# Manual
# CANopen Interface
# esiMot

**esitron-electronic GmbH**
Ernst-Zimmermann-Str. 18 Tel. +49 (0)7541-6000-0
D-88045 Friedrichshafen Fax +49 (0)7541-6000-11
Internet: www.esitron.de E-Mail: info@esitron.de

Doc. No: 040.174.07

# 1. General

## 1.1 How to use this manual

This manual covers technical data and properties of the esiMot compact motor in the versions:

esiMot_/_.4xx CANopen Interface

There are other user guides for the other versions of the esiMot.

This user manual contains safety instructions which have to be observed

## 1.2 Signs and Symbols

**DANGER**

This sign indicates a risk of injury, if the instructions are not strictly observed.

**Attention**

This sign indicates a risk of damage to the product or other pieces of equipment, if the instructions are not strictly observed.

**Note**

This sign indicates tips and useful information for easier handling of the product

## 1.3 Safety instructions

**DANGER**

Please read this manual carefully before using the esiMot.
In case of questions, please contact your dealer.

Inappropriate use of this product may cause damages to the motors or other pieces of equipment or products.
Make sure, the CE-Regulations are met by the complete system.
Depending on the application, other national regulations, like UL or DIN, have to be met as well.
Strictly observe safety precautions VDE0100, VDE0113, VDE0160; EN50178 and the accident prevention regulations of your employer's liability insurance. Other regulations may apply.

## 1.4 Changes

The information in this document is subject to changes without proclamation. The manufacturer accepts no liability in case of errors in this manual.

## 1.5 Copyright



## 1.6 Conclusion

There may be working functions in the controller which are not described in this manual. However, there are no claims to these functions for new delivered or repaired parts.
We have checked this manual with the hard- and software. Yet there might be differences. We appreciate all suggestions for improvement.

## 1.7 Room for notes

# 2. Preface

## 2.1 CANopen overview

CANopen is a networking system based on the CAN serial bus.
The CANopen specifications cover amongst others application layer and communication profile (CiA DS-301) as well as recommendations for cables and connectors (CiA DRP303-1).
CANopen is supplemented by a number of standardized device profiles. Applicable for drives is the DSP402.
A CANopen device can be divided into three parts:

- Communication interface and protocol software
- Object dictionary
- Process interface and application program

The communication interface and protocol software provide services to transmit and to receive communication objects over the bus. The object dictionary describes all data types, communication objects and application objects used in this device. It is the interface to the application software.
The elements required for the esiMot are described in this manual. However, if you're not using a controller with integrated CANopen handling or if you're not taking advantage of a protocol stack it's recommended to get at least the DS301.

☞ CAN Specifications are available from:

CAN in Automation (CiA) GmbH
Kontumazgarten 3
90429 Nuremberg
GERMANY
P: +49 (0) 911-928819-0
F: +49 (0) 911-928819-79
http://www.can-cia.org
Mail: headquarters@can-cia.org

☞ The most important part of a CANopen device is the object dictionary. It is essentially a grouping of objects accessible via the network in an ordered pre-defined fashion. The object dictionary of the esiMot is in chapter 12.
The overall function of a CANopen device is controlled by the state machine. The state machine and its transitions are described in chapter 10.3

☞ See also bibliography at the end of this document.

## 2.2 Abbreviations

*COB* Communication Object.
A unit of transportation in a CAN network. Data must be sent across a CAN network inside a COB. There are 2048 different COBs in a CAN network. A COB can contain at most 8 bytes of data.

*COB-ID* Each COB is uniquely identified in a CAN network by a number called the COB Identifier (COB-ID). The COB-ID determines the priority of that →COB for the →MAC sub-layer.

*Remote COB* A COB whose transmission can be requested by another device.

*CSDO* Client →SDO

*MAC* Medium Access Control.
One of the sub-layers of the data link layer in the CAN reference model that controls who gets access to the medium to send a message.

*NMT* Network Management.
One of the service elements of the application layer in the CAN reference model. The NMT serves to configure, initialise and handle errors in a CAN network.

*NODE-ID* The Node-ID of the NMT slave has to be assigned uniquely or 0. If 0, the protocol addresses all NMT slaves.

*PDO* Process Data Object.
The real-time data transfer is performed by PDOs. PDOs are transmitted in a non-confirmed mode.

*RPDO* Receive →PDO.

*SDO* Service Data Object.
The read and write access to entries of a device's object dictionary is performed with SDOs.

*SSDO* Server →SDO.

*SYNC* Synchronisation Object.

*TPDO* Transmit →PDO.
The producer sends a TPDO with a specific identifier, which corresponds to RPDOs of one or more consumers.

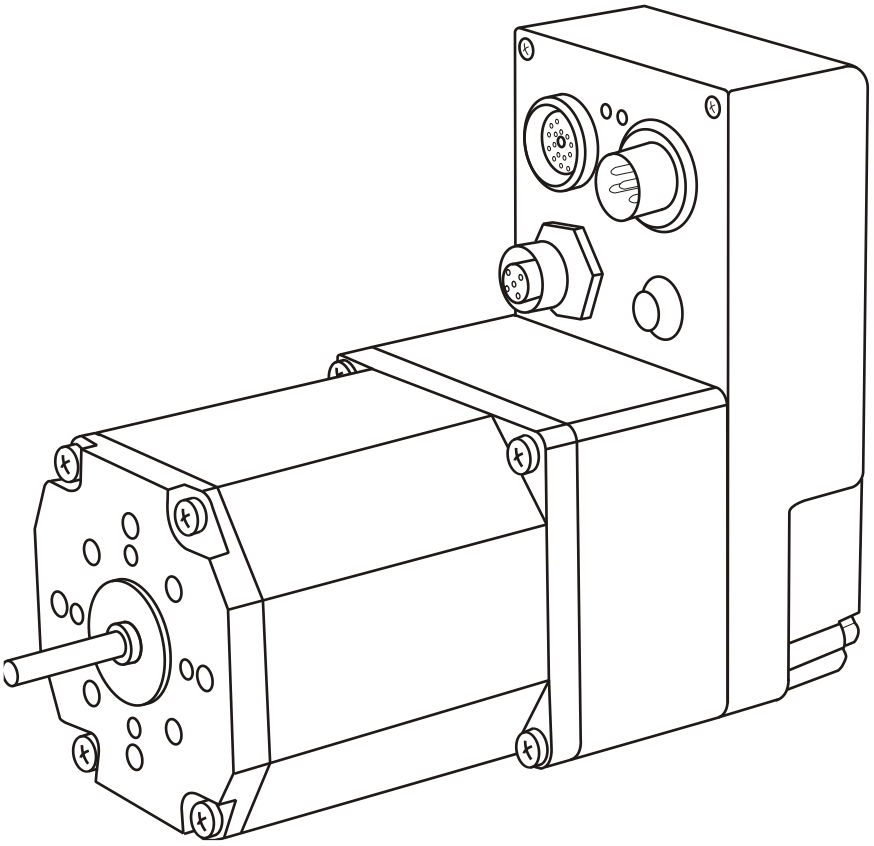
## 3. Abstract esiMot „all in one“

The esiMot digital compact motor can handle speed control as well as positioning with brushless Servo-motors.

The brushless drive with integrated 4Q controller facilitates the trouble-free connection to the common fieldbus CANopen. The fieldbus protocol CANopen DSP402 is supported.

There is a comfortable Windows-software available for setting-up operation, parameter setting, getting status information and diagnostics.

The esiMot is available with 5 different motors from 120W to 600W. Combinations with one to three stage planetary gear boxes offer output torque up to ca. 100 Nm.



**DANGER**

**Wiring and start-up of this device may be done by trained personnel only. Read the manual carefully. Note especially the installation requirements and notes for initial start-up. Non-adherence to the instructions will result in loss of warranty and liability on the part of the manufacturer.**
**This unit monitors internal operation conditions as well as on equipment side.**
**However, malfunctions caused by defective elements cannot be prevented in any case.**
**Personal danger has to be avoided at system side by interrupting the operating voltage through an emergency stop chain**

## 3.1 System description

The complete controller is in a compact housing which is flanged to the motor.

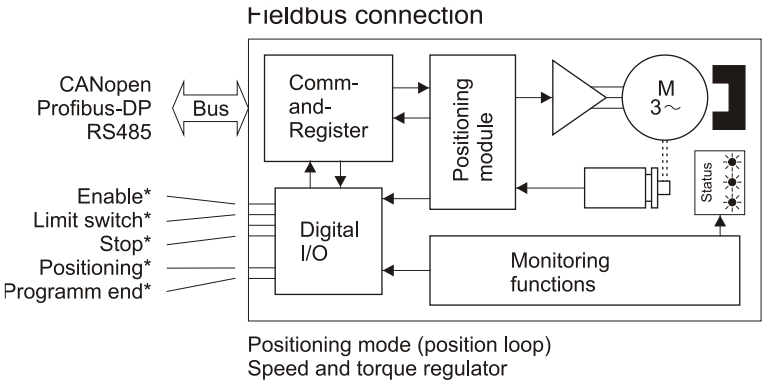
All electric connections to the complete system are done with 3 circular plug-in connectors, which face in direction of the axis (A-bearing shield).

Configuration can be done through the serial RS232 interface using the PC-Software “Servo Link”. Diagnostics are feasible via this interface as well. All parameters can be set through the fieldbus.

## 3.2 Interfacing the esiMot

The esiMot is controlled entirely via fieldbus.

Digital inputs and outputs may be assigned to desired functions. See table of functions on page 21ff.



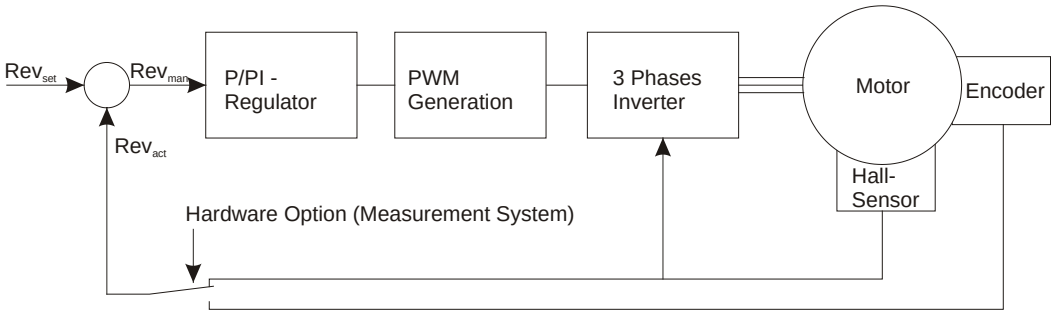

The available input and output functions are shown on page 21ff

# 4. Operating Modes

## 4.1 Mode 1: Speed regulation

The set value for the revolution speed is compared to the actual value. Load-dependent changes in revolution speed are compensated within current- and power-limits of motor.

The parameters for speed control (P- and I-gain) determine the reaction of the regulator.




## 4.2 Mode 4: Positioning

### 4.2.1 Absolute Positioning:

Move to an absolute target position (in increments). Positioning speed, acceleration ramp and deceleration ramp may be adjusted.

### 4.2.2 Relative Positioning:

Move a distance to the actual position (in increments). Positioning speed, acceleration ramp and deceleration ramp may be adjusted.

### 4.2.3 Jog-mode (Rapid traverse/creep speed):

Each positive edge on the input causes the drive to travel the amount of increments, which is adjusted in the parameters. Continuous input sets the drive to continuous move. The maximum travel is 2.140 million increments. After that the drive stops. A new positive edge starts it again. The speed is determined via parameters for rapid traverse and creep speed. The selection of what speed to take is determined by another input. Creep speed is used, if no input is assigned.

## 4.3 Homing mode

The homing modes of the DSP402 are supported. There is an additional parameter $209A_h$ to determine the homing deceleration.

☞

| **Note** |
| --- |
| A SDO write access to object $607A_h$ during homing is treated as successful termination of the homing run. Thus the user is able to assign an arbitrary home position. |

A detailed description of the homing modes you can find in chapter 10.7 on page 37.

## 4.4 Electronic Axle

To activate the operating mode "electronic axle" the inputs 3 and 4 have to be assigned to Clock/Channel A and Direction/Channel B.

There is the choice of clock/direction and channel A/Channel B in the user settings of the parameters. There it's also possible to invert the sense of rotation and to select the edge to detect. Furthermore, a gear-ratio may be selected.

☞

| **Note** |
| --- |
| This operating mode is selected by setting the operation mode to -1 via object $6060_h$. |

# 5. Brake

Some esiMot are equiped with a standstill brake. See order code on page 97.

There are three different ways how the brake is treated by the controller.

- Brake remains open as long as 24V logic supply is present
- Brake is open when the regulator is active (enabled)
- Brake management: The brake is opened when a drive command is processed and closes after the target is reached.

## 5.1 Brake always open

24V at the logic supply open the brake. There is no other condition for this.

Parameter setting:

*Motor / Standstill brake / ⊙ No*

**Danger**

**Hazard: Hanging loads may slump.**

**Note**

This setting is recommended for initial setting up of the machine only.

## 5.2 Brake open on activation

The brake is open as long as the regulator is active. The brake remais open between drive commands. This setting allows fast starts between consecutive drive commands.

Parameter setting:

*Motor / Standstill brake / ⊙ Yes*

*User / Brake management / ⊙ Deactivated*

## 5.3 Bake management – Clamping

This brake management works as clamping. After each positioning the brake is closed automatically.

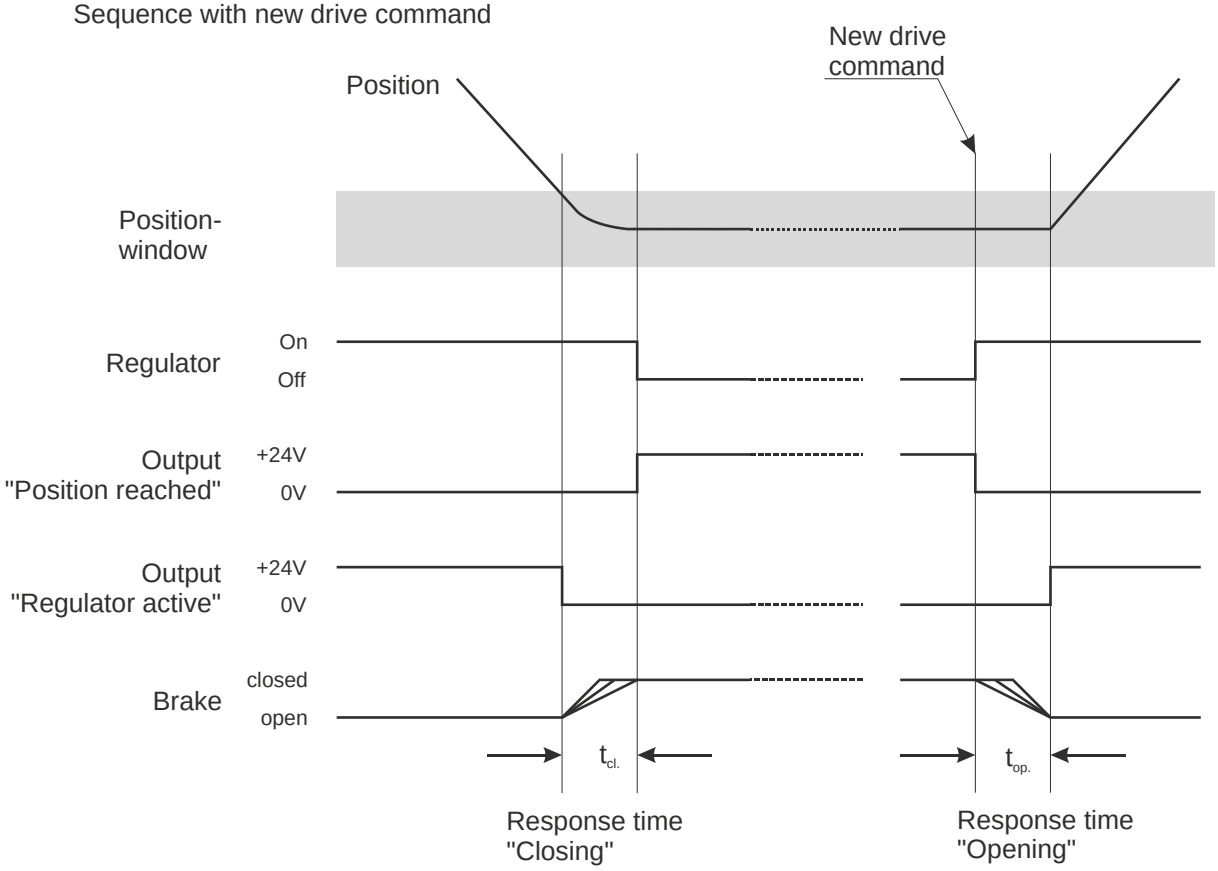
Parameter setting:

*Motor / Standstill brake / ⊙ Yes*

*User / Brake management / ⊙ Activated*

**Clamping**

Sequence with new drive command




## 5.4 Closing the brake on errors

Errors which cause the regulator to shut-off or a missing enable input cause the brake to close. Depending on the type of error the controller reacts with different modes.

☞

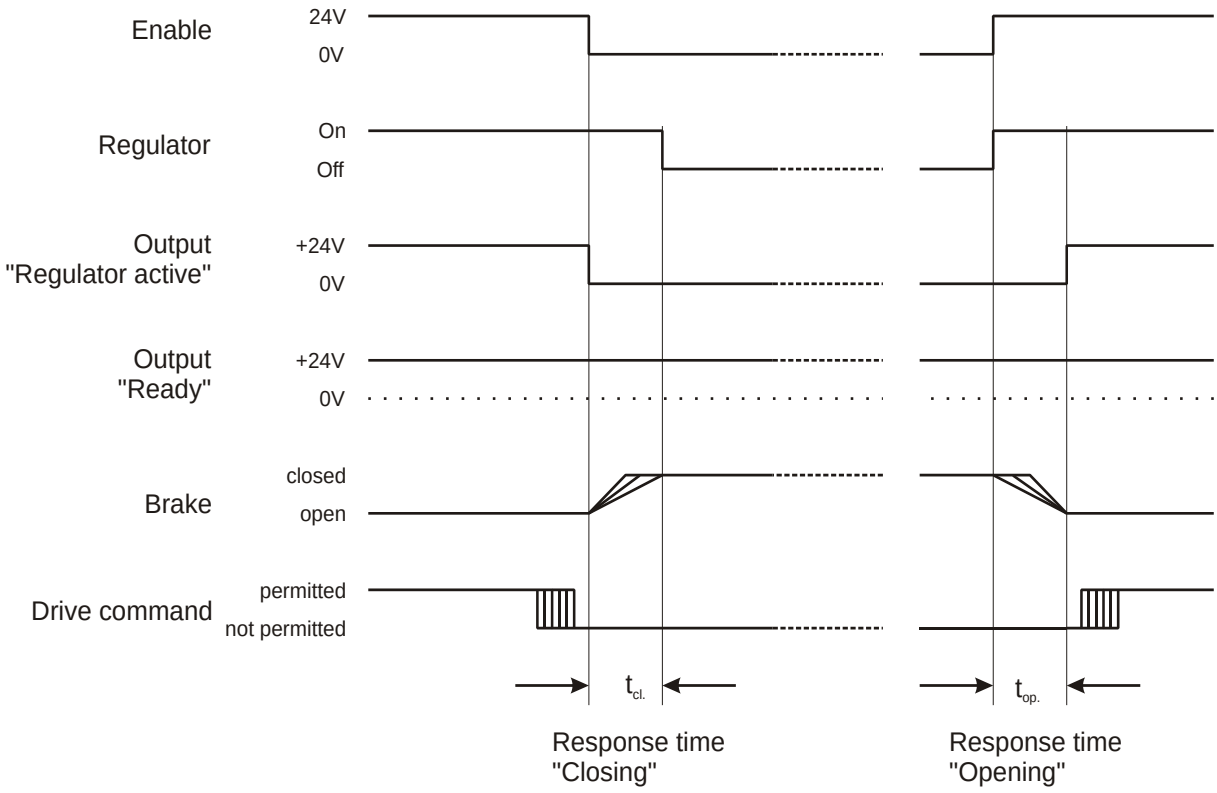
| **Note** |
| --- |
| Not every error closes the standstill brake. The table in chapter 13 on page 80 shows the appropriate brake mode. |

### 5.4.1 Brake-mode A:

The standstill brake and the output „Brake“ is activated with the falling edge of the enable input. The regulator is disabled after the response-time of the brake.

The regulator is enabled immediately with the rising edge of the enable input. A drive command may only be given after the response-time. The user may check the output „Regulator state“.

