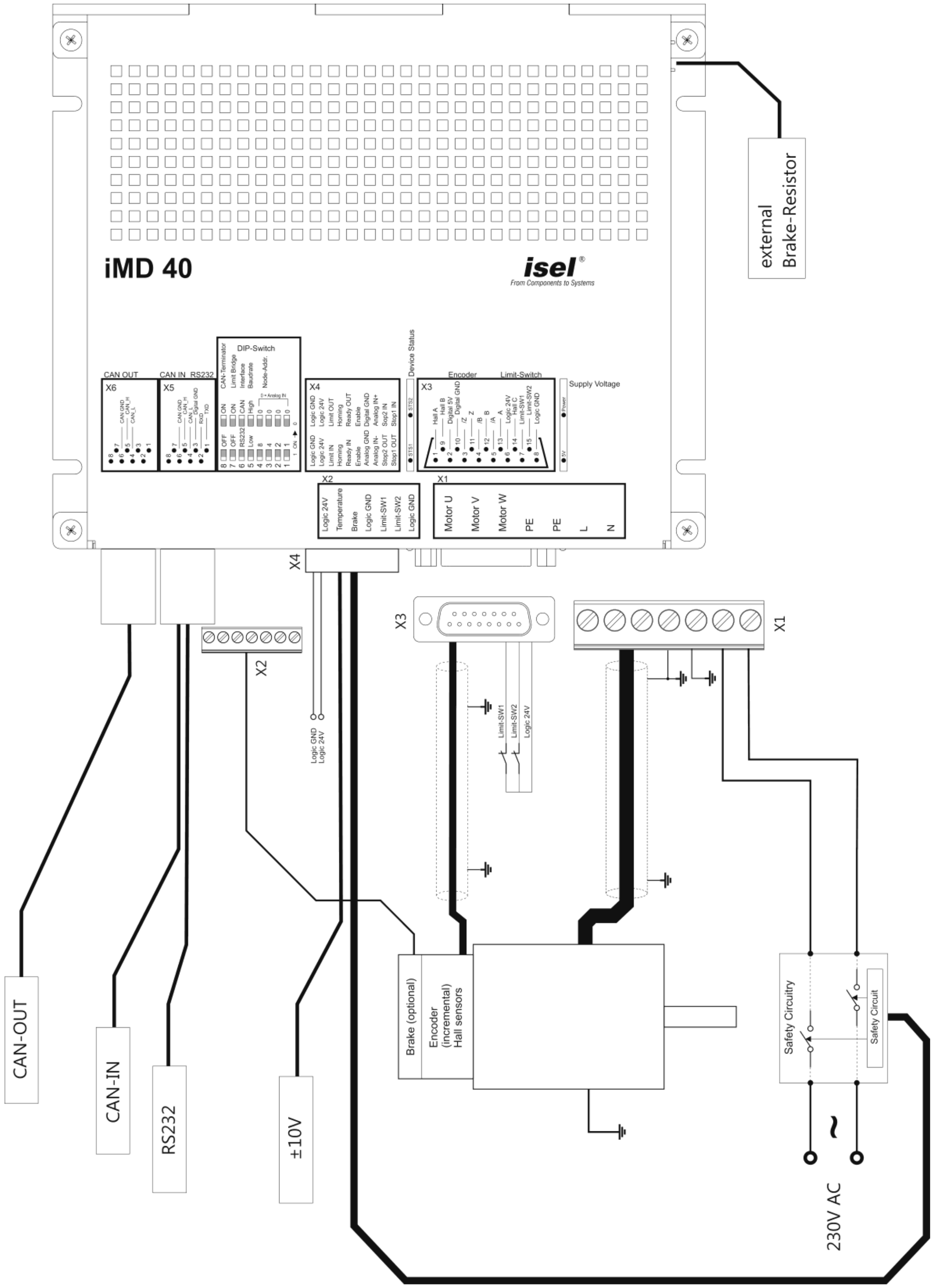


IMD20 package dimensions

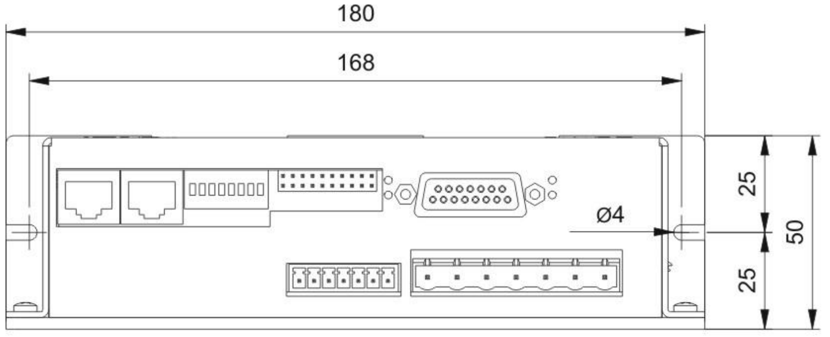


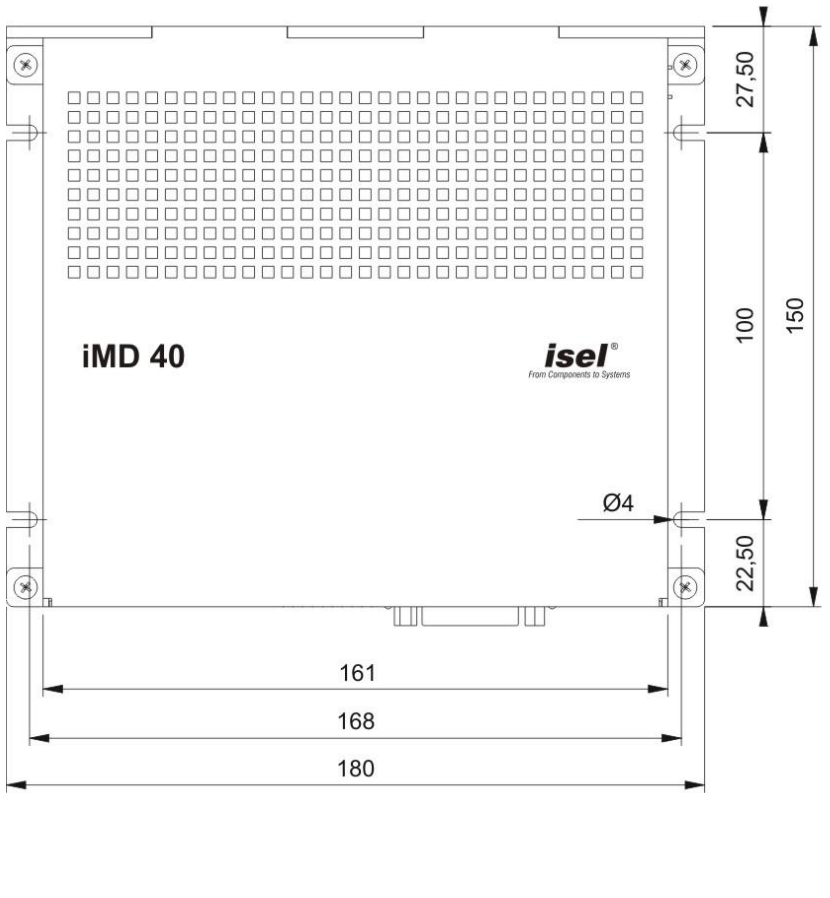


IMD40 connection



IMD40 package dimensions





Index

U

V

W