**Brakemode A**

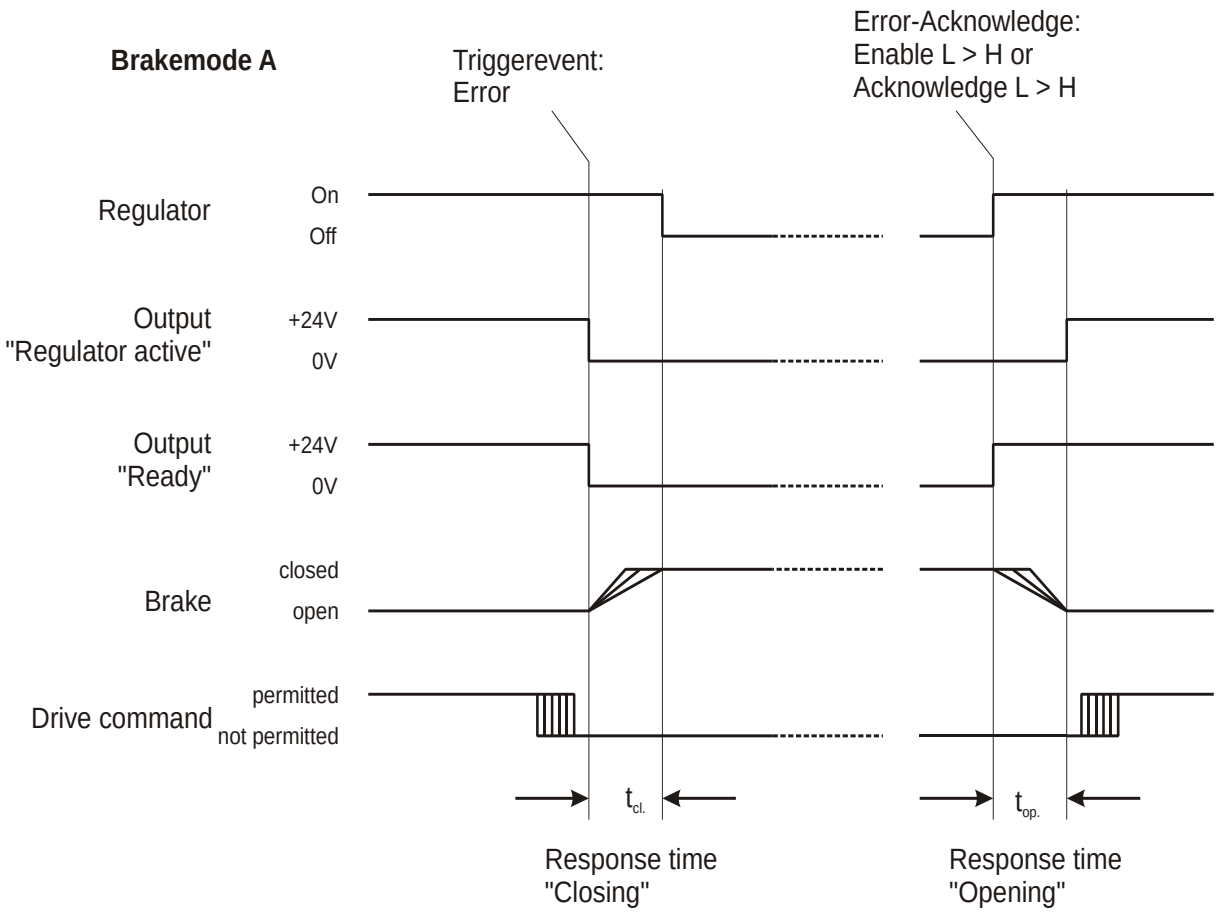



The standstill brake and the output „Brake“ is activated with the error. The motor regulator is disabled after the response-time „Closing“.

The regulator is enabled immediately with error acknowledgement. A drive command may only be given after the response-time. The user may check the output „Regulator state“.

☞

| **Note** |
| --- |
| Not every error activates the standstill brake. The table in chapter 13 on page 80 shows the appropriate reaction and brake-mode |


Brakemode A
Triggerevent:
Error
Error-Acknowledge:
Enable L > H or
Acknowledge L > H
Regulator
On
Off
Output
"Regulator active"
+24V
0V
Output
"Ready"
+24V
0V
Brake
closed
open
Drive command
permitted
not permitted
$t_{cl.}$
$t_{op.}$
Response time
"Closing"
Response time
"Opening"

### 5.4.2 Brake-mode B:

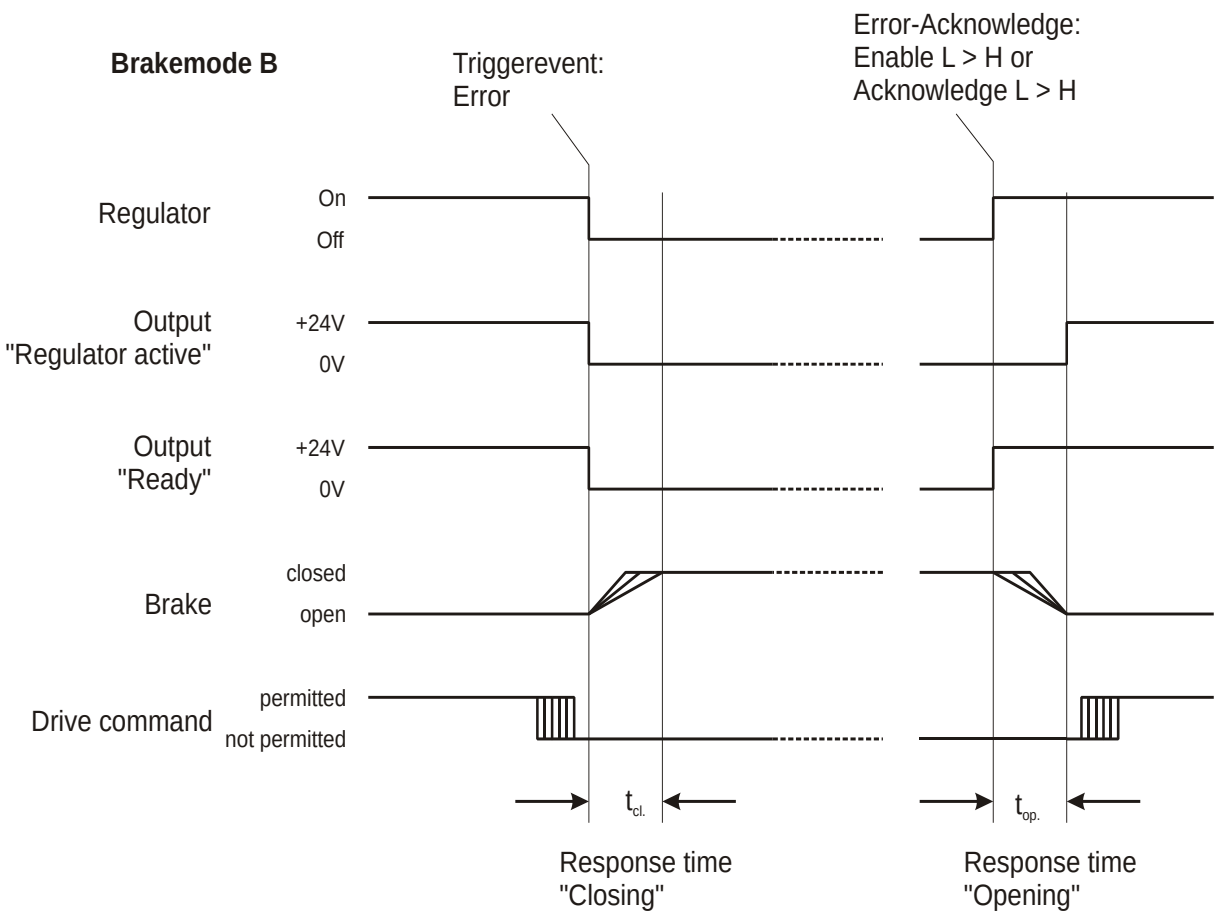
The standstill brake and the output „Brake“ is activated with the error. The motor regulator is disabled with the occurrence of the error.

The regulator is enabled immediately with error acknowledgement. A drive command may only be given after the response-time. The user may check the output „Regulator state“.

☞

**Note**

Not every error activates the standstill brake. The table in chapter 13 on page 80 shows the appropriate reaction and brake-mode



# 6. Parameter Setting

The PC-Software features different password protected levels to adjust parameters. Those parameters which require access-level 1 to adjust are marked with "[1]" and those which require access-level 2 are marked with "[2]". Parameters marked with "[3]" are for information only and may be adjusted by the manufacturer only.

The correct adjustment of parameters depends on the motor-type and the fitted options.

## 6.1 Electronics configuration

| | |
|---|---|
| Internal threshold values: | |
| Warning temperature output stage[3] | |
| Error temperature output stage[3] | |
| Warning temperature motor[3] | |
| Error temperature motor[3] | |
| Undervoltage motor[3] | |
| Overvoltage motor[3] | |
| Limits | |
| Maximum current[3] | (Maximum the adjusted hardware limits) |
| Nominal current[3] | (Maximum the adjusted hardware limits) |
| Ballast threshold[3] | (Maximum the adjusted hardware limits) |
| Electronics data[3] | Electronic name plate |
| Hardware configuration[3] | I/Os and Fieldbus |

## 6.2 Motor configuration

| | |
|---|---|
| Basic Data | |
| Number of pole pairs*[3] | 1 – 6 |
| Peak current[3] | 14 A (≤ hardware limits) |
| e.m.f. constant[3] | [mV/1/min] |
| Motor overload time*[3] | [s] |
| Motor-PTC*[3] | Existing / not existing |
| Commutation mode*[3] | Mode A/B |
| Axis resolution*[3] | [incr./rev] |
| Nominal current*[3] | 0 – 10 A (≤ hardware limits) |
| Nominal voltage*[3] | 24 – 60V (≤ hardware limits) |
| Nominal speed*[3] | [rpm] |
| Actual value acquisition | |
| Speed*[3] | Via Hall sensors / Encoder |
| Travel *[3] | Via Hall sensors / Encoder |
| Motor data [3] | Electronic name plate |
| Direction of control | |
| Speed regulator | Inverted / normal |
| Position control | Inverted / normal |
| Limits | |
| Maximum speed *[3] | Maximum the adjusted hardware limits |
| Standstill brake | |
| Brake*[2] | [Yes/No] |
| Response time "Close"*[3] | [msec] |
| Response time "Opening"*[3] | [msec] |

**Note**

Setting the brake to “no” keeps it open as long as 24V are supplied.

**DANGER**

This setting may cause hanging loads to slump when the regulator is switch off while 24V are supplied.

## 6.3 User configuration

| | |
|---|---|
| Basic Data | |
| Bus address | 1 - 255 |
| Maximum speed at 100%*[2] | [rpm] |
| Maximum current [2] | [%] of peak current |
| Working mode*[2] | Speed regulation, Positioning, Electronic Axle |
| Reference run[2] | required / not required |
| Operating Voltage [2] | [V] |
| Rotational direction | |
| Speed regulator[2] | Inverted / normal |
| Position control[2] | Inverted / normal |
| Response to errors | |
| Stop-mode on errors[1] | Ramp stop / immediate stop |
| Deceleration ramp[1] | [msec] |
| Reference run | |
| Acceleration ramp[2] | [msec] |
| Deceleration ramp[2] | [msec] |
| Search speed[2] | [rpm] |
| Positioning speed[2] | [rpm] |
| Offset [2] | [incr.] |
| Torque limit for "Homing to block" | [%] |
| Reference run mode[2] | See table on page 36 |
| Range of Travel* | |
| Positive Direction | [incr.] |
| Negative Direction | [incr.] |
| Electronic axle | |
| Type [2] | Clock-Direction / Channel A – Channel B |
| Direction of rotation [2] | Inverted / Normal |
| Edge detection [2] | Positive edge / Negative edge / Both edges |
| Ratio [2] | 1 : [0,001 ... 9999,999] |

☞ **Note**

Bus address and baud rate may only be adjusted via software if not optional DIP-switches are fitted or all, i.e. bus – address and baud rate switches are set to off.

☞ **Note**

There is no error message and to blink code when reaching the limit of travel. The drive just stops. To detect this state check status via RS232 or fieldbus.

☞ **Note**

The limits of travel (Software limit switch) are effective after reference-run only. This is also valid if "No reference-run required" is selected in the parameters. Remedies: Use reference-run mode 35 (Set to zero) after power up.

☞ **Note**

The limits of travel are effective only for positioning (Jog-mode also). They are not valid in speed regulation mode.

## 6.4 Input / Output Functions

Available Input Functions:

| Name: | Dependence: | Remarks: |
|---|---|---|
| Reference-switch | Level | Required for referencing. See chapter 0 on page 36. |
| Limit switch + | Level | 0V at this input indicates that the limit switch in positive direction has been reached. (Use brake contact switch). 24V at input indicate travel inside admissible range. |
| Limit switch – | Level | 0V at this input indicates that the limit switch in negative direction has been reached. (Use brake contact switch). 24V at input indicate travel inside admissible range. |
| Jog + | Level | A short pulse at this input causes the drive to move in positive direction for the number of steps which are set in the parameters. A permanent input causes the drive to go to continuous mode with the speed determined by the Rapid traverse/Creep-Speed input. |
| Jog - | Level | A short pulse at this input causes the drive to move in negative direction for the number of steps which are set in the parameters. A permanent input causes the drive to go to continuous mode with the speed determined by the Rapid traverse/Creep-Speed input |
| Rapid traverse | Level | 0=Creep speed; 1=Rapid traverse for jog-mode |
| Clock / Channel A | Edge | Available on input 3 only |
| Direction / Channel B | Edge | Available on input 4 only |

☞

**Note**

Independent of the direction of travel, each falling edge at any limit-switch input stops the drive. Reason for this procedure is the risk of confused limit switches during installation shall not cause damages.

☞

**Note**

If the same function is assigned to different inputs, the first input becomes effective.

Available Output Functions:

| Name: | Function |
|---|---|
| Ready | 24V : Ready for operation<br>0V: Error Message |
| Regulator active | 24V : Regulator working<br>0V: Regulator disabled |
| Error | 24V: Error message<br>0V: Ready for operation |
| Warning | 24V: Motor or Controller has reached critical temperature, but still working.<br>0V: No critical temperature reached. |
| Position reached | The drive has reached the desired position. |
| Axle in motion | 24V: The drive is in motion; no new positioning command is accepted<br>0V: The drive is ready for new commands. |
| Ballast active | 24V: Ballast circuit is activated<br>0V: Ballast circuit is not activated. |

## 6.5 Regulator

| Note |
|---|
| Modified parameters need to be transferred to the esiMot.<br><br>Press button  |

### 6.5.1 Torque regulator

| | | |
|---|---|---|
| P - portion | 1 – 1000 | The proportional gain of the torque regulator. |
| I - portion | 1 – 1000 1/s | Integral contribution of the control loop of the torque regulator. |
| I – limitation | 0 – 100 % | Limitation of the integral contribution of the torque regulator |

| Note |
|---|
| The torque regulator is trimmed to the motor. Changes are not recommended. |

### 6.5.2 Speed regulator

| | | |
|---|---|---|
| P - portion[1] | 1 – 1000 | The proportional gain of the speed regulator |
| I - portion[1] | 1 – 1000 1/s | Integral contribution of the control loop of the speed regulator |
| I - limitation[1] | 0 – 100 % | Limitation of the integral contribution of the speed regulator |
| Ramp time to Vmax[1] | [msec] | Acceleration time from standstill to Vmax in the working mode „speed regulator“ |
| Speed window | [rpm] | Permitted speed tolerance in speed regulation mode |

### 6.5.3 Position controller

| | | |
|---|---|---|
| P - portion[1] | 1 – 1000 | The proportional loop gain of the positioning controller. |
| Target window[1] | Increments | Permissible error between set point and current position. |
| Lag error[1] | Increments | Permissible distance between calculated and actual position during the movement. |
| Store position safe against power outage? [2] | Yes/no | The drive features the possibility to safe the actual position when powering down to avoid the need of repeatedly reference-runs.<br>☞ See procedure described in chapter 0<br>☞ The motor may not turn anymore or the stored position is not defined. Make sure the position is not changed when it’s switched off.<br>☞ The optional absolute encoder provides the position at any time. Set “store position safe against power outage” to “no” with absolute encoder. |

The target position window respectively the target velocity has to be reached to trigger the according CAN message.

### 6.5.4 Procedure to store position safe against power outage

The motor must not turn anymore. Switch off motor power supply first. Wait for the error message "undervoltage motor". Now the position is stored. The logic power supply may be switched off now.

## 6.6 Jog

| | | |
|---|---|---|
| Acceleration ramp [1] | [msec] | Acceleration in Jog mode. The stated ramp time represents the time from standstill to the max speed adjusted by the user. (Servo Link: Parameter/User/Max. speed at 100%) (See chapter 6.3). |
| Deceleration ramp [1] | [msec] | Deceleration in jog mode. The stated ramp time represents the time from the max speed adjusted by the user. (Servo Link: Parameter/User/Max. speed at 100%) to standstill. (See chapter 6.3). |
| Creep speed [1] | [rpm] | Speed for movements of tip increments or constant speed after completion of the acceleration ramp, as long as input „Rapid traverse“ is not active. |
| Rapid traverse [1] | [rpm] | Speed for movements of tip increments or constant speed after completion of the acceleration ramp, as long as input „Rapid traverse“ is active. |
| Tip increments[1] | [Incr.] | Number of increments the drive moves per each pulse on the input „Jog+“ or „Jog-“.<br>☞ If the number of tip increments is set to 0, there is no delay before the drive goes to continuous move. The rising edge of the input starts the movement with the speed determined by input “Rapid traverse”. |

[1] access level 1 required to modify this parameter.

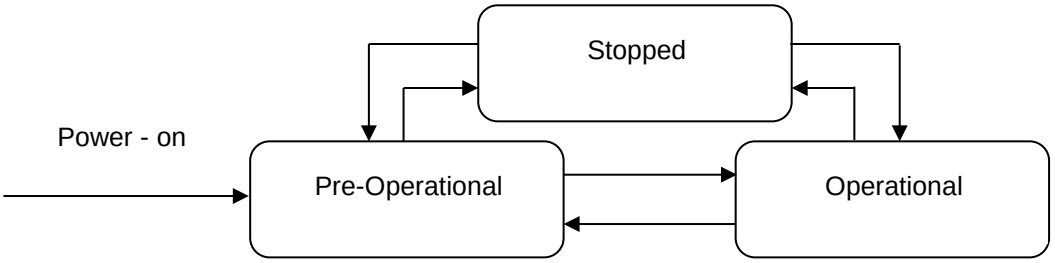
# 7. Network Management

The Network Management or NMT provides functions for initialisation, configuration and monitoring of all subscribers on the bus. One device is needed in the network to work as a NMT master, the CANopen master. All other devices are called slave and may be monitored by the master.

Each slave has a Node-ID in the range 1-127 to be clearly identifiable. Node-ID 0 may not be used because it is reserved for broadcasts, which means messages to all subscribers.

## 7.1 States

The following diagram shows the possible states and their transitions.




The different states are indicated with the bus LED. The indicator states and flash rates are implemented according to DR-303-3.[1]

## 7.2 Slave Monitoring

### 7.2.1 Slave monitoring through Node Guarding

Node Guarding is a cyclic monitoring of the state of a slave through remote frame by the NMT master. This method is not recommended for new developments.

### 7.2.2 Heartbeat monitoring

The slaves cyclically transmit their state to the master and the other consumers. The time between two telegrams is set in the parameter index 1017h „Producer Heartbeat Time“ in multiples of milliseconds.

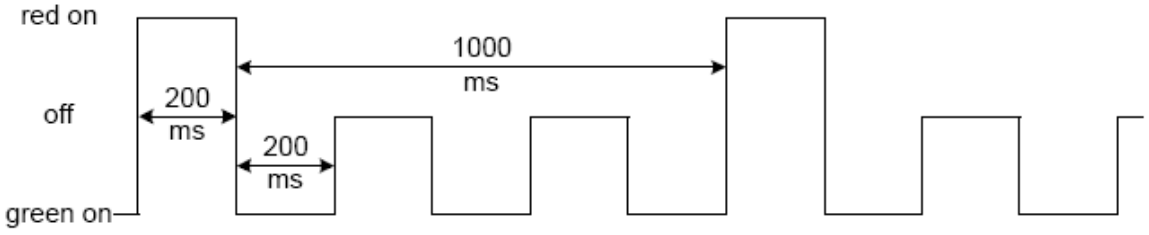
## 7.3 LED Indicator

The actual operating state is indicated with the bus LED. The indicator states and flash rates are according to DR-303-3.[2]

The bus LED is a bi-colour LED which means that two states may be shown at the same time. The following sketch shows an example:

The error LED is red and indicates the status of the physical CAN layer. In this example it shows a NMT warning.

The Run LED is green and indicates the status of the CANopen network state machine. In this example it shows the state „PRE-OPERATIONAL“.



# 8. Installation

## 8.1 esiMot installation

The surface of the esiMot may reach temperatures 30°C above ambience, depending on load conditions and ventilation. However, on heavy load conditions, the motor will get much hotter and will heat up additionally the esiMot.

The esiMot has to be mounted in a way a sufficient air flow is ensured.

**Danger**

**Make sure, the esiMot is not hot, before touching.**

## 8.2 Ballast resistor

An external ballast resistor is required only on heavy load conditions to the esiMot. Many applications do not require an external ballast resistor.

The mounting of the ballast resistor has to be in a well ventilated place. The ballast resistor has to be mounted in a way a sufficient air flow is ensured.

**Danger**

**The surface of the brake resistor can reach temperatures of up to 450°C.**
**The resistor may only be mounted in areas where heat can't cause fire. Take care that no flammable or heat-sensitive materials are close to the resistor.**

## 8.3 Electrical Installation

### 8.3.1 Safety advice

Please read the manual before using this product.

**Danger**

**This unit monitors internal operation conditions as well as on equipment side.**
**However, malfunctions caused by defective elements cannot be prevented in any case.**
**Personal danger has to be avoided at system side by interrupting the operating voltage through an emergency stop chain.**

We assume, as an expert, you are familiar with the relevant safety regulations, the accident prevention regulations of the employer's liability insurance company and the DIN regulations and that you can use and apply them.

**Danger**

**Depending on the application, it might be necessary to implement additional safety equipment in order to get a fail-safe system.**

### 8.3.2 Mounting instructions

Wiring and start-up of this device may be done by trained personnel only.

Keep wires as close to housing or enclosure as possible.

Route signal and power cables separately. The shield / PE connection must be a short cable (1,5mm²) and make full contact and conducts well to the housing or enclosure.

**Danger**

**Make sure, all equipment is powered down, during installation of the esiMot.**

As a general rule, the motor shall run freely, i.e. without load, for initial start-up.

### 8.3.3 Max voltage

**Attention**

**Exceeding the maximum admissible voltage for the motor power supply (depending on type of esiMot) or 30V for the logic power supply will destroy the esiMot.**

### 8.3.4 EMC

The esiMot compact motor is compliant to applicable CE-regulations. However, emission of high frequency electromagnetic waves can cause interference to other pieces of equipment in some cases.

This might require additional remedial action. The use of EMI-filters is recommended.

**Danger**

**Persons with a cardiac pacemaker may be affected by the magnetic radiation of the motor.**
**Avoid exposure to magnetic radiation.**

## 8.4 Repairs

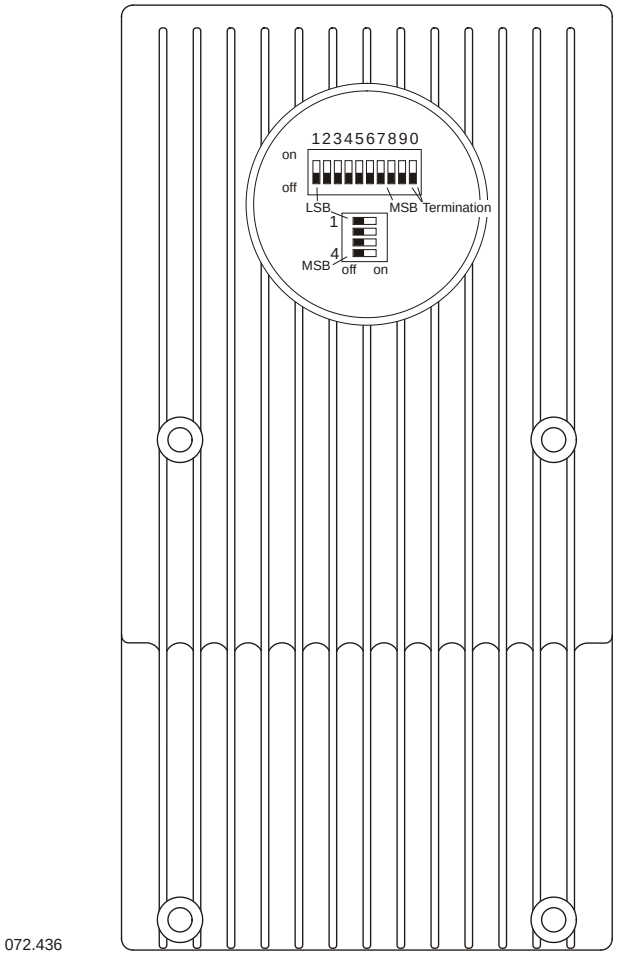
Repairs may be conducted only by the manufacturer or authorised personnel.

**Danger**

**Unauthorised opening and inappropriate repairs may cause substantial risk on injury for the user or damage to equipment.**

# 9. CANopen commissioning

## 9.1 Bus address, Bus termination and Baud rate




The Bus address is adjustable through the DIP-switches on the backside on the esiMot. The address range is 1 – 127.

☞ **Note**

Each address used in a CANopen network must be unique and may not be used by other devices.

The Bus termination is activated through DIP-switches 9 and 10. To enable the termination resistor both have to be set to ON.

The Baud rate is adjusted via the 4-pin DIP-switch.

| switch 1 | switch 2 | switch 3 | switch 4 | baud rate [kbit] |
|---|---|---|---|---|
| OFF | OFF | OFF | OFF | 50<br>100 |
| OFF | ON | OFF | OFF | 125 |
| ON | ON | OFF | OFF | 250 |
| OFF | OFF | ON | OFF | 500 |
| ON | OFF | ON | OFF | 1000 |

| ☞ | **Note** |
|---|---|
| | In addition to this, the CAN – Bus address is adjustable through the PC-software „Servo Link“. (Only if all DIP-switches are not assembled or set to OFF) |

## 9.2 CANopen and RS-232

The programming and parameter interface of the esiMot is always active and may be used. Reading of parameters and the diagnostic mode do not interfere with the current state of the controller. To transmit parameters from PC to esiMot the regulator needs to be switched off. This means, if a drive command is executed it will be aborted. The state machine is reset with the command „Disable Voltage“.

| ☞ | **Note** |
|---|---|
| | CANopen and RS-232 are active at the same time! |

# 10. Device Profile Drives and Motion Control (DSP 402)

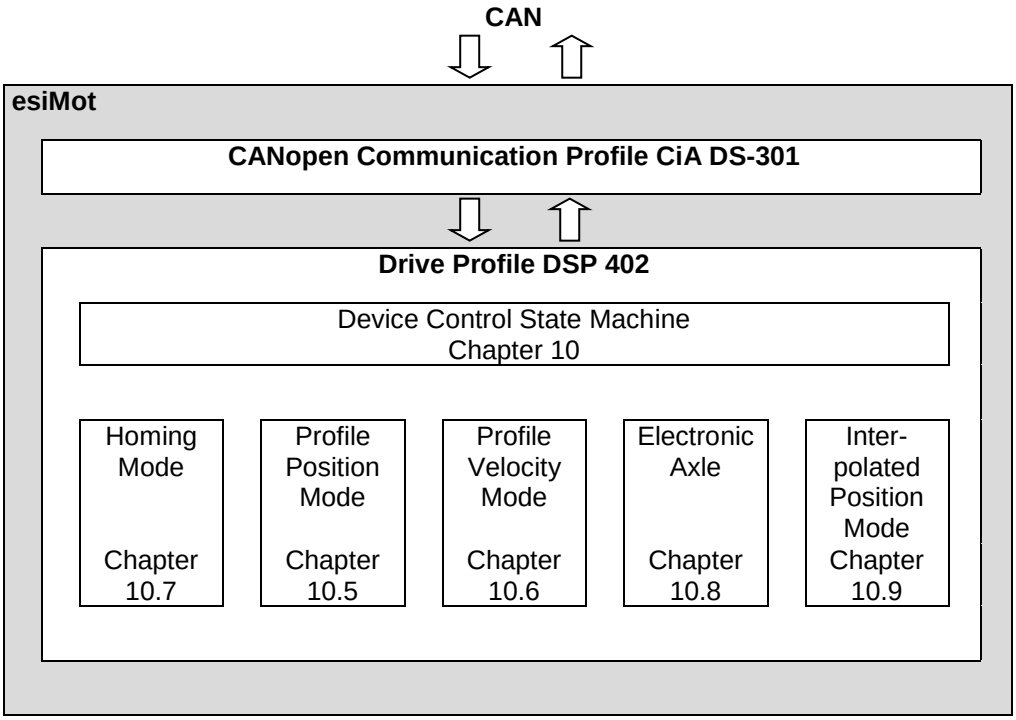
## 10.1 Scope

The device profile DSP 402 is a very comprehensive profile and may contains up to 200 objects. In addition to this communication parameters need to be implemented as well.

☞

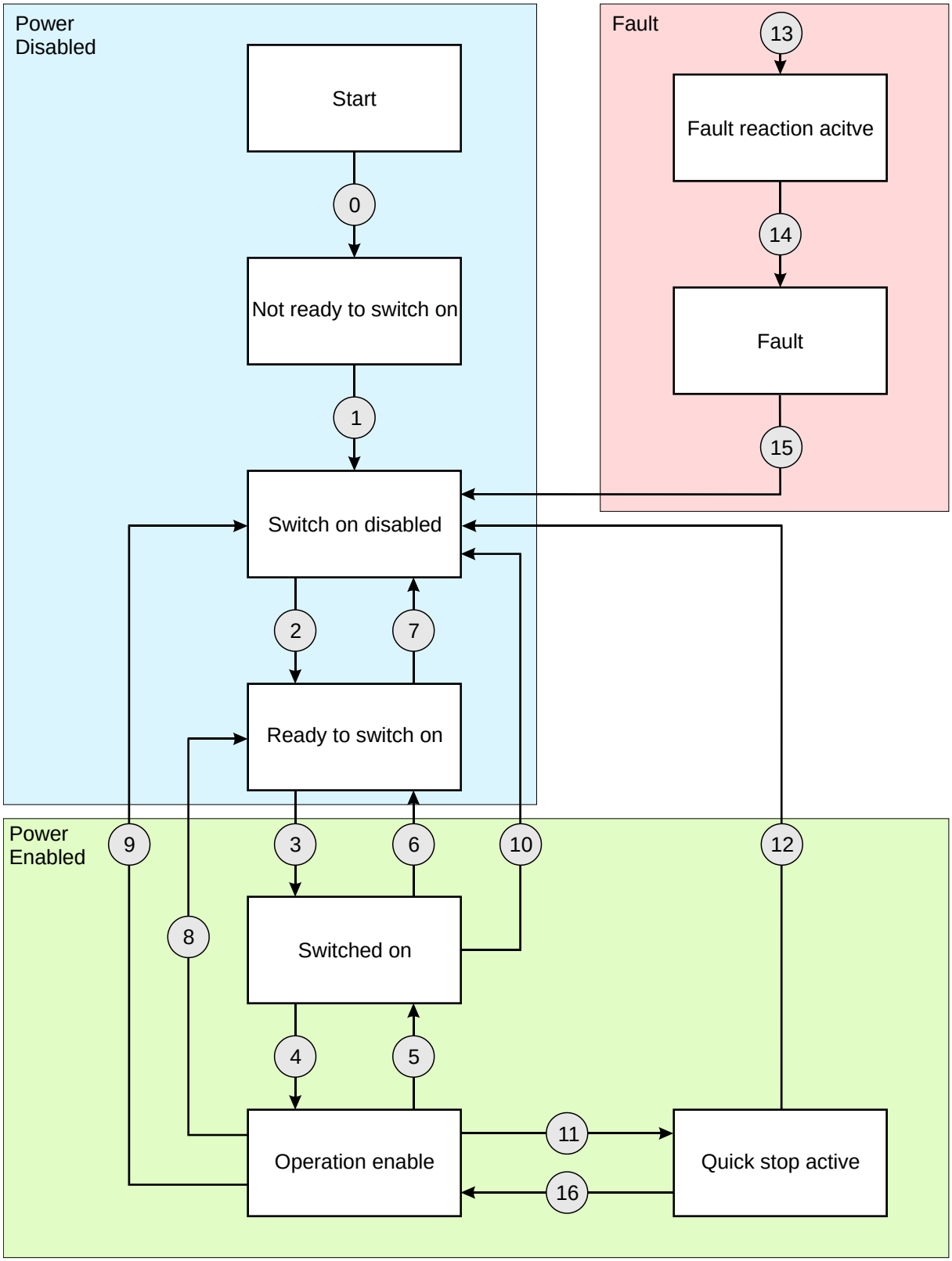
| **Note** |
| --- |
| The DSP 402 is a draft standard proposal. Its implementation in the esiMot may differ from profiles of other manufacturers. |

## 10.2 Drive Profile Architecture




The DSP402 defines a state machine and different modes of operation, which are shown in more detail on the next pages.

### 10.3 The state machine according to DSP 402




**If a command is received which causes a change of state, this command must be processed completely and the new state attained before the next command can be processed.**

### 10.3.1 Transitions

| | |
|---|---|
| *State Transition 0* | START >> NOT READY TO SWITCH ON<br>Event: Reset<br>Action: The drive self-tests and/or self-initializes. |
| *State Transition 1* | NOT READY TO SWITCH ON >> SWITCH ON DISABLED<br>Event: The drive has self-tested and/or self-initialized successfully.<br>Action: Activate communication. |
| *State Transition 2* | SWITCH ON DISABLED >> READY TO SWITCH ON<br>Event: “Shutdown” command received from host.<br>Action: none |
| *State Transition 3* | READY TO SWITCH ON >> SWITCHED ON<br>Event: “Switch On” command received from host<br>Action: The power section is switched on if it is not already switched on. |
| *State Transition 4* | SWITCHED ON >> OPERATION ENABLE<br>Event: “Enable Operation” command received from host.<br>Action: The drive function is enabled. |
| *State Transition 5* | OPERATION ENABLE >> SWITCHED ON<br>Event: “Disable Operation” command received from host.<br>Action: The drive operation will be disabled. |
| *State Transition 6* | SWITCHED ON >> READY TO SWITCH ON<br>Event: “Shutdown” command received from host.<br>Action: The power section is switched off. |
| *State Transition 7* | READY TO SWITCH ON >> SWITCH ON DISABLED<br>Event: “Quick Stop” and “Disable Voltage” command received from host.<br>Action: None. |
| *State Transition 8* | OPERATION ENABLE >> READY TO SWITCH ON<br>Event: “Shutdown” command received from host.<br>Action: The power section is switched off immediately and the motor is free to rotate if unbraked. |
| *State Transition 9* | OPERATION ENABLE >> SWITCH ON DISABLED<br>Event: “Disable Voltage” command received from host<br>Action: The power section is switched off immediately and the motor is free to rotate if unbraked. |
| *State Transition 10* | SWITCHED ON >> SWITCH ON DISABLED<br>Event: “Disable Voltage” or “Quick Stop” command received from host.<br>Action: The power section is switched off immediately and the motor is free to rotate if unbraked. |
| *State Transition 11* | OPERATION ENABLE >> QUICKSTOP ACTIVE<br>Event: “Quick Stop” command received from host.<br>Action: The quick stop function is executed. |
| *State Transition 12* | QUICK STOP ACTIVE >> SWITCH ON DISABLED<br>Event: “Quick Stop” is completed or “Disable Voltage” command received from host. This transition is possible, if the Quick-Stop-Option-Code is different from “5” (stay in the state “Quick Stop Active”)<br>Action: The power section is switched off. |
| *State Transition 13* | All states >> FAULT REACTION ACTIVE<br>Event: A fault has occurred in the drive.<br>Action: Execute appropriate fault reaction. |
| ☞ | “Fault occurred” implies that a fault in the drive has occurred. This causes a state transition to the state FAULT REACTION ACTIVE. In this state the device will execute the designated fault reaction. After the completion of this fault reaction the device will switch to the state FAULT. This state can only be left by the command “Fault Reset”, but only if there is no active fault anymore. |

*State Transition 14* FAULT REACTION ACTIVE >> FAULT
Event: The fault reaction is completed.
Action: The drive is disabled. The power section may be switched off.

*State Transition 15* FAULT >> SWITCH ON DISABLED
Event: “Fault Reset” Command received from host.
Action: A reset of the fault condition is carried out if no fault exists currently on the drive. After leaving the state FAULT the bit “Fault Reset” of the controlword has to be cleared by the host.

*State Transition 16* QUICK STOP ACTIVE >> OPERATION ENABLE
Event: “Enable Operation” command received from host. This transition is possible if the Quick-Stop-Option-Code is 5, 6, 7 or 8.
Action: The drive function is enabled.

☞ If a command is received which causes a change of state, this command must be processed completely and the new state attained before the next command can be processed.

### 10.3.2 Overview to transition commands

☞ All of this commands refer to the controlword ($6040_h$)

*Shutdown* Transitions 2, 6 and 8
State transition from SWITCH ON DISABLED; SWITCHED ON or OPERATION ENABLE
to READY TO SWITCH ON

*Switch on* Transition 3
State transition from READY TO SWITCH ON
to SWITCHED ON

*Disable voltage* Transitions 7, 9, 10 and 12
State transitions from READY TO SWITCH ON; OPERATION ENABLE; SWITCHED ON or QUICK STOP ACTIVE
to SWITCH ON DISABLED

☞ Transition 12 depends on the Quick-Stop-Option-Code

*Quick stop* Transitions 7, 10
State transitions from READY TO SWITCH ON or SWITCHED ON
to SWITCH ON DISABLED
Transition 11
State transition from OPERATION ENABLE
to QUICK STOP ACTIVE

*Disable operation* Transition 5
State transition from OPERATION ENABLE
to SWITCHED ON

*Enable operation* Transition 4
State transition from SWITCHED ON
to OPERATION ENABLE
Transition 16
State transition from QUICK STOP ACTIVE
to OPERATION ENABLE

*Fault reset* Transition 15
State transition from FAULT
to SWITCH ON DISABLED

☞ After leaving the state FAULT the bit “Fault Reset” of the controlword has to be cleared by the host.

## 10.4 Factor group

The factor group is not supported by esiMot. Objects $6089_h$ to $608E_h$ are implemented in the CANopen Master for display purposes.

The following units apply:

| | |
|---|---|
| Position Units | Increments. |
| Velocity Units | rpm |
| Acceleration Units | rpm/s |

## 10.5 Profile Position Mode

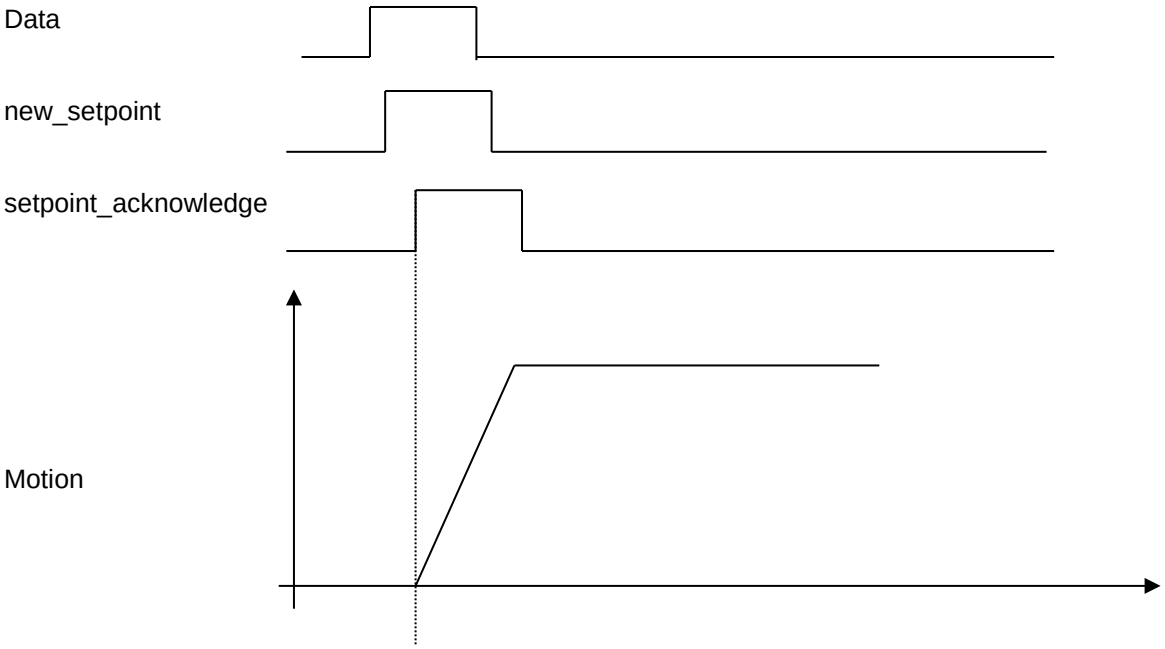
### 10.5.1 Control of drive commands

The controller supports the mode „Single drive command“ as well as „Sequence of drive commands“.

- Single drive command:

  When the drive has reached its target position it stops and communicates this to the master by setting the bit „target_reached“.

The transmission of the drive data in mode „Single Drive Command“ follows this scheme:




The master sets the ramp data (e.g. speed, acceleration and deceleration). When finished the master signals this to the controller with a rising edge of the bit „new_setpoint“. Now the controller checks the data. This is acknowledged with „setpoint_acknowldege“ if the data check is successful and the drive begins to move.

☞

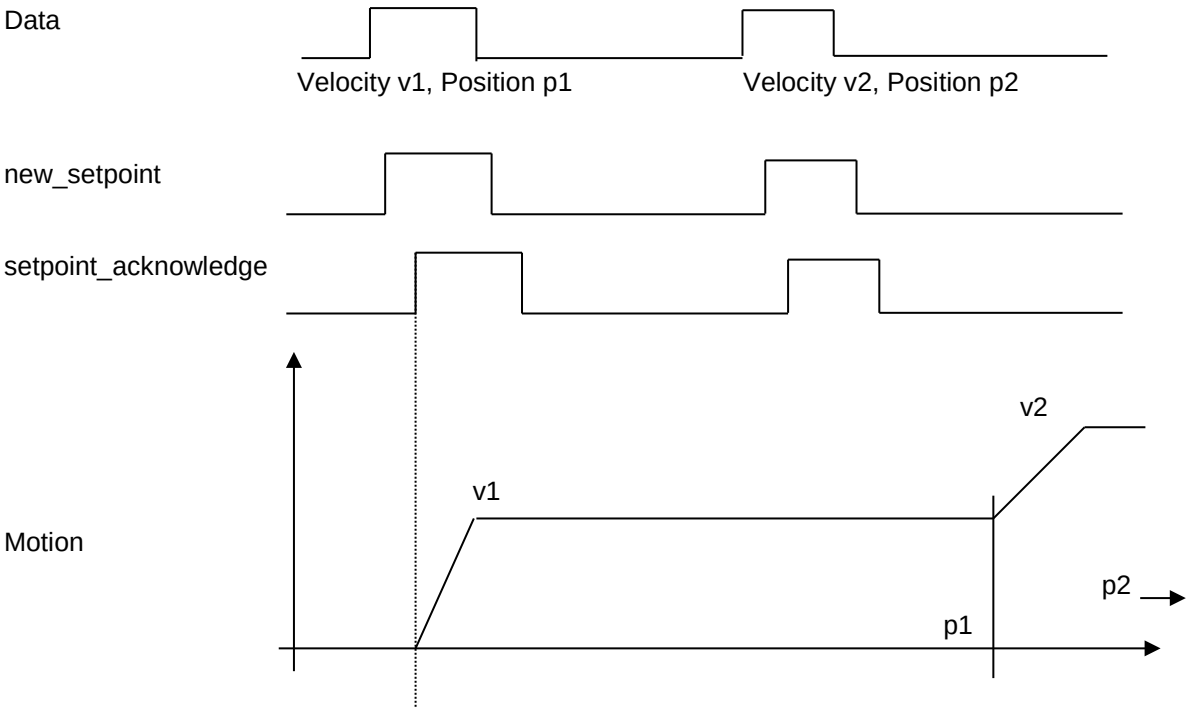
| **Note** |
|---|
| The master needs to reset the bit „new_setpoint“ prior to the transmission of new drive data. |

- Sequence of drive commands:

  When the drive has reached its first target position it's aiming immediately the next one without stopping in between.

  Another option is the use of the bit „Change Set Immediately". As soon as the command is sent, the new drive command is processed.

  The transmission of the drive data in mode „Sequence of Drive Commands"
  without the Bit change_set_immediately follows this scheme:




This sequence of drive commands is achieved when a drive command without the Bit „change_set_immediately" is given as soon as the first drive command is being processed. The drive moves to position p1 and proceeds immediately with the next drive command which is already in the memory. There is no stop in between. The position p2 is irrelevant. If p2 < p1 a ramp to the negative velocity is calculated in order to reach position p2.

If more than one position is transmitted as long as the drive is in motion, the last one sent is valid for the next positioning.

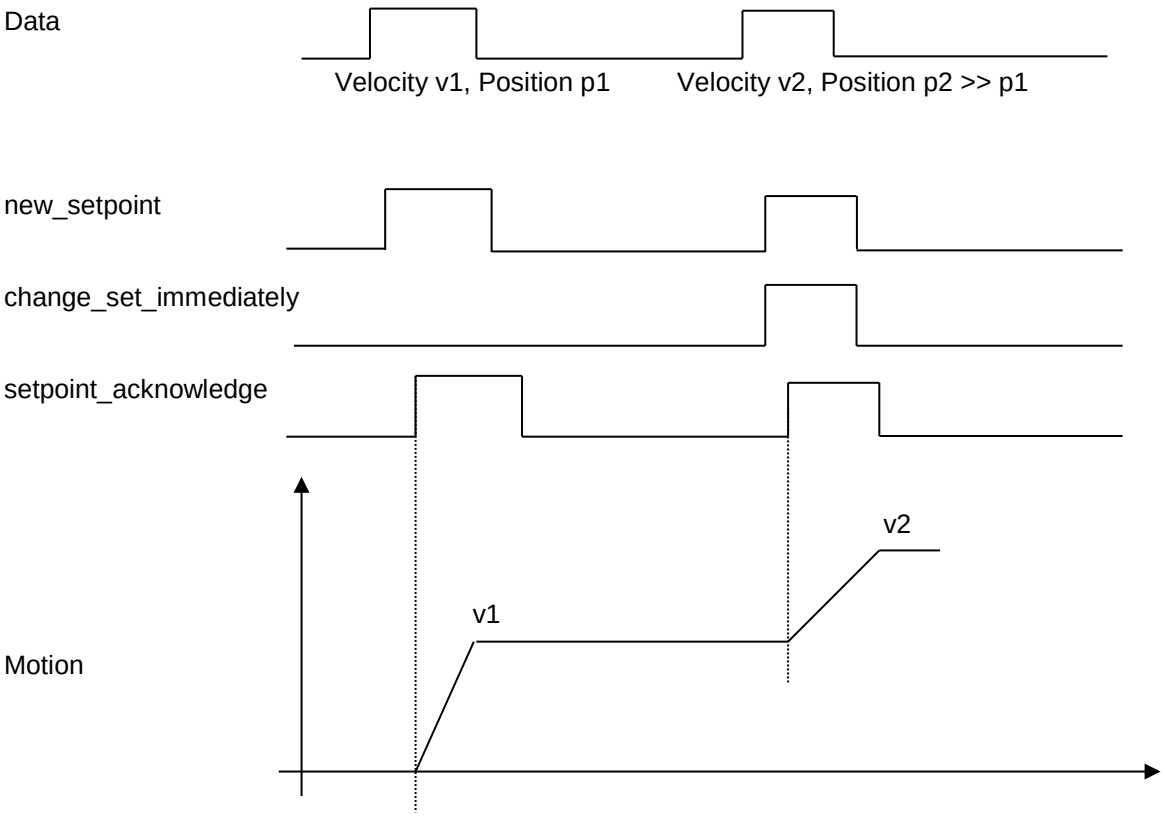
☞

| **Note** |
| --- |
| The transition from one drive command to the next takes place at the target position. If there is no new target position stored into the buffer of the drive it stops. The drive does not stop in between drive commands if there is a new target position for the successive drive command. |

☞

| **Note** |
| --- |
| The transition between drive commands is position dependent. |

Two drive commands with change set immediately




The main difference to the scheme above it the use of the bit „change_set_immediately“. In this case the controller does not finish the first positioning but proceeds immediately with the new drive command.

☞

| **Note** |
| --- |
| With the use of the bit „change_set_immediately“ the transition to the next drive command is not position dependent anymore, but time specific. |

☞

| **Note** |
| --- |
| The transition between drive commands is dependent on the time the command is sent |

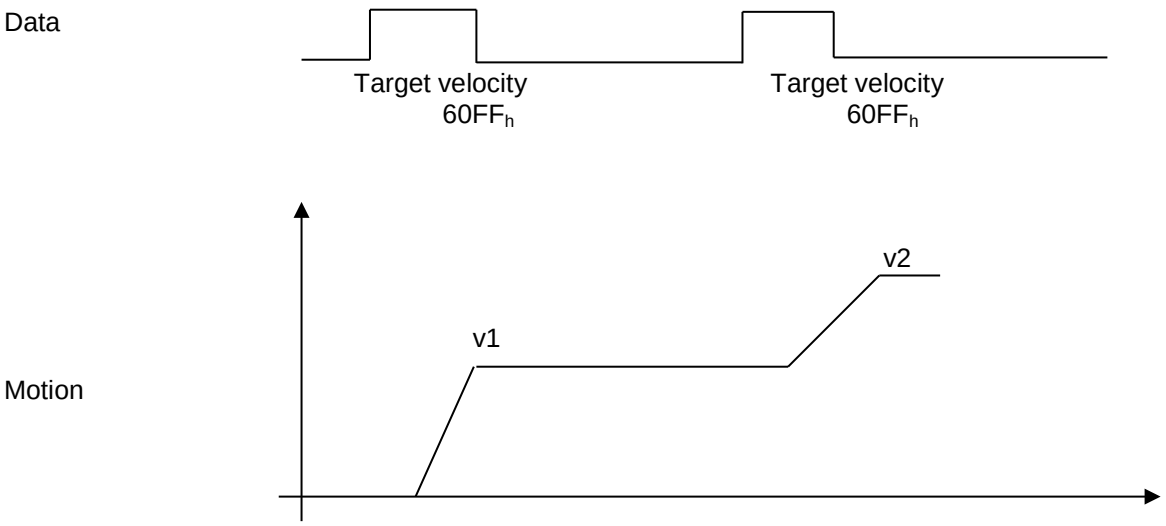
### 10.5.2 Acceleration and deceleration

The data for acceleration and deceleration is given in rpm/s.

## 10.6 Profile Velocity Mode

In Profile Velocity Mode the acceleration ramp is defined by parameter $20A0_h$ (ramp time to $V_{max}$). The reference $V_{max}$ is the Maximum Profile Velocity ($607F_h$).

The control of the velocity follows this scheme:




In the controlword for the profile velocity mode is a HALT bit. At the moment, the HALT bit is set to 0 and a PDO 4 defines a target velocity, the drive ramps to this velocity. If there is following a new PDO containing with a new velocity, this new velocity is the new target velocity. Direct control of the velocity through SDO is not possible. In this case the drive needs to be stopped with the HALT bit and restarted again.

## 10.7 Homing Mode

The method is controlled by object $6098_h$. The user configuration determines whether a homing run is required prior to any positioning. (Object $209C_h$)

### 10.7.1 Parameter

| | | |
|---|---|---|
| Acceleration ramp | [rpm/s] | (Object $609A_h$) |
| Deceleration ramp | [msec] | (Object $209A_h$) |
| Search velocity | [rpm] | (Object $6099_h$ Sub-Index 1) |
| Positioning velocity | [rpm] | (Object $6099_h$ Sub-Index 2) |
| Home offset | [incr.] | (Object $607C_h$) |
| Torque limit for "Homing to block" | [%] | (Object $209B_h$) |

| Homing method | Homing switch type | Search direction | Index pulse | Positioning direction |
|---|---|---|---|---|
| -3 | To block / negative search direction | | | |
| -2 | To block / positive search direction | | | |
| 1 | Brake contact (NLS) | - | yes | + |
| 2 | Brake contact (PLS) | + | yes | - |
| 3 | Make contact | + | yes | - |
| 4 | Brake contact | - | yes | + |
| 5 | Make contact | - | yes | + |
| 6 | Brake contact | + | yes | - |
| 7 | Make contact | + | yes | - |
| 10 | Make contact | + | yes | + |
| 11 | Make contact | - | yes | + |
| 14 | Make contact | - | yes | - |
| 17 | Brake contact | - | no | + |
| 18 | Brake contact | + | no | - |
| 19 | Make contact | + | no | - |
| 20 | Brake contact | - | no | + |
| 21 | Make contact | - | no | + |
| 22 | Brake contact | + | no | - |
| 23 | Make contact | + | no | - |
| 26 | Make contact | + | no | + |
| 27 | Make contact | - | no | + |
| 30 | Make contact | - | no | - |
| 33 | / | - | yes | / |
| 34 | / | + | yes | / |
| 35 | Zero setting | | | |

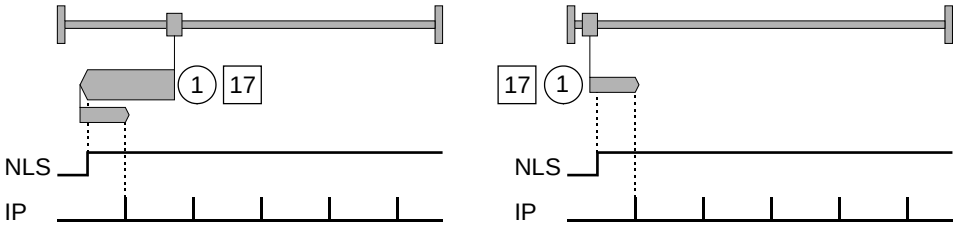
### 10.7.2 Abbreviations

NLS: Negative Limit Switch
PLS: Positive Limit Switch
HS: Homing Switch
IP: Index Pulse

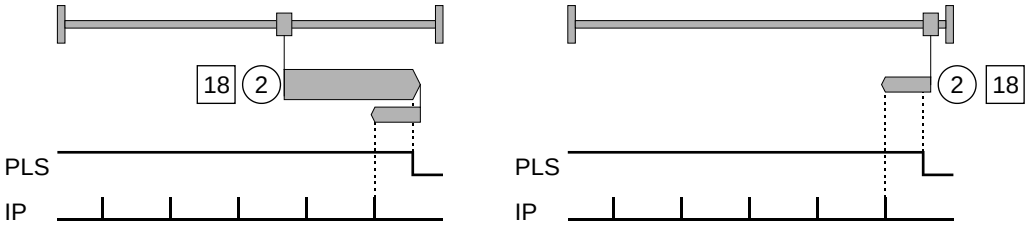
### 10.7.3 Sequence

Modes with a circle around the number home to the index pulse.
Modes with a square around the number home to the appropriate edge of the switch.

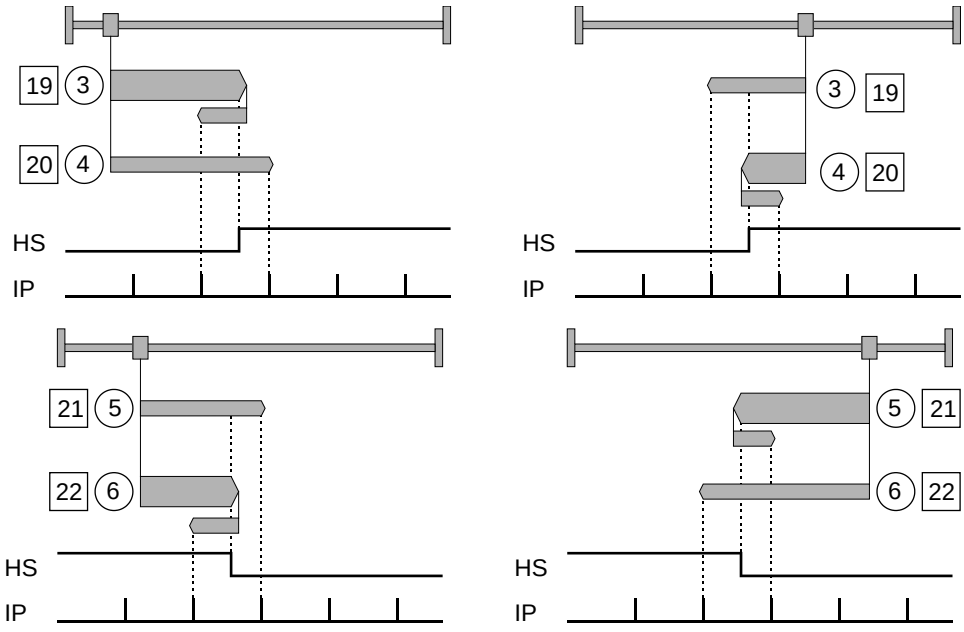
#### *10.7.3.1 Homing to negative limit switch (NLS)*



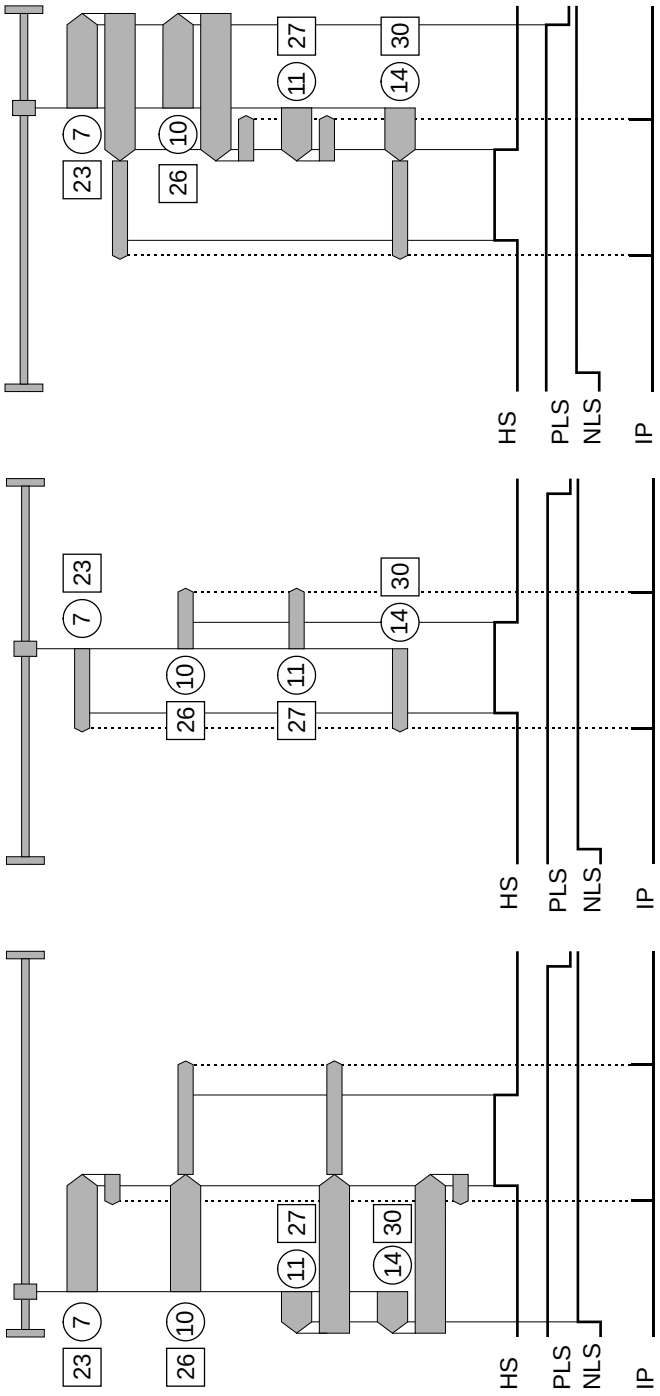

#### *10.7.3.2 Homing to positive limit switch (PLS)*



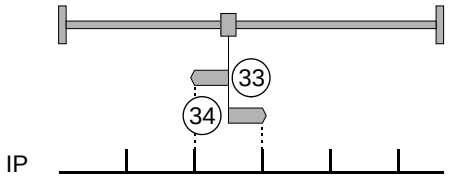

#### *10.7.3.3 Homing on the home switch*



### 10.7.4 Homing to the home switch:




### 10.7.5 Homing to the index pulse



### 10.7.6 Sequence „Homing to block“

Bit 4 in the controlword of homing mode starts the homing. The method is selected through object $6098_h$. Homing to block is method –3 negative direction or –2 positive direction.

### 10.7.7 Sequence „Zero setting"

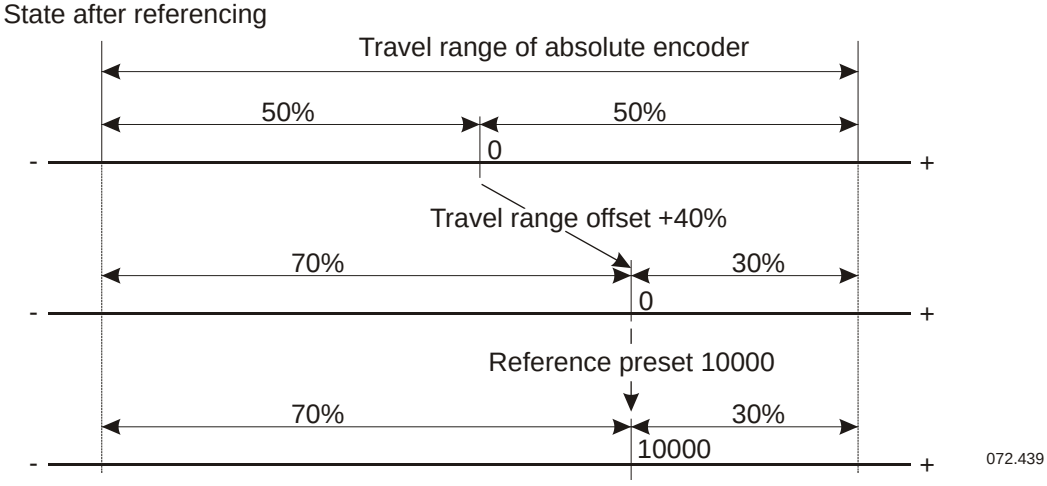
Bit 4 in the controlword of homing mode starts the homing. The method is selected through object $6098_h$. „Zero setting“ is method 35.

### 10.7.8 Homing with the absolute encoder

Bit 4 in the controlword of homing mode starts the homing. The method is selected through object $6098_h$.
With absolute encoder only methods –3, -2, 17 ... 30 and 35 are supported.

The absolute encoder provides the actual position value at any time. It is not necessary to reference after each initialising. However, it is necessary to adjust the absolute encoder once.

In the parameter area – not accessible to the user – the travel range of the absolute encoder is stored. This is the maximum travel range for the drive. On delivery the same amount of revolutions in positive and negative direction are allowed. The parameter „Absolute encoder travel range offset“ allows the adjustment of this to suit the application respectively the mounting situation.




The travel range offset is in respect to the available range. I.e. 40% of 50%.

## 10.8 Electronic axle

In the operation mode „Electronic Axle“ the controller counts all clock/direction or Channel A/Channel B inputs and calculates them to movements of the axle. A ratio can be can be taken into account and adjusted in objects $0x20C0_h$ to $0x20C3_h$.

☞

| **Note** |
| --- |
| To activate the electronic axle the mode of operation must be set to „-1“ (Object $6060_h$; manufacturer specific mode).<br>Inputs 3 and 4 need to be assigned to clock/direction respectively Channel A / Channel B. |

### 10.8.1 Adjustments electronic axle

To configure the electronic axle the following parameters can be adjusted:

- Type (Clock/direction or Channel A / Channel B)
- Edge detection (rising, falling or both edges)
- Sense of rotation (normal or inverted)
- Ratio in 1/1000

Example:

The resolution of the motor encoder is 4096 lpr. The user has a hand wheel with 100 increments per revolution. One revolution of the hand wheel shall represent 10 motor revolutions. Only rising edges count and the mode is clock/direction.

Ratio: $\frac{10 \cdot 4096}{100} = 409{,}6$ .

Generally:

$$Ratio = \frac{Enocder\_resolution \cdot Number\_of\_revolutions \cdot Pulse\_factor}{Increments\_of\_handwheel}$$

Pulse factor is defined through the edge detection and the type of electronic axle. It is 1, 2 or 4.

## 10.9 Interpolated Position Mode

The interpolated position mode can be used to control multiple axles or a single axle with time-interpolated set-points. The interpolated position mode uses the SYNC mechanism to synchronise.

For synchronous operation the cycle time is defined using the object interpolation time period and the SYNC definition.

Example:

The SYNC Time shall be defined to 100ms. Every 100 msec a SYNC Message is sent by the Master.

Parameter $60C2_h$, Sub-Index 1 = 100

Parameter $60C2_h$, Sub-Index 2 = -3 [1 msec]

Parameter $60C3_h$, Sub-Index 2 = 1

Another configuration could be:

Parameter $60C2_h$, Sub-Index 1 = 1

Parameter $60C2_h$, Sub-Index 2 = -1 [100 msec]

Parameter $60C3_h$, Sub-Index 2 = 1

### 10.9.1 Control- and Statusword

Controlword:

The Interpolated Position Mode is activated using Bit 4 of the controlword. Once this bit is set SYNC messages are used to take the position in $60C1_h$, Sub-Index 1 and interpolate it.

Statusword:

Once the Interpolated Position Mode is active, Bit 12 of the statusword is set.

Once the Position is reached the “target reached” (Bit 10 of the statusword) is set.

There are two different interpolation modes selectable. One is the linear interpolation using SYNC mechanism to synchronise. The other is linear interpolation using SYNC and SYNC-Time to synchronise. The second one is helpful for SPS which have difficulties with giving a clear SYNC every sync-period

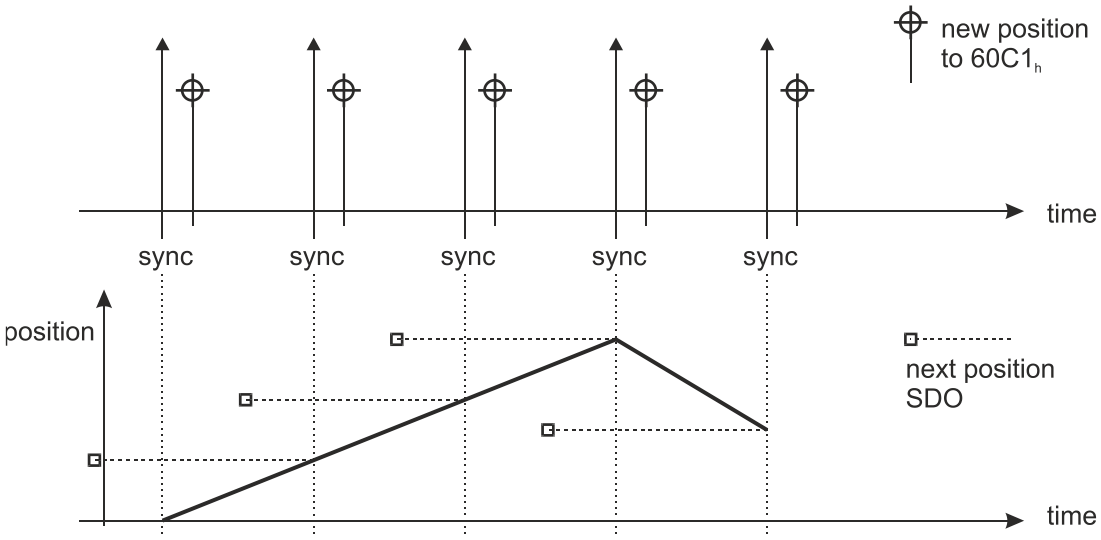
☞ **Note**

**If no SYNC message is received the axle will stop after the SYNC-Time immediately.**

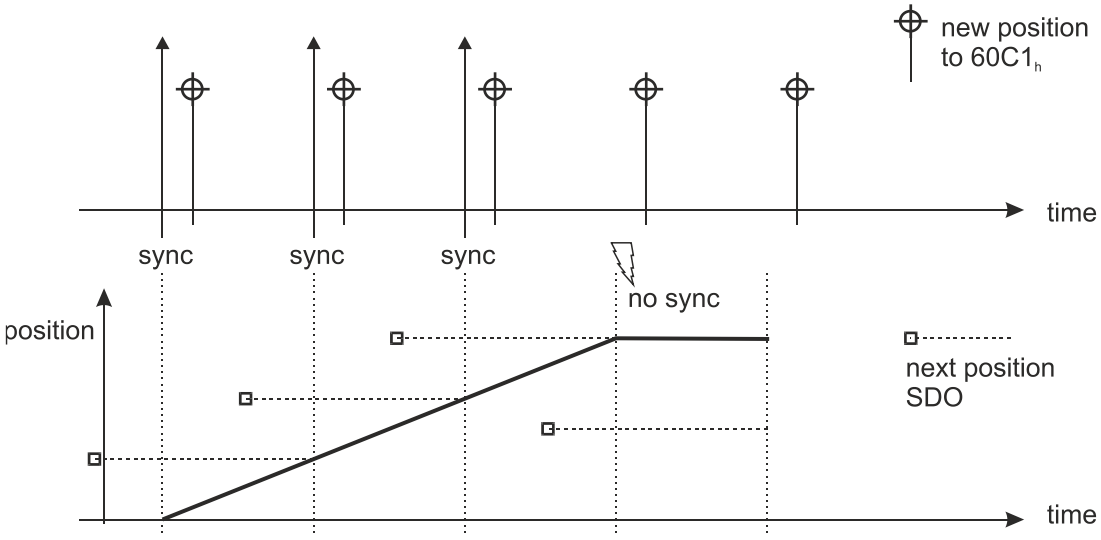
### 10.9.2 Parameters

- Interpolation sub mode select ($60C0_h$)
- Interpolation data record ($60C1_h$)
- Interpolation time period ($60C2_h$)
- Interpolation sync definition ($60C3_h$)
- Interpolation data configuration ($60C4_h$)

### 10.9.3 Linear Interpolation Sub Mode 1 ($60C0_h$)



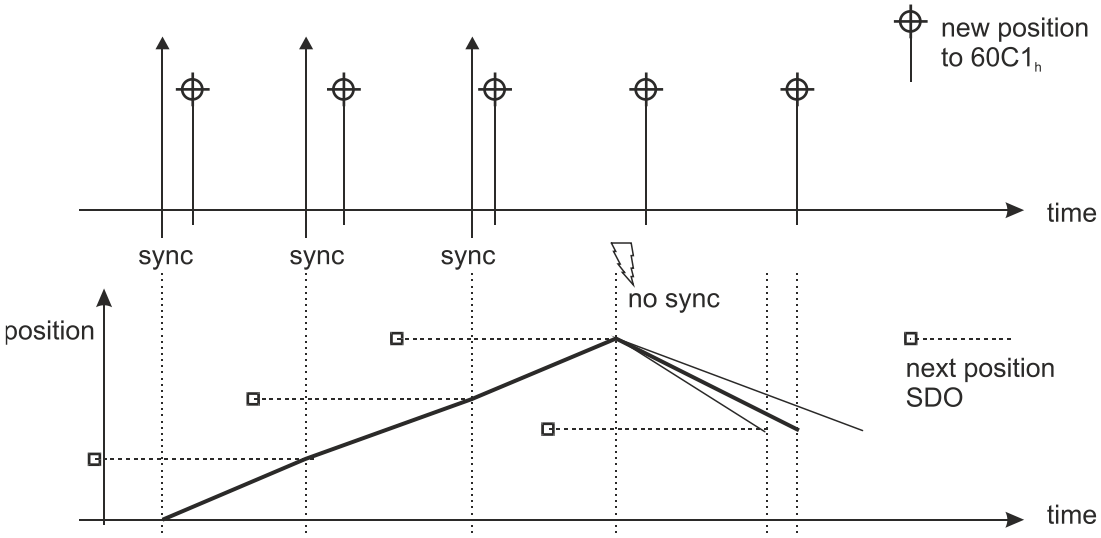

The motor position controller will automatically interpolate between the actual start and the new end position within his own regulation cycle.




If the SYNC object is not received before the controller is finished, the axle will stop after the defined SYNC cycle.

### 10.9.4 Linear Interpolation Sub Mode –1 ($60C0_h$)

Since some slower SPS System have problems with a constant SYNC Message period or do have a big Jitter on this signal it is recommended to use this mode instead.




The controller checks if the position value in Object $60C1_h$ has been updated. If now the Controller has finished the linear interpolation within the SYNC-period instead of stopping it will continue interpolation with this new value until the next SYNC is received. When the delayed SYNC arrives it will recalculate the interpolation.

# 11. Controller with Control- and Statusword

## 11.1 Object $6040_h$: Controlword

The controlword consists of bits for:

- the controlling of the state
- the controlling of operating modes

The bits of the controlword are defined as follows:

| 15..11 | 10 | 9 | 8 | 7 | 6..4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| manufacturer specific | reserved | | halt | fault reset | operation mode specific | enable operation | quick stop | enable voltage | switch on |

LSB

**Bits 0 – 3 and 7:**

Device control commands are triggered by the following bit patterns in the controlword:

| Command | Bit of the controlword | | | | | Transitions |
|---|---|---|---|---|---|---|
| | 7<br>Fault reset | 3<br>Enable operation | 2<br>Quick stop | 1<br>Enable voltage | 0<br>Switch on | |
| Shutdown | 0 | X | 1 | 1 | 0 | 2,6,8 |
| Switch on | 0 | 0 | 1 | 1 | 1 | 3* |
| Switch on | 0 | 1 | 1 | 1 | 1 | 3** |
| Disable voltage | 0 | X | X | 0 | X | 7,9,10,12 |
| Quick stop | 0 | X | 0 | 1 | X | 7,10,11 |
| Disable operation | 0 | 0 | 1 | 1 | 1 | 5 |
| Enable operation | 0 | 1 | 1 | 1 | 1 | 4,16 |
| Fault reset | ↑ | X | X | X | X | 15 |

* In the state „Switched on“ the drive executes the functionality of this state

** There is no functionality in the state „Switched on“. The drive does not do anything in this state.

**Bits 4 - 6 and 8:**

These bits are operation mode specific.

| Bit | Operation Mode | | | |
|---|---|---|---|---|
| | Interpolated Positioning Mode | Profile Position Mode | Profile Velocity Mode | Homing Mode |
| 4 | Enable IP Mode | New Set-point | Reserved | Homing operation start |
| 5 | - | Change set immediately | Reserved | Reserved |
| 6 | - | Absolute / Relative | Reserved | Reserved |
| 8 | Halt | Halt | Halt | Halt |

**Bits 9 –15:**

These bits are reserved for future use.

## 11.2 Object $6041_h$: Statusword

The statusword indicates the current state of the drive. No bits are latched. The statusword consists of bits for:

- the current state of the drive,
- the operating state of the mode.

| Bit | Description |
|---|---|
| 0 | Ready to switch on |
| 1 | Switched on |
| 2 | Operation enabled |
| 3 | Fault |
| 4 | Voltage enabled |
| 5 | Quick stop |
| 6 | Switch on disabled |
| 7 | Warning |
| 8 | Reserved |
| 9 | Remote |
| 10 | Target reached |
| 11 | Internal limit active |
| 12-13 | Operation mode specific |
| 14 | mode of operation finished |
| 15 | Reserved |

**Bits 0 – 3, 5 and 6:**

The following bits indicate the status of the device:

| Value (binary) | State |
|---|---|
| xxxx xxxx x0xx 0000 | Not ready to switch on |
| xxxx xxxx x1xx 0000 | Switch on disabled |
| xxxx xxxx x01x 0001 | Ready to switch on |
| xxxx xxxx x01x 0011 | Switched on |
| xxxx xxxx x01x 0111 | Operation enabled |
| xxxx xxxx x00x 0111 | Quick stop active |
| xxxx xxxx x0xx 1111 | Fault reaction active |
| xxxx xxxx x0xx 1000 | Fault |

**Bit 4: Voltage Enabled:**

High voltage is applied to the drive when this bit is set to 1.

**Bit 5: Quick Stop:**

When reset, this bit indicates that the drive is reacting on a quick stop request

**Bit 7: Warning:**

A drive warning is present if bit 7 is set. The cause means no error but a state that has to be mentioned, e.g. temperature limit, job refused. The status of the drive does not change. The cause of this warning may be found by reading the fault code parameter. The bit is set and reset by the device.

**Bit 9: Remote:**

If bit 9 is set, then parameters may be modified via the CAN-network, and the drive executes the content of a command message. If the bit remote is reset, then the drive is in local mode and will not execute the command message.

**Bit 10: Target reached:**

If bit 10 is set by the drive, then a set-point has been reached. The set-point is dependent on the operating mode.

**Bit 11: Internal Limit Active:**

This bit set by the drive indicates, that an internal limitation is active (e.g. position range limit).

**BIT 12 and BIT 13:**

These bits are operation mode specific.

| Bit | Operation Mode | | | |
|---|---|---|---|---|
| | Interpolated Positioning Mode | Profile Position Mode | Profile Velocity Mode | Homing Mode |
| 12 | Interpolated Position Mode Active | Set-point acknowledge | Speed | Homing attained |
| 13 | reserved | Following error | Max slippage error | Homing error |

**BIT 14:**

Mode of operation finished / ready for command.

# 12. Object Dictionary

## 12.1 Scope

The object dictionary is divided into the following sections:

- communications segment
- manufacturer specific segment
- profile segment

Used abbreviations:

Data type:

| | | |
|---|---|---|
| INTEGER 8 | 8bit-value | -128, ..., 127 |
| INTEGER 16 | 16bit-value | -32768, ..., 32767 |
| INTEGER 32 | 32bit-value | -2147483648, ..., 2147483647 |
| UNSIGNED 8 | 8bit-value | 0, ..., 255 |
| UNSIGNED 16 | 16bit-value | 0, ..., 65535 |
| UNSIGNED 32 | 32bit-value | 0, ..., 4294967295 |
| VISIBLE STRING | ASCII-Character | $20_h$, ..., $7E_h$ |

Access:

| | | |
|---|---|---|
| RO | Read Only | The object value is read only. |
| RW | Read Write | The object value may be read and written. |
| WO | Write Only | The object value may only be written. |
| C | Constant | The object value is constant and not changeable. |

Upper/Lower Limit:

During write access to the object dictionary the limits are checked. The limits may differ from the recommendations in DSP 402.

Units:

esiMot does not support the factor group. The following units apply:

| | |
|---|---|
| Position Units | incr. (Increments) – Consider resolution of encoder respectively Hall-sensors |
| Velocity Units | rpm |
| Acceleration Units | rpm /s |

### 12.1.1 Broadcast Objects of the Pre-defined Connection Set

| *Object* | Function code (binary) | Resulting COB-ID | Communication Parameters at Index |
|---|---|---|---|
| *NMT* | 0000 | 0 | - |
| *SYNC* | 0001 | 128 ($80_h$) | $1005_h$ |

### 12.1.2 Peer-to-Peer Objects of the Pre-defined connection Set

| *Object* | Function code (binary) | Resulting COB-ID | Communication Parameters at Index |
|---|---|---|---|
| *EMERGENCY* | $0001_b$ | 129 ($81_h$) – 255 ($FF_h$) | $1014_h$, $1015_h$ |
| *PDO1 (tx)* | $0011_b$ | 385 ($181_h$) – 511 ($1FF_h$) | $1800_h$ |
| *PDO1 (rx)* | $0100_b$ | 513 ($201_h$) – 639 ($27F_h$) | $1400_h$ |
| *PDO2 (tx)* | $0101_b$ | 641 ($281_h$) – 767 ($2FF_h$) | $1801_h$ |
| *PDO2 (rx)* | $0110_b$ | 769 ($301_h$) – 895 ($37F_h$) | $1401_h$ |
| *PDO3 (tx)* | $0111_b$ | 897 ($381_h$) – 1023 ($3FF_h$) | $1802_h$ |
| *PDO3 (rx)* | $1000_b$ | 1025 ($401_h$) – 1151 ($47F_h$) | $1402_h$ |
| *PDO4 (tx)* | $1001_b$ | 1153 ($481_h$) – 1279 ($4FF_h$) | $1803_h$ |
| *PDO4 (rx)* | $1010_b$ | 1281 ($501_h$) – 1407 ($57F_h$) | $1403_h$ |
| *SDO (tx)* | $1011_b$ | 1409 ($581_h$) – 1535 ($5FF_h$) | $1200_h$ |
| *SDO (rx)* | $1100_b$ | 1537 ($601_h$) – 1663 ($67F_h$) | $1200_h$ |
| *NMT Error Control* | $1110_b$ | 1793 ($701_h$) – 1919 ($77F_h$) | $1016_h$,$1017_h$ |

☞ **Note**

This table has to be seen from the devices point of view.

The pre-defined connection set always applies to the standard CAN frame with 11-bit identifier, even if extended CAN – frames are present in the network.

## 12.2 Communications Segment

For a detailed description of all communication parameters see CiA Standard 301[3] and DSP 402[4].

| Index | Sub-index | Name | Data type | Upper Limit | Lower Limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| $1000_h$ | / | Device Type | UNSIGNED 32 | / | / | / | RO | Indicates the used profile and type.<br><br>Bit 0 – 15 Profile number (DSP 402)<br>Bit 16 – 23 Drive type (Servo Drive Bit 17)<br>Bit 24 – 31 not used |
| $1001_h$ | / | Error Register | UNSIGNED 8 | / | / | / | RO | Represents the error state of the device. |

| Index | Sub-index | Name | Data type | Upper Limit | Lower Limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| $1002_h$ | / | Manufacturer Status Register | UNSIGNED 32 | / | / | / | RO | This object indicates the operation modes and error states of the device.<br>Bit 0: Initialisation OK<br>Bit 1: Homed<br>Bit 2: Regulator active<br>Bit 3: Position reached<br>Bit 4: Axle in motion<br>Bit 5: Ramp active<br>Bit 6: Brake active<br>Bit 7: Error<br>Bit 8: Direction (only in amplifier version)<br>Bit 9: Enabled<br>Bit 10: Stop<br>Bit 11: Hardware limit switch P<br>Bit 12: Hardware limit switch N<br>Bit 13: Software travel limit active<br>Bit 14: Software limit of travel P<br>Bit 15: Software limit of travel N<br>Bit 16: Logic supply voltage too low<br>Bit 17: Logic supply voltage too high<br>Bit 18: Reference voltage too low<br>Bit 19: -<br>Bit 20: Motor supply voltage too low<br>Bit 21: Motor supply voltage too high<br>Bit 22: -<br>Bit 23: Ballast voltage too high<br>Bit 24: Warning temperature output stage<br>Bit 25: Error temperature output stage<br>Bit 26: -<br>Bit 27: -<br>Bit 28: Warning temperature motor<br>Bit 29: Error temperature motor<br>Bit 30: -<br>Bit 31: - |

| Index | Sub-index | Name | Data type | Upper Limit | Lower Limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| $1003_h$ | $00_h$ | Pre-defined Error Field | UNSIGNED 32 | / | / | / | RW | The object at index 1003h holds the errors that have occurred on the device and have been signalled via the Emergency Object. In doing so it provides an error history. Up to 10 error messages are hold.<br>Writing a „0“ to sub-index 0 deletes the entire error history (empties the array). Other values are not permitted. |
| | $01_h$ | standard error field | - | - | - | - | RO | 0: no error<br>1: Wrong or missing data<br>2: Wrong or missing regulator data<br>3: Tracking error<br>4: Timeout drive<br>5: Axle in motion<br>6: Wrong ramp parameter<br>7: Travel for ramp to far<br>8: Limit switch +<br>9: Limit switch -<br>10: Limit of travel range + exceeded<br>11: Limit of travel range – exceeded<br>12: Motor temperature too high<br>13: Temperature of output stage too high<br>14: Over-voltage motor<br>15: Under-voltage motor<br>16: Over-voltage logic<br>17: Under-voltage logic<br>18: EEPROM Checksum error<br>19: Stop applied – Status information only -<br>20: Stop applied (error message)<br>21: Overload motor<br>22: Profibus login missing<br>23: Error at initialisation<br>24: Waiting for enable<br>25: Working mode wrong<br>26: Bus off-line<br>27: RS232/RS485 Trigger Timeout<br>28: Axle not referenced |

| Index | Sub-index | Name | Data type | Upper Limit | Lower Limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 29: Wrong value – Status information only -<br>30: Reference voltage is too low<br>31: Range of travel exceeded<br>32: Reference lost<br>33: Wrong set value mode<br>34: Regulator state wrong<br>35: CANopen error<br>36: Encoder error<br>37: Reference type not supported |
| | $02_h$ | standard error field | - | - | - | - | RO | (see Sub-Index $01_h$) |
| | $03_h$ | standard error field | - | - | - | - | RO | (see Sub-Index $01_h$) |
| | $04_h$ | standard error field | - | - | - | - | RO | (see Sub-Index $01_h$) |
| | $05_h$ | standard error field | - | - | - | - | RO | (see Sub-Index $01_h$) |
| | $06_h$ | standard error field | - | - | - | - | RO | (see Sub-Index $01_h$) |
| | $07_h$ | standard error field | - | - | - | - | RO | (see Sub-Index $01_h$) |
| | $08_h$ | standard error field | - | - | - | - | RO | (see Sub-Index $01_h$) |
| | $09_h$ | standard error field | - | - | - | - | RO | (see Sub-Index $01_h$) |
| | $0A_h$ | standard error field | - | - | - | - | RO | (see Sub-Index $01_h$) |
| $1005_h$ | $00_h$ | COB-ID Sync | UNSIGNED 32 | - | - | - | RW | Index $1005_h$ defines the COB-ID of the Synchronisation Object (SYNC). Further, it defines whether the device generates the SYNC. The SYNC-Message is transmitted as Broadcast (NODE-ID 0).<br>Bit 0-10: Identifier<br>Bit 11- 28: If Bit 29 = 1, Bits 11-28 of 29 Bit Identifier<br>Bit 29: 0 = 11 bit Identifier 1 = 29 bit Identifier<br>Bit 30: 0 = device is no SYNC Producer<br>1 = device is SYNC Producer<br>Bit 31: don't care |
| $1008_h$ | $00_h$ | Manufacturer Device Name | STRING | - | - | - | C | Contains the manufacturer device name. (esiMot) |
| $1009_h$ | $00_h$ | Manufacturer Hardware Version | STRING | - | - | - | C | Contains the manufacturer hardware version description |
| $100A_h$ | $00_h$ | Manufacturer Software Version | STRING | - | - | - | C | Contains the manufacturer software version description. |

| Index | Sub-index | Name | Data type | Upper Limit | Lower Limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| $100C_h$ | $00_h$ | Guard Time | UNSIGNED 16 | 65535 | 0 | msec | RW | The objects at index $100C_h$ and $100D_h$ include the guard time in milliseconds and the life time factor. The life time factor multiplied with the guard time gives the life time for the Life Guarding Protocol. It is0 if not used |
| $100D_h$ | $00_h$ | Life Time Factor | UNSIGNED 8 | 255 | 0 | - | RW | The life time factor multiplied with the guard time gives the life time for the node guarding protocol. It is 0 if not used. |
| $1010_h$ | $00_h$ | Store Parameter Field | UNSIGNED 8 | - | - | - | RO | This object supports the saving of parameters in non-volatile memory.<br><br>Sub index 1: all parameter<br>Sub index 2: Communications parameter<br>Sub index 3: Applications parameter<br><br>To do this write a “save” to the according sub-index. |
| | $01_h$ | all parameter | UNSIGNED 32 | - | - | - | RW | Safe all parameter |
| | $02_h$ | Communications parameter | UNSIGNED 32 | - | - | - | RW | Safe Communications parameter |
| | $03_h$ | Applications parameter | UNSIGNED 32 | - | - | - | RW | Safe Applications parameter |
| $1011_h$ | $00_h$ | Restore Parameter Field | UNSIGNED 8 | - | - | - | RO | With this object the default values of parameters according to the communication or device profile are restored. By read access the device provides information about its capabilities to restore these values.<br><br>Sub index 1: all parameter<br>Sub index 2: Communications parameter<br>Sub index 3: Applications parameter<br><br>To do this write a “load” to the according sub-index. |
| | $01_h$ | all parameter | UNSIGNED 32 | - | - | - | RW | Restore all parameter |

| Index | Sub-index | Name | Data type | Upper Limit | Lower Limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| | $02_h$ | Communications parameter | UNSIGNED 32 | - | - | - | RW | Restore Communications parameter |
| | $03_h$ | Applications parameter | UNSIGNED 32 | - | - | - | RW | Restore Applications parameter |
| $1014_h$ | $00_h$ | COB-ID EMCY | UNSIGNED 32 | - | - | - | RO | Index $1014_h$ defines the COB-ID of the Emergency Object (EMCY).<br>The COB-ID is $80_h$ + Node-ID. |
| $1015_h$ | $00_h$ | Inhibit Time Emergency | UNSIGNED 16 | 65535 | 0 | 100µs | RW | The inhibit time for the EMCY message can be adjusted via this entry. If this entry exists it must be writeable in the object dictionary. The time is multiplied by of 100µs<br>This parameter is used to avoid bus locking by higher priority messages.<br>A 0 deactivates this feature. |
| $1016_h$ | $00_h$ | Consumer Heartbeat Time | UNSIGNED 32 | - | - | - | RW | The consumer heartbeat time defines the expected heartbeat cycle time and thus has to be higher than the corresponding producer heartbeat time configured on the device producing this heartbeat. Monitoring starts after the reception of the first heartbeat. If the consumer heartbeat time is 0 the corresponding entry is not used. The time has to be a multiple of 1ms (lo WORD). The lo BYTE of the hi WORD defines the node ID to be monitored. |
| $1017_h$ | $00_h$ | Producer Heartbeat Time | UNSIGNED 16 | 65535 | 0 | msec | RW | The producer heartbeat time defines the cycle time of the heartbeat. The producer heartbeat time is 0 if it is not used. The time is a multiple of 1ms.<br>With active monitoring the device sends cyclic its actual state to the NMT-Master. In this case the motor regulator is a heartbeat producer and the NMT-Master is consumer. |

| Index | Sub-index | Name | Data type | Upper Limit | Lower Limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| $1018_h$ | 00h | Identity Object | | - | - | - | - | The object at index $1018_h$ contains general information about the device.<br>• Vendor-ID<br>• Product code<br>• Revision number<br>• Serial number<br>Not used at this time. |
| $1200_h$ | $00_h$ | Server SDO Parameter 1 | UNSIGNED 8 | - | - | - | RO | This object defines the COB-IDs for the default Server-SDO.<br>A SDO-Server may also be a slave containing an object dictionary. In this case the esiMot. An ID for exchange between client and server is defined. |
| | $01_h$ | COB-ID Client -> Server (rx) | UNSIGNED 32 | - | - | - | RO | $600_h$ + NODE-ID |
| | $02_h$ | COB-ID Server -> Client | UNSIGNED 32 | - | - | - | RO | $580_h$ + NODE-ID |
| $1400_h$ | $00_h$ | Receive PDO Communication Parameter 1 | UNSIGNED 8 | - | - | - | RO | Contains the communication parameters for the PDOs the device is able to receive. According to DSP 402 the first receive PDO is appropriate for the state machine of the drive<br>This controlword controls the state machine.<br>PDOs may only be processed in NMT-state operational.<br>A parameter change may be accomplished in state pre-operational. |
| | $01_h$ | COB-ID | UNSIGNED 32 | - | - | - | RW | COB-ID of PDO |
| | $02_h$ | Transmission Type | UNSIGNED 8 | - | - | - | RW | Transmission Type<br>1, ..., 240: Synchronous transmission, triggered by every n-th SYNC-Message.<br>254, 255: Asynchronous transmission. |

| Index | Sub-index | Name | Data type | Upper Limit | Lower Limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| $1401_h$ | $00_h$ | Receive PDO Communication Parameter 2 | UNSIGNED 8 | - | - | - | RO | According to DSP 402 the second receive PDO is appropriate for the operating mode of the drive<br>This controlword controls the state machine and modes of operation.<br>PDOs may only be processed in NMT-state operational.<br>A parameter change may be accomplished in state pre-operational. |
| | $01_h$ | COB-ID | UNSIGNED 32 | - | - | - | RW | COB-ID of PDO |
| | $02_h$ | Transmission Type | UNSIGNED 8 | - | - | - | RW | Transmission Type<br>1, ..., 240: Synchronous transmission, triggered by every n-th SYNC-Message.<br>254, 255: Asynchronous transmission. |
| $1402_h$ | $00_h$ | Receive PDO Communication Parameter 3 | UNSIGNED 8 | - | - | - | RO | According to DSP 402 the third receive PDO is appropriate for the position profile mode of the drive<br>This controlword controls the state machine and the target position (pp).<br>A positioning may be relative or absolute started or stopped.<br>PDOs may only be processed in NMT-state operational.<br>A parameter change may be accomplished in state pre-operational. |
| | $01_h$ | COB-ID | UNSIGNED 32 | - | - | - | RW | COB-ID of PDO |
| | $02_h$ | Transmission Type | UNSIGNED 8 | - | - | - | RW | Transmission Type<br>1, ..., 240: Synchronous transmission, triggered by every n-th SYNC-Message.<br>254, 255: Asynchronous transmission. |

| Index | Sub-index | Name | Data type | Upper Limit | Lower Limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| $1403_h$ | $00_h$ | Receive PDO Communication Parameter 4 | UNSIGNED 8 | - | - | - | RO | According to DSP 402 the forth receive PDO is appropriate for the profile velocity mode of the drive This controlword controls the state machine and the target velocity (pv).<br>A positioning may be relative or absolute started or stopped.<br>PDOs may only be processed in NMT-state operational.<br>A parameter change may be accomplished in state pre-operational. |
| | $01_h$ | COB-ID | UNSIGNED 32 | - | - | - | RW | COB-ID of PDO |
| | $02_h$ | Transmission Type | UNSIGNED 8 | - | - | - | RW | Transmission Type<br>1, ..., 240: Synchronous transmission, triggered by every n-th SYNC-Message.<br>254, 255: Asynchronous transmission. |
| $1600_h$ | $00_h$ | Receive PDO Mapping Parameter 1 | UNSIGNED 8 | - | - | - | RW | Number of mapped objects. |
| | $01_h$ | $1^{st}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit)<br>Object $6040_h$ (Controlword DSP 402) |
| | $02_h$ | $2^{nd}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $03_h$ | $3^{rd}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $04_h$ | $4^{th}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $05_h$ | $5^{th}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |

| Index | Sub-index | Name | Data type | Upper Limit | Lower Limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| | $06_h$ | $6^{th}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $07_h$ | $7^{th}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $08_h$ | $8^{th}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| $1601_h$ | $00_h$ | Receive PDO Mapping Parameter 2 | UNSIGNED 8 | - | - | - | RW | Number of mapped objects. |
| | $01_h$ | $1^{st}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit)<br>Object $6040_h$ (Controlword DSP 402) |
| | $02_h$ | $2^{nd}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit)<br>Object $6060_h$ (Modes of Operation DSP 402) |
| | $03_h$ | $3^{rd}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $04_h$ | $4^{th}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $05_h$ | $5^{th}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $06_h$ | $6^{th}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $07_h$ | $7^{th}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |

| Index | Sub-index | Name | Data type | Upper Limit | Lower Limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| | $08_h$ | $8^{th}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| $1602_h$ | $00_h$ | Receive PDO Mapping Parameter 3 | UNSIGNED 8 | - | - | - | RW | Number of mapped objects. |
| | $01_h$ | $1^{st}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit)<br>Object $6040_h$ (Controlword DSP 402) |
| | $02_h$ | $2^{nd}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit)<br>Object $607A_h$ (Target Position DSP 402) |
| | $03_h$ | $3^{rd}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $04_h$ | $4^{th}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $05_h$ | $5^{th}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $06_h$ | $6^{th}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $07_h$ | $7^{th}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $08_h$ | $8^{th}$ mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| $1603_h$ | $00_h$ | Receive PDO Mapping Parameter 4 | UNSIGNED 8 | - | - | - | RW | Number of mapped objects. |

| Index | Sub-index | Name | Data type | Upper Limit | Lower Limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| | $01_h$ | 1st mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit)<br>Object $6040_h$ (Controlword DSP 402) |
| | $02_h$ | 2nd mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit)<br>Object $60FF_h$ (Target Velocity DSP 402) |
| | $03_h$ | 3rd mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $04_h$ | 4th mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $05_h$ | 5th mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $06_h$ | 6th mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $07_h$ | 7th mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $08_h$ | 8th mapped object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| $1800_h$ | $00_h$ | 1st Transmit PDO Parameter | UNSIGNED 8 | - | - | - | RO | The communications parameter of PDOs.<br>Number of Subindices 5. |
| | $01_h$ | COB-ID | UNSIGNED 32 | - | - | - | RW | COB-ID of PDO |
| | $02_h$ | Transmission Type | UNSIGNED 8 | - | - | - | RW | Transmission type:<br>1, ..., 240: Synchronous transmission, triggered by n-th SYNC-Message.<br>254, 255: Asynchronous Event Driven |
| | $03_h$ | Inhibit Time | UNSIGNED 16 | - | - | 100 msec | RW | This parameter defines the minimum time between two successive Transmit-PDOs. |

| Index | Sub-index | Name | Data type | Upper Limit | Lower Limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| | $04_h$ | Compatibility Entry | UNSIGNED 16 | - | - | - | RW | - |
| | $05_h$ | Event Timer | UNSIGNED 16 | - | - | msec | RW | The parameter “event timer “ defines the cycle time of the PDO transmission. |
| $1801_h$ | $00_h$ | $2^{nd}$ Transmit PDO Parameter | UNSIGNED 8 | - | - | - | RO | Defines the communication parameters of the PDO. Number of Subindices 5. |
| | $01_h$ | COB-ID | UNSIGNED 32 | - | - | - | RW | COB-ID of PDO |
| | $02_h$ | Transmission Type | UNSIGNED 8 | - | - | - | RW | Transmission type:<br>1, ..., 240: Synchronous transmission, triggered by n-th SYNC-Message.<br>254, 255: Asynchronous Event Driven |
| | $03_h$ | Inhibit Time | UNSIGNED 16 | - | - | 100 msec | RW | This parameter defines the minimum time between two successive Transmit-PDOs. |
| | $04_h$ | Compatibility Entry | UNSIGNED 16 | - | - | - | RW | - |
| | $05_h$ | Event Timer | UNSIGNED 16 | - | - | msec | RW | The parameter “event timer “ defines the cycle time of the PDO transmission. |
| $1802_h$ | $00_h$ | $3^{rd}$ Transmit PDO Parameter | UNSIGNED 8 | - | - | - | RO | Defines the communication parameters of the PDO. Number of Subindices 5. |
| | $01_h$ | COB-ID | UNSIGNED 32 | - | - | - | RW | COB-ID of PDO |
| | $02_h$ | Transmission Type | UNSIGNED 8 | - | - | - | RW | Transmission type:<br>1, ..., 240: Synchronous transmission, triggered by n-th SYNC-Message.<br>254, 255: Asynchronous Event Driven |
| | $03_h$ | Inhibit Time | UNSIGNED 16 | - | - | 100 msec | RW | This parameter defines the minimum time between two successive Transmit-PDOs. |
| | $04_h$ | Compatibility Entry | UNSIGNED 16 | - | - | - | RW | - |
| | $05_h$ | Event Timer | UNSIGNED 16 | - | - | msec | RW | The parameter “event timer “ defines the cycle time of the PDO transmission. |
| $1803_h$ | $00_h$ | $4^{th}$ Transmit PDO Parameter | UNSIGNED 8 | - | - | - | RO | The communications parameter of PDOs. Number of Subindices 5. |
| | $01_h$ | COB-ID | UNSIGNED 32 | - | - | - | RW | COB-ID of PDO |

| Index | Sub-index | Name | Data type | Upper Limit | Lower Limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| | $02_h$ | Transmission Type | UNSIGNED 8 | - | - | - | RW | Transmission type:<br>1, ..., 240: Synchronous transmission, triggered by n-th SYNC-Message.<br>254, 255: Asynchronous Event Driven |
| | $03_h$ | Inhibit Time | UNSIGNED 16 | - | - | 100 msec | RW | This parameter defines the minimum time between two successive Transmit-PDOs. |
| | $04_h$ | Compatibility Entry | UNSIGNED 16 | - | - | - | RW | - |
| | $05_h$ | Event Timer | UNSIGNED 16 | - | - | msec | RW | The parameter “event timer “ defines the cycle time of the PDO transmission. |
| $1A00_h$ | $00_h$ | $1^{st}$ Transmit PDO Mapping | UNSIGNED 8 | - | - | - | RW | Contains the mapping for the PDOs the device is able to transmit. |
| | $01_h$ | $1^{st}$ Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit)<br>Object $6041_h$ (Statusword DSP 402) |
| | $02_h$ | $2^{nd}$ Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $03_h$ | $3^{rd}$ Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $04_h$ | $4^{th}$ Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $05_h$ | $5^{th}$ Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $06_h$ | $6^{th}$ Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $07_h$ | $7^{th}$ Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |

| Index | Sub-index | Name | Data type | Upper Limit | Lower Limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| | $08_h$ | $8^{th}$ Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| $1A01_h$ | $00_h$ | $2^{nd}$ Transmit PDO Mapping | UNSIGNED 8 | - | - | - | RW | Here are the objects defined which are mapped into the RPDO. |
| | $01_h$ | $1^{st}$ Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit)<br>Object $6041_h$ (Statusword DSP 402) |
| | $02_h$ | $2^{nd}$ Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit)<br>Object $6061_h$ (Modes of Operation Display DSP 402) |
| | $03_h$ | $3^{rd}$ Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $04_h$ | $4^{th}$ Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $05_h$ | $5^{th}$ Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $06_h$ | $6^{th}$ Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $07_h$ | $7^{th}$ Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $08_h$ | $8^{th}$ Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| $1A02_h$ | $00_h$ | $3^{rd}$ Transmit PDO Mapping | UNSIGNED 8 | - | - | - | RW | Here are the objects defined which are mapped into the RPDO. |

| Index | Sub-index | Name | Data type | Upper Limit | Lower Limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| | $01_h$ | 1st Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit)<br>Object $6041_h$ (Statusword DSP 402) |
| | $02_h$ | 2nd Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit)<br>Object $6064_h$ (Position Actual Value DSP 402) |
| | $03_h$ | 3rd Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $04_h$ | 4th Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $05_h$ | 5th Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $06_h$ | 6th Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $07_h$ | 7th Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | $08_h$ | 8th Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| $1A03_h$ | $00_h$ | 4th Transmit PDO Mapping | UNSIGNED 8 | - | - | - | RW | Here are the objects defined which are mapped into the RPDO. |
| | $01_h$ | 1st Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit)<br>Object $6041_h$ (Statusword DSP 402) |
| | $02_h$ | 2nd Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit)<br>Object $606C_h$ (Velocity Actual Value DSP 402) |

| Index | Sub-index | Name | Data type | Upper Limit | Lower Limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| | 03h | 3rd Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | 04h | 4th Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | 05h | 5th Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | 06h | 6th Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | 07h | 7th Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |
| | 08h | 8th Mapped Object | UNSIGNED 32 | - | - | - | RW | Bit 0 – 7: Object length (8bit)<br>Bit 8 – 16: Sub-Index (8 bit)<br>Bit 16 – 31: Index (16 bit) |

## 12.3 Manufacturer specific segment

| Index | Sub-index | Name | Data type | Upper limit | Lower limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| $2001_h$ | $00_h$ | Elapsed hour counter | UNSIGNED 32 | / | / | min | RO | Shows the time in minutes |
| $2002_h$ | $00_h$ | Stored position | INTEGER 32 | **2147483647** | **-2147483648** | Incr. | RO | If the mode „store positioning against power outage“ is selected, the positioning is stored here. |
| $2003_h$ | $00_h$ | Axle number Servo Link Protocol | UNSIGNED 8 | 255 | 0 | - | RO | Defines the axle number for the serial protocol. This parameter needs to be 1 if no RS485 bus is used. |
| $2004_h$ | $00_h$ | Bus address | UNSIGNED 8 | 127 | 1 | - | RW | Defines the CANopen Bus address of the esiMot.<br><br>This feature is available form version 010.108.02up only. |

| Index | Sub-index | Name | Data type | Upper limit | Lower limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| $2005_h$ | $00_h$ | Operating Mode at Start-up | UNSIGNED 8 | 4 | 1 | - | RW | This defines the operating mode when the controller changes to “operation enabled” after power up.<br><br>Value 1: Speed control<br>Value 4: Positioning |
| $2006_h$ | $00_h$ | Speed measurement system | UNSIGNED 8 | 3 | 0 | - | RO | The speed measurement system<br><br>Value 0: Hall sensors<br>Value 1: Incremental encoder<br>Value 2: Absolute encoder<br>Value 3: external encoder |
| $2007_h$ | $00_h$ | Position measurement system | UNSIGNED 8 | 2 | 0 | - | RO | The position measurement system<br><br>Value 0: Hall sensors<br>Value 1: Incremental encoder<br>Value 2: Absolute encoder<br>Value 3: external encoder |
| $2009_h$ | $00_h$ | store positioning against power outage | UNSIGNED 8 | 1 | 0 | - | RW | Defines the position storage mode<br><br>0: Do not store position<br>1: Store position |
| $200A_h$ | $00_h$ | Position regulator axis resolution | UNSIGNED 32 | - | - | - | RO | Internal value |
| $200C_h$ | $00_h$ | Response to errors | UNSIGNED 8 | 1 | 0 | - | RW | Defines the stop mode on errors.<br><br>0: Immediate stop<br>1: Ramp stop |
| $200D_h$ | $00_h$ | baudrate | UNSIGNED 8 | 5 | 0 | - | RW | Defines the baud rate:<br>0 : 50 kBit/s<br>1: 100 kBit/s<br>2: 125 kBit/s<br>3: 250 kBit/s<br>4: 500 kBit/s<br>5: 1000 kBit/s |

| Index | Sub-index | Name | Data type | Upper limit | Lower limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| $2010_h$ | $00_h$ | Input function Input 1 | UNSIGNED 8 | 15 | 0 | - | RW | Defines the function of input 1.<br><br>0: Not used<br>8: Reference switch<br>9: Limit switch +<br>10: Limit switch -<br>13: Jog +<br>14: Jog -<br>15: Rapid traverse (jog mode)<br><br>Other values are not permitted. A „reset node“ is needed after each change. |
| $2011_h$ | $00_h$ | Input function Input 2 | UNSIGNED 8 | 15 | 0 | - | RW | Defines the function of input 2.<br>(see object $2010_h$) |
| $2012_h$ | $00_h$ | Input function Input 3 | UNSIGNED 8 | 25 | 0 | - | RW | Defines the function of input 3.<br><br>0: Not used<br>8: Reference switch<br>9: Limit switch +<br>10: Limit switch -<br>13: Jog +<br>14: Jog -<br>15: Rapid traverse (jog mode)<br>25: Clock / Channel A<br><br>Other values are not permitted. A „reset node“ is needed after each change. |

| Index | Sub-index | Name | Data type | Upper limit | Lower limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| $2013_h$ | $00_h$ | Input function Input 4 | UNSIGNED 8 | 26 | 0 | - | RW | Defines the function of input 4.<br><br>0: Not used<br>8: Reference switch<br>9: Limit switch +<br>10: Limit switch -<br>13: Jog +<br>14: Jog -<br>15: Rapid traverse (jog mode)<br>26: Direction / Channel B<br><br>Other values are not permitted. A „reset node“ is needed after each change. |
| $2016_h$ | $00_h$ | Input function Input 7 | UNSIGNED 8 | 15 | 0 | - | RW | Defines the function of input 7.<br>(see object $2010_h$) |
| $2017_h$ | $00_h$ | Input function Input 8 | UNSIGNED 8 | 15 | 0 | - | RW | Defines the function of input 8.<br>(see object $2010_h$) |
| $2018_h$ | $00_h$ | Output function Output 1 | UNSIGNED 8 | 7 | 0 | - | RW | Defines the function of output 1<br><br>0: Not used<br>1: Ready<br>2: Regulator active<br>3: Error<br>4: Warning<br>5: Position reached<br>6: Axle in motion<br>7: Ballast active<br><br>Other values are not permitted. A „reset node“ is needed after each change. |
| $2019_h$ | $00_h$ | Output function Output 2 | UNSIGNED 8 | 7 | 0 | - | RW | Defines the function of output 2<br>(see object $2018_h$) |
| $201A_h$ | $00_h$ | Output function Output 3 | UNSIGNED 8 | 7 | 0 | - | RW | Defines the function of output 3<br>(see object $2018_h$) |
| $201B_h$ | $00_h$ | Output function Output 4 | UNSIGNED 8 | 7 | 0 | - | RW | Defines the function of output 4<br>(see object $2018_h$) |

| Index | Sub-index | Name | Data type | Upper limit | Lower limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| $201C_h$ | $00_h$ | Output status | UNSIGNED 8 | - | - | - | RO | Actual status of outputs |
| $201D_h$ | $00_h$ | Input status | UNSIGNED 8 | - | - | - | RO | Actual status of inputs |
| $2020_h$ | $00_h$ | Response time Brake opening | UNSIGNED 8 | 65535 | 0 | msec | RO | Defines the response time of the brake opening. |
| $2021_h$ | $00_h$ | Response time Brake closing | UNSIGNED 16 | 65535 | 0 | msec | RO | Defines the response time of the brake closing. |
| $2022_h$ | $00_h$ | Brake management | UNSIGNED 8 | 1 | 0 | - | R0 | Controls the brake management<br>0: off<br>1: on |
| $209A_h$ | $00_h$ | Homing Deceleration | UNSIGNED 16 | 10000 | 0 | msec | RW | Defines the deceleration ramp on homing. |
| $209B_h$ | $00_h$ | Homing Torque Limit | UNSIGNED 8 | 100 | 0 | % | RW | Torque limit for „Homing to block“. |
| $209C_h$ | $00_h$ | Reference run required | BOOLEAN | 1 | 0 | - | RW | Indicates the necessity of homing prior to a positioning. |
| $20A0_h$ | $00_h$ | Ramp to Vmax | UNSIGNED 16 | 9999 | 0 | msec | RW | Defines the ramp time to Vmax. (=$607F_h$ max. Profile velocity) |
| $20B0_h$ | $00_h$ | Jog acceleration | UNSIGNED 16 | 9999 | 0 | msec | RW | Defines the acceleration time in jog mode |
| $20B1_h$ | $00_h$ | Jog deceleration | UNSIGNED 16 | 10000 | 1 | msec | RW | Defines the deceleration time in jog mode |
| $20B2_h$ | $00_h$ | Jog tip increments | UNSIGNED 32 | 65535 | 0 | incr. | RW | Defines the number of increments on each tip in jog mode |
| $20B3_h$ | $00_h$ | Jog creep speed | UNSIGNED 16 | 10000 | 1 | Rpm | RW | Defines the creep speed in jog mode |
| $20B4_h$ | $00_h$ | Jog rapid traverse | UNSIGNED 16 | 10000 | 1 | Rpm | RW | Defines the rapid traverse speed in jog mode |
| $20C0_h$ | $00_h$ | Electronic axle type | UNSIGNED 8 | 1 | 0 | - | RW | Electronic axle type<br>0: Clock + direction<br>1: Channel A + Channel B |
| $20C1_h$ | $00_h$ | Electronic axle direction | UNSIGNED 8 | 1 | 0 | - | RW | Sense of rotation<br>0: normal<br>1: inverted |

| Index | Sub-index | Name | Data type | Upper limit | Lower limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| $20C2_h$ | $00_h$ | Electronic axle edge detection | UNSIGNED 32 | 2 | 0 | - | RW | Edge detection<br>0: Positive<br>1: Negative<br>2: Both edges |
| $20C3_h$ | $00_h$ | Electronic axle gear ratio | UNSIGNED 32 | 9999999 | 1 | /1000 incr. | RW | Defines the ratio between input increments and axle move increments. (1000 = 1:1) |

## 12.4 Profile segment

| Index | Sub-index | Name | Data type | Upper limit | Lower limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| $6007_h$ | $00_h$ | Abort Connection Option Code | INTEGER 16 | 2 | 3 | - | RO | The content of this object selects the function to be performed when the connection to the network is lost or a “Preop” or “Stop” command is received.<br><br>2: Device control command ‘Disable Voltage’<br>3: Device control command ‘Quick Stop’ |
| $6040_h$ | $00_h$ | Controlword | UNSIGNED 16 | - | - | - | RO | The device controlword controls all functions of the drive. For a detailed description see chapter 11.1. |
| $6041_h$ | $00_h$ | Statusword | UNSIGNED 16 | - | - | - | RO | The state of the drive is shown in the statusword For details see chapter 11.2. |
| $605A_h$ | $00_h$ | Quick Stop Option Code | INTEGER 16 | 2 | 2 | - | RW | The Quick Stop option code may only be set to 2.<br><br>2: slow down on quick stop ramp |
| $605B_h$ | $00_h$ | Shutdown Option Code | INTEGER 16 | 0 | 0 | - | RW | The Shutdown option code describes the response to a shutdown command. It may only be set to 0.<br><br>0: Disable drive function |
| $605C_h$ | $00_h$ | Disable Operation Option Code | INTEGER 16 | 1 | 1 | - | RW | The Disable Operation Option Code may only be set to 1.<br><br>1: Quick stop |

| Index | Sub-index | Name | Data type | Upper limit | Lower limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| $605D_h$ | $00_h$ | Halt Option Code | INTEGER 16 | 1 | 1 | - | RW | The Halt option code may only be set to 1.<br><br>1: Quick stop |
| $605E_h$ | $00_h$ | Fault Reaction Option Code | INTEGER 16 | 1 | 2 | - | RW | Defines the slow down mode on fault conditions.<br><br>1: slow down on slow down ramp<br>2: slow down on quick stop ramp<br><br>After slow down, the motor is free to rotate.<br>**Note:** On some critical faults, like undervoltage of the output stage, the drive is disabled immediately, without a slowdown ramp and free to rotate. |
| $6060_h$ | $00_h$ | Modes of Operation | INTEGER 8 | - | - | - | WO | This object defines the adjusted mode of operation<br><br>-1: Electronic Axle<br>0: Selection through object 2005h<br>1: Profile Position Mode<br>3: Profile Velocity Mode<br>6: Homing Mode |
| $6061_h$ | $00_h$ | Modes of Operation Display | INTEGER 8 | - | - | - | RO | This object indicates the adjusted mode of operation.<br><br>-1: Electronic Axle<br>0: Selection through object 2005h<br>1: Profile Position Mode<br>3: Profile Velocity Mode<br>6: Homing Mode |
| $6064_h$ | $00_h$ | Position Actual Value | INTEGER 32 | **2147483647** | **-2147483648** | incr. | RO | Displays the actual position. |

| Index | Sub-index | Name | Data type | Upper limit | Lower limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| $6065_h$ | $00_h$ | Following Error Window | UNSIGNED 32 | 65535 | 1 | incr. | RW | The following error window defines a range of tolerated position values symmetrically to the position demand value.<br><br>The following error monitoring is only active in Profile Position and Homing mode. |
| $6067_h$ | $00_h$ | Position Window | UNSIGNED 32 | 65535 | 0 | incr. | RW | The position window defines a symmetrical range of accepted positions relatively to the target position. If the actual value of the position encoder is within the position window, this target position is regarded as reached.<br><br>The position window is monitored only in Profile Position Mode. |
| $606B_h$ | $00_h$ | Velocity Demand Value | INTEGER 32 | 10000 | 0 | rpm | RO | The output value of the trajectory generator may be corrected by the output value of the position control function. It is then provided as a demand value for the velocity controller and given in the velocity units. |
| $606C_h$ | $00_h$ | Velocity Actual Value | INTEGER 32 | 10000 | 0 | rpm | RO | The velocity actual value is also represented in velocity units and is coupled to the velocity used as input to the velocity controller. |
| $606D_h$ | $00_h$ | Velocity Window | UNSIGNED 16 | 1000 | 0 | rpm | RW | The velocity window monitors whether the required process velocity has been achieved after an eventual acceleration or deceleration (braking) phase. It is given in velocity units |
| $6078_h$ | $00_h$ | Current Actual Value | INTEGER 16 | 1000 | -1000 | %o | RO | The current actual value refers to the instantaneous current in the drive motor. The value is given per thousand of rated current. |
| $607A_h$ | $00_h$ | Target Position | INTEGER 32 | 2147483647 | -2147483648 | incr. | RW | The target position is the position that the drive should move to in position profile mode using the current settings of motion control parameters such as velocity, acceleration, deceleration, etc. The target position is given in increments. |

| Index | Sub-index | Name | Data type | Upper limit | Lower limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| 607Ch | $00_h$ | Home Offset | UNSIGNED 32 | 2147483647 | -2147483648 | incr. | RW | The home offset object is the difference between the zero position for the application and the machine home position (found during homing), it is measured in increments. |
| 607Dh | $00_h$ | Software Position Limit | INTEGER 32 | - | - | - | RO | Software position limit contains the sub-parameters min position limit and max position limit. These parameters define the absolute position limits for the position demand value and the position actual value. Every new target position is checked against these limits. |
| | $01_h$ | Software Limit N | INTEGER 32 | 2147483647 | -2147483648 | incr. | RW | Min position limit |
| | $02_h$ | Software Limit P | INTEGER 32 | 2147483647 | -2147483648 | incr. | RW | Max position limit |
| 607Eh | $00_h$ | Polarity | UNSIGNED 8 | - | - | - | RW | Position demand value and position actual value are multiplied by 1 or -1 depending on the value of the polarity flag.<br><br>A „reset node“ is needed after each change. |
| 607Fh | $00_h$ | Max Profile Velocity | UNSIGNED 32 | 10000 | 1 | rpm | RW | The max profile velocity is the maximum allowed speed in either direction during a profiled move. It is given in rpm. The upper limit is checked against object 6080h. |
| 6080h | $00_h$ | Max Motor Speed | UNSIGNED 32 | 10000 | 1 | rpm | RO | The max motor speed is the maximum allowable speed for the motor in either direction and is given in rpm. |
| 6081h | $00_h$ | Profile Velocity | INTEGER 32 | 10000 | 1 | rpm | RW | The profile velocity is the velocity normally attained at the end of the acceleration ramp during a profiled move and is valid for both directions of motion. |
| 6083h | $00_h$ | Profile Acceleration | UNSIGNED 32 | - | $> \frac{max.\ rpm}{9{,}999\ s}$ | rpm/s | RW | The profile acceleration is given in rpm/s. |
| 6084h | $00_h$ | Profile Deceleration | UNSIGNED 32 | - | $> \frac{max.\ rpm}{9{,}999\ s}$ | rpm/s | RW | The profile deceleration is given in rpm/s |

| Index | Sub-index | Name | Data type | Upper limit | Lower limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| 6085h | $00_h$ | Quick Stop Deceleration | UNSIGNED 32 | - | $> \frac{max.\ rpm}{9{,}999\ s}$ | rpm/s | RW | The quick stop deceleration is the deceleration used to stop the motor if the 'Quick Stop' command is given and the quick stop option code (see 605Ah) is set to 2. |
| 6086h | $00_h$ | Motion Profile Type | INTEGER 16 | 0 | 0 | - | RW | The motion profile type is used to select the type of motion profile used to perform a profiled move. Only linear move is supported.<br>0: linear |
| 6098h | $00_h$ | Homing Method | INTEGER 8 | | | - | RW | The homing method object determines the method that will be used during homing<br><br>The homing modes are described in chapter 10.7 from page 36 on. |
| 6099h | $00_h$ | Homing Speeds | UNSIGNED 32 | - | - | rpm | RO | This entry in the object dictionary defines the speeds used during homing and is given in rpm |
| | $01_h$ | Search speed | UNSIGNED 32 | 10000 | 0 | rpm | RW | Speed during search for switch |
| | $02_h$ | Position speed | UNSIGNED 32 | 10000 | 0 | rpm | RW | Speed during search for zero |
| 609Ah | $00_h$ | Homing Acceleration | UNSIGNED 16 | - | $> \frac{max.\ rpm}{9{,}999\ s}$ | rpm/s | RW | The homing acceleration establishes the acceleration to be used for all accelerations and decelerations with the standard homing modes and is given in rpm/s. |
| 60C0h | $00_h$ | Interpolation sub mode select | INTEGER 16 | 0 | -1 | - | RW | 1: Linear Interpolation<br>-1: Linear Interpolation (SYNC-Tolerant) |
| 60C1h | $00_h$ | Interpolation data record | INTEGER 32 | - | - | - | RW | |
| | $01_h$ | Next Positon record | INTEGER 32 | 2147483647 | -2147483648 | Incr. | RW | Position Value. |
| 60C2h | $00_h$ | Interpolation time period | RECORD | - | - | - | RW | The actual sync cycle is calculated with Interpolation time units x Interpolation time index |
| | $01_h$ | Interpolation time units | UNSIGNED 8 | 255 | 1 | - | RW | 1 – 255 time units |

| Index | Sub-index | Name | Data type | Upper limit | Lower limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| | $02_h$ | Interpolation time index | INTEGER 8 | -3 | 0 | - | RW | –3: msec<br>–2: 10ms<br>–1: 100ms<br>0: 1s |
| 60C3h | $00_h$ | Interpolation sync definition | UNSIGNED 8 | - | - | - | RW | If ever a SYNC event is not 1 the SYNC period recalculates<br>Interpolation time units x Interpolation time index x Used SYNC event. |
| | $01_h$ | Synchronize on group | UNSIGNED 8 | 0 | 0 | - | RW | SYNC Group always 0. |
| | $02_h$ | Ip sync on every n event | UNSIGNED 8 | 255 | 0 | - | RW | Every n SYNC event is used for interpolation. |
| 60C4h | $00_h$ | Interpolation data configuration | RECORD | | | | - | |
| | $01_h$ | Maximum buffer size | UNSIGNED 32 | 1 | 1 | - | RO | Maximum buffer size. Always 1. |
| | $02_h$ | Actual buffer size | UNSIGNED 32 | 1 | 1 | - | RW | Actual buffer size. Always 1. |
| | $03_h$ | Buffer organization | UNSIGNED 8 | 0 | 0 | - | RW | FIFO |
| | $04_h$ | Buffer position | UNSIGNED 16 | 1 | 1 | - | RW | Always 1. |
| | $05_h$ | Size of data record | UNSIGNED 8 | 1 | 1 | - | WO | Always 1. |
| | $06_h$ | Buffer clear | UNSIGNED 8 | - | - | - | WO | N/A |
| 60F6h | $00_h$ | Torque Control Parameter Set | UNSIGNED 8 | - | - | - | RO | The torque control parameters |
| | $01_h$ | Gain | UNSIGNED 16 | 1000 | 1 | | RW | P-portion of the torque regulator |
| | $02_h$ | Time | UNSIGNED 16 | 1000 | 1 | | RW | I-portion of the torque regulator |
| | $03_h$ | Int-Limit | UNSIGNED 16 | 100 | 1 | | RW | I-limitation of the torque regulator |
| 60F9h | $00_h$ | Velocity Control Parameter Set | UNSIGNED 8 | - | - | | RO | The speed control parameters |

| Index | Sub-index | Name | Data type | Upper limit | Lower limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| | $01_h$ | Gain | UNSIGNED 16 | 1000 | 1 | - | RW | P-portion of the speed regulator |
| | $02_h$ | Time | UNSIGNED 16 | 1000 | 1 | - | RW | I-portion of the speed regulator |
| | $03_h$ | Int-Limit | UNSIGNED 16 | 100 | 1 | % | RW | I-limitation of the speed regulator |
| 60FBh | $00_h$ | Position Control Parameter Set | UNSIGNED 8 | - | - | - | RO | The position control parameters |
| | $01_h$ | Gain | UNSIGNED 16 | 1000 | 1 | - | RW | P-portion of the position controller |
| | $02_h$ | Time | UNSIGNED 16 | 1000 | 1 | - | RO | reserved |
| | $03_h$ | Int-Limit | UNSIGNED 16 | 100 | 1 | % | RO | reserved |
| 60FFh | $00_h$ | Target Velocity | INTEGER 32 | 10000 | -10000 | rpm | RW | The target velocity is the input for the trajectory generator in Profile Velocity Mode. |
| 6410h | $00_h$ | Motor Data | UNSIGNED 8 | - | - | - | RO | This object contains information about the connected motor |
| | $01_h$ | Pair of Poles | UNSIGNED 8 | - | - | - | RO | Number of pole pairs |
| | $02_h$ | Maximum Motor Speed | UNSIGNED 16 | - | - | rpm | RO | Maximum motor speed |
| | $03_h$ | Rated Motor Speed | UNSIGNED 16 | - | - | rpm | RO | Rated motor speed |
| | $04_h$ | Max Motor Current | UNSIGNED 16 | - | - | 0,1 A | RO | Maximal motor current |
| | $05_h$ | Nominal Motor Current | UNSIGNED 16 | - | - | 0,1 A | RO | Nominal motor current |
| | $06_h$ | Nominal Voltage | UNSIGNED 16 | - | - | V | RO | Nominal motor voltage |
| | $07_h$ | I2T-Time | UNSIGNED 16 | - | - | s | RO | Motor overload time |
| | $08_h$ | Commutation Mode | UNSIGNED 8 | - | - | - | RO | Commutation mode |
| | $09_h$ | e.m.f. constant | UNSIGNED 16 | - | - | mV/1/ min | RO | e.m.f. constant |

| Index | Sub-index | Name | Data type | Upper limit | Lower limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| | $0A_h$ | Encoder Resolution | UNSIGNED 16 | 10000 | 1 | incr. | RO | Axis resolution |
| | $0B_h$ | Main Configuration Register | UNSIGNED 32 | - | - | - | RO | Hardware configuration |
| | $0C_h$ | Motor supply | UNSIGNED 16 | 60 | 24 | V | RW | Motor supply voltage |
| | $0D_h$ | User Current Limit | UNSIGNED 8 | 100 | 0 | % | RW | Maximum current in % of peak current |
| | $0E_h$ | Sense of rotation speed regulator | UNSIGNED 8 | - | - | - | RO | Internal |
| | $0F_h$ | Direction of control speed regulator | UNSIGNED 8 | - | - | - | RO | Internal |
| | $10_h$ | Sense of rotation position regulator | UNSIGNED 8 | - | - | - | RO | Internal |
| | $11_h$ | Direction of control position regulator | UNSIGNED 8 | - | - | - | RO | Internal |
| | $12_h$ | Polarity | UNSIGNED 8 | - | - | - | RO | internal |
| 6510h | $00_h$ | Drive Data | UNSIGNED 8 | - | - | - | RO | Drive data |
| | $01_h$ | Output stage temperature warning | UNSIGNED 16 | - | - | - | RO | Output stage temperature warning threshold |
| | $02_h$ | Output stage temperature error | UNSIGNED 16 | - | - | - | RO | Output stage temperature error threshold |
| | $03_h$ | Motor temperature warning | UNSIGNED 16 | - | - | - | RO | Motor temperature warning threshold |
| | $04_h$ | Motor temperature error | UNSIGNED 16 | - | - | - | RO | Motor temperature error threshold |
| | $05_h$ | Motor supply voltage low | UNSIGNED 16 | - | - | - | RO | Under-voltage motor supply threshold |
| | $06_h$ | Motor supply over-voltage | UNSIGNED 16 | - | - | - | RO | Over-voltage motor supply threshold |
| | $07_h$ | Maximal current | UNSIGNED 16 | - | - | - | RO | Maximal current |

| Index | Sub-index | Name | Data type | Upper limit | Lower limit | Unit | Access | Description |
|---|---|---|---|---|---|---|---|---|
| | $08_h$ | Nominal current | UNSIGNED 16 | - | - | - | RO | Nominal current |
| | $09_h$ | Ballast threshold | UNSIGNED 16 | - | - | - | RO | Ballast threshold voltage |

# 13. Diagnosis and trouble shooting

## 13.1 CAN error messages

Error messages are transmitted through emergency message. The error codes correspond to the following table. Object 1003h with its subindices holds the error history with error-numbers.

## 13.2 Status LED's

| Error No. Object 1003h | CAN error COB-ID 80h + Node ID | Description | Red Led (H1) | Multicolour LED (H2) | Ready – Signal | Acknowledgement required | Brake mode |
|---|---|---|---|---|---|---|---|
| **Axis State Errors** | | | | | | | |
| 0 | - | „Enable" present, no errors (normal operating mode) | Off | Green On | On | No | - |
| 24 | 18h | Wait for enable | Off | fast blink Green | On | No | A |
| 3 | FF54h | Tracking error | On | Blink green 1x | Off | Yes | B |
| 4 | FF52h | Timeout drive | On | Blink orange 1x | Off | Yes | - |
| 5 | FF53h | Axle moving | On | Blink green 2x | Off | Yes | - |
| 6 | FF55h | Wrong ramp parameter | On | Blink orange 4x | Off | Yes | - |
| 7 | FF56h | Travel for ramp too far | On | Blink orange 4x | Off | Yes | - |
| 19 | 13h | Stop without error-message applied | Off | Blink orange 2x | On | No | - |
| 20 | FF59h | Stop with error-message applied | On | Blink orange 2x | Off | Yes | - |
| 25 | FF50h | Working mode wrong | On | Blink orange 3x | Off | Yes | - |
| 28 | FF22h | Axle not referenced | On | Blink orange 3x | Off | Yes | - |
| 31 | FF15H | Range of travel exceeded | On | Blink orange 3x | Off | Yes | - |
| 32 | FF13h | Reference lost | On | Blink orange 3x | Off | Yes | - |
| 33 | FF24h | Wrong set value mode | On | Blink orange 3x | Off | Yes | - |
| 34 | FF50H | Regulator state wrong | On | Blink orange 3x | Off | Yes | - |
| 37 | FF61h | Reference type not supported | Off | Green on | Off | Yes | - |

| Error No. Object 1003h | CAN error COB-ID 80h + Node ID | Description | Red Led (H1) | Multicolour LED (H2) | Ready – Signal | Acknowledgement required | Brake mode |
|---|---|---|---|---|---|---|---|
| 41 | FF65h | Output-stage overload | On | Blink red 3x | Off | Yes | B |
| 43 | FF67h | Limit switch + direction deviant | On | Blink green 3x | Off | Yes | B |
| 44 | FF68h | Limit switch - direction deviant | On | Blink green 3x | Off | Yes | B |
| 49 | 31h | Temperature sensor motor | On | Red on | Off | Yes | B |
| **Supply and Temperature Errors** | | | | | | | |
| 12 | 4310h | Motor temperature too high | On | Blink red 1x | Off | Yes | A |
| 13 | 4210h | Temperature output stage too high | On | Blink red 2x | Off | Yes | B |
| 14 | 3210h | Over-voltage motor | On | Blink red 3x | Off | Yes | B |
| 15 | 3220h | Under-voltage motor | On | Blink red 3x | Off | Yes | B |
| 16 | 3110h | Over-voltage logic | On | Blink red 4x | Off | Yes | B |
| 17 | 3120h | Under-voltage logic | On | Blink red 4x | Off | Yes | B |
| 21 | 2220h | Overload motor | On | Blink red 1x | Off | Yes | A |
| 30 | FF57h | Reference voltage too low | On | Blink red 4x | Off | Yes | B |
| **Communication Errors** | | | | | | | |
| 26 | 1Ah | Bus offline | Blink | Blink orange 1x | Off | Yes | A |
| 22 | 16h | Profibus login missing | Blink | Blink orange 2x | Off | Yes | - |
| 27 | FF51h | RS232/RS485 time out | Blink | Blink orange 3x | Off | Yes | A |
| 35 | 23h | CANopen error | Blink | Blink orange 1x | Off | Yes | A |
| **Internal Errors** | | | | | | | |
| 23 | FF58h | Error at initialisation | On | Red on | Off | Yes | B |

| Error No. Object 1003h | CAN error COB-ID 80h + Node ID | Description | Red Led (H1) | Multicolour LED (H2) | Ready – Signal | Acknowledgement required | Brake mode |
|---|---|---|---|---|---|---|---|
| 18 | 5530h | EEPROM Checksum error | On | Red on | Off | Yes | B |
| 36 | FF60h | Encoder error | On | Red on | Off | Yes | B |
| 42 | FF66h | Temperature sensor output stage | On | Red on | Off | Yes | B |

### 13.3 Status description:

| Error | Error message / Status | Error cause / Remedies |
|---|---|---|
| 0 | No error | - |
| 1 | Wrong or missing data | |
| 2 | Wrong or missing regulator data | |
| 3 | Tracking error | The drive is unable to follow the set value. The drive may be sluggish or even blocked. The parameter for the allowable tracking error is set too small. The acceleration is too high. Encoder signals erroneous. Incorrect direction of control. Loop gain insufficient or excessive (drive is oscillating). |
| 4 | Timeout drive | Desired position was not reached within the given time. Tune regulator settings and/or enlarge target position window. |
| 5 | Axle moving | It was attempt to send a new drive or positioning command while the drive was in motion. |
| 6 | Erroneous ramp parameter | The drive can't comply with the given data. The acceleration time is too long |
| 7 | Travel for ramp to far. | It's not possible to calculate the ramp for the given data. |
| 12 | Motor temperature (PTC sensor) | If the motor temperature rises due to heavy load the temperature can become too high. Check the load of the motor for any problems.<br>☞ The temperature error can be acknowledged when the temperature of the controller went down again. |
| 13 | Output-stage temperature too high | The temperature of the controller is too high. For safety reason the controller has been shut down. Check the nominal current/maximum current of the motor and machine.<br>☞ The temperature error can be acknowledged only when the temperature of the controller went down again. |
| 14 | Over-voltage motor | When the Motor source exceeds one of the limits defined in the settings this error is given. This can happen due to spikes in the supply voltages. In most cases, the over-voltage can also occur if the motor needs to slow down a heavy load. Check machine and state. An external ballast resistor can help. |
| 15 | Under-voltage motor | The motor voltage undercuts the limits given by parameter. This may be caused by spikes on the supply voltage. In most cases under-voltage is a result of a heavy duty. A more powerful power supply may help.<br>☞ If the enable input is low, the motor-voltage is not monitored. There is no error message if the voltage is switched on ahead of a start. |

| Error | Error message / Status | Error cause / Remedies |
|---|---|---|
| 16 | Over-voltage logic | In most cases this can happen due to spikes in the supply voltages. If the motor and logic use the same source the over-voltage can occur if the motor needs to slow down a heavy load. Check machine and state. An external ballast resistor can help. It's recommended to use separate power supplies for motor and logic. |
| 17 | Under-voltage logic | The logic supply falls below 18V. Maybe there are drop-outs in the supply voltage. Insufficient filtering of the supply voltage. A more powerful power supply may help. |
| 18 | EEPROM checksum error | Unit was switched off, during parameter input. Check parameters. |
| 19 | Stop applied<br>– Status Information - | "Stop without error message" present<br>There was a falling edge on the stop input. This can be caused by user or by a power loss on the stop input. No acknowledgement is required. |
| 20 | Stop applied<br>- Error message - | "Stop with error message" present.<br>There was a falling edge on the stop input. This can be caused by user or by a power loss on the stop input. Check machine and state. If no error or dangerous situation can be discovered try to acknowledge the "Stop" Error with a raising edge on the enable input. Be aware that acknowledgement can only be successful when the error which caused this state no longer exists. |
| 21 | Overload motor | Demanded motor power too high, The drive may be sluggish or even blocked, high acceleration values, high motor load with high ambient temperature |
| 22 | reserved | |
| 23 | Error at initialisation | Internal initialisation error. No access to EEPROM. Please consult your dealer. |
| 24 | Waiting for enable | All configuration and controller data is correct. The motor control is able to operate and waits for enable. |
| 25 | Working mode wrong | A function was selected which is in the current working mode not possible. For example a position value was given to the speed regulator. |
| 26 | Bus off-line | The connection to the fieldbus has failed.<br>☞ The LED H3 is flashing if the bus connection couldn't be established after power-on. |
| 27 | RS232/RS485-Trigger timeout | The RS232/RS485 interface link was disconnected, while the axle was moving with a drive command initiated from RS232/RS485. |
| 28 | Axle not referenced | A drive command was given whilst no reference run was done. |
| 29 | Wrong value<br>– Status Information – | An incorrect value was transmitted e.g. speed > $n_{max}$ |
| 30 | Reference voltage too low | The internal reference voltage is too low. Please check the logic supply. |
| 31 | Range of travel exceeded. | The maximum countable number of increments has been exceeded. |
| 32 | Reference lost | The motor moved after the voltage was switched off.<br>The procedure for position storage safe against power outage was not accomplished. |
| 33 | Wrong set value mode | Clock/Direction inputs are active and it was tried to start a record or another set value was given. |
| 34 | Wrong regulator state | The response time of the brake was not taken care for and a new drive command given. |
| 35 | CANopen error | Reserved error |
| 36 | Encoder error | A not existing encoder was selected in the parameters. |
| 37 | Reference type not supported | A not supported referencing mode was selected. No error message, only status information. |
| 38 | reserved | |
| 39 | reserved | |

| Error | Error message / Status | Error cause / Remedies |
|---|---|---|
| 40 | reserved | |
| 41 | Output-stage overload | Possible cause: The drive may be sluggish or even blocked, high acceleration values, high load with high ambient temperature, motor faulty, cables broken |
| 42 | Temperature sensor output stage error | The value is not plausible. There is a malfunction of the sensor. |
| 43 | Limit switch + direction deviant | The limit switch in + direction was activated while the motor was moving in – direction |
| 44 | Limit switch – direction deviant | The limit switch in - direction was activated while the motor was moving in + direction |
| 45 | reserved | |
| 46 | reserved | |
| 47 | reserved | |
| 48 | reserved | |
| 49 | Motor temperature sensor error | The value is not plausible. There is a malfunction of the sensor. |
| 55 | EEPROM access error | queue error |
| 56 | EEPROM access error | write error |
| 57 | EEPROM access error | read error |

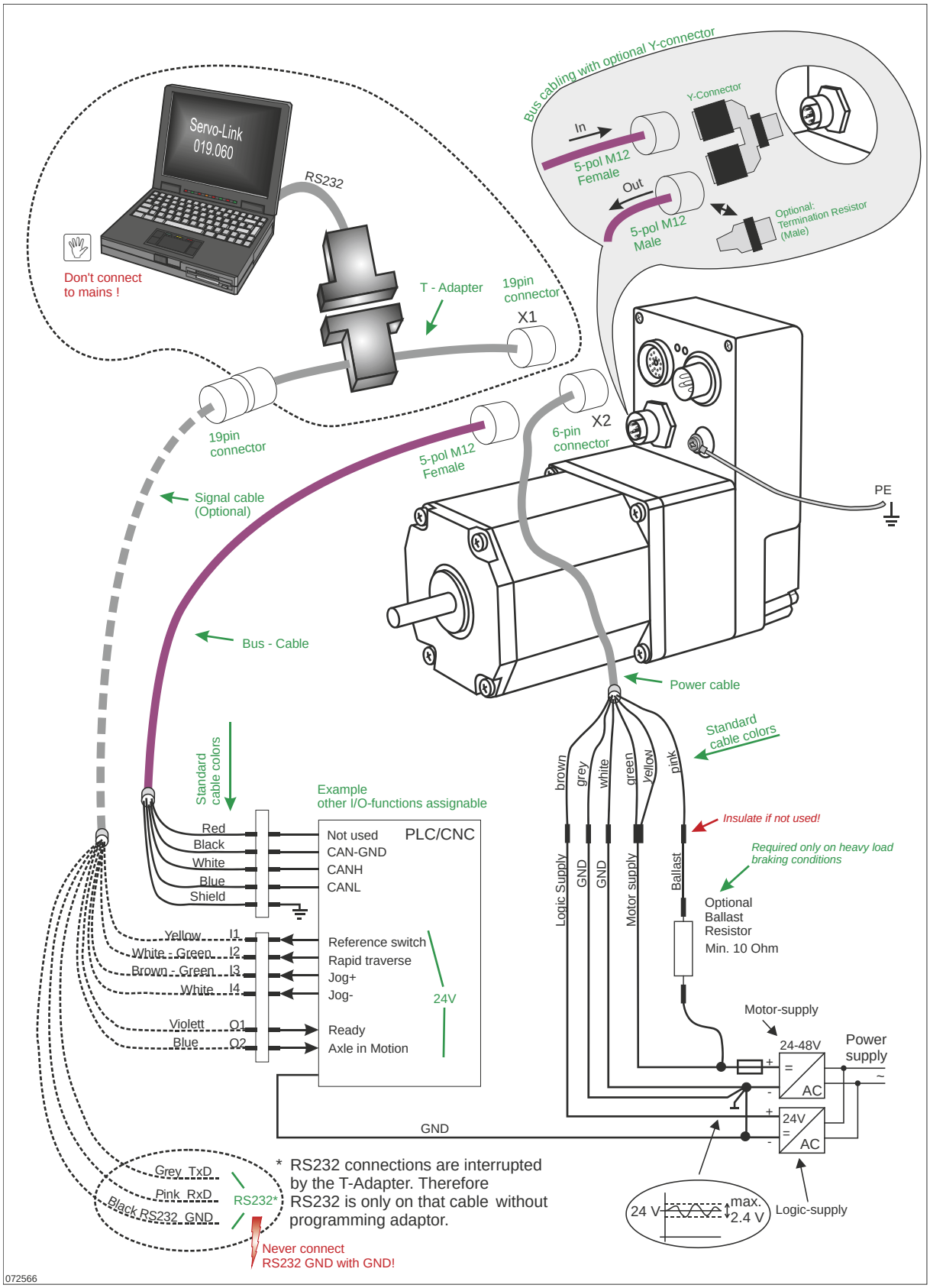
☞

**Note**

All error messages must be acknowledged through the appropriate command through fieldbus

See transitions in the state machine in chapter 10.3 on page 30.

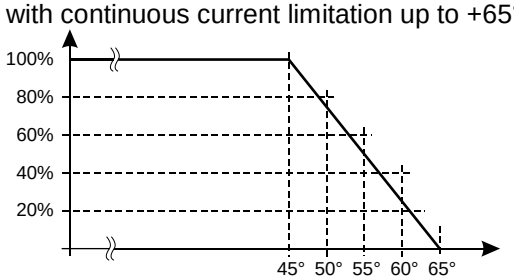
## 14. Schematic view



# 15. Technical Data

### 15.1.1 General

| | |
|---|---|
| Ambient temperature: | -25 ... +45 °C<br>with continuous current limitation up to +65°C |




| | |
|---|---|
| Storage temperature: | -40 ... +85 °C |
| Humidity class: | RH 1 |
| Emission: | EN 61800-3 Cat. C2 / Industrial |
| Interference: | EN 61800-3 Cat. C2 / Industrial |

### 15.1.2 Power supply

Input voltage **Logic & I/O – supply**:

| | |
|---|---|
| Nominal value: | 24 VDC |
| Admissible range: | 18 – 30 VDC |
| Ripple: | ≤ 10 % |
| Reverse polarity protection: | With diode |
| External fuse: | max 1 A / medium slow |
| Current consumption: | ca. 0,1 A at 24 VDC |
| Resistance to drop-outs: | |
| Duration of drop-outs: | 10 msec |
| Repetition: | 1 s |

Input voltage **Motor – supply**:

| | |
|---|---|
| esiMot6/1 Nominal value: | 24V VDC |
| esiMot6/1 Admissible range: | 18 – 48V DC |
| esiMot6/5 Nominal value: | 48V DC |
| esiMot6/5 Admissible range: | 18 – 60V DC |
| esiMot7/1 Nominal value: | 48V DC |
| esiMot7/1 Admissible range: | 18 – 60V DC |
| esiMot7/2 Nominal value: | 48V DC |
| esiMot7/2 Admissible range: | 18 – 60V DC |
| Galvanic isolation: | no galvanic isolation to logic |
| Reverse polarity protection: | Yes |
| External fuse: | max 10 A / medium slow |
| Current consumption: | Up to 14A, depending on load / motor type see table on page 91 |

### 15.1.3 Analogue input (optional)

| | |
|---|---|
| Type: | Differential input |
| Resolution: | 11 Bit |
| Max. Error: | 2% |
| Galvanic isolation: | No |

#### *15.1.3.1 Voltage input*

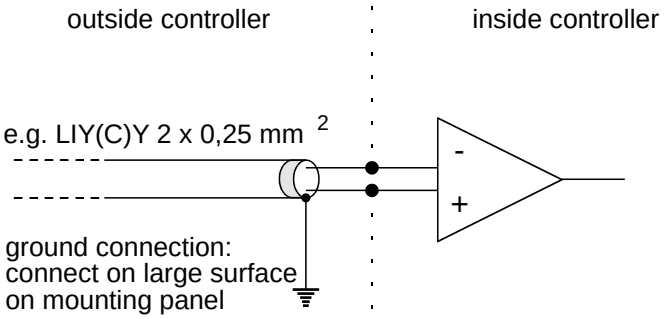
| | |
|---|---|
| Nominal value: | ± 10 V |
| Maximal input voltage:[1)] | +30 V |
| Minimal input voltage:[1)] | -12 V |

1) in relation to power supply ground

| | |
|---|---|
| Input impedance: | 27 kOhm |

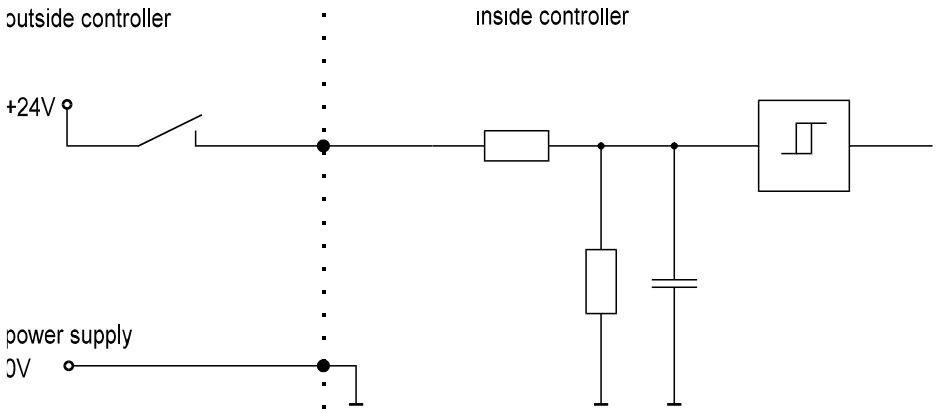
#### *15.1.3.2 Current input (Custom specific, only on request)*

| | |
|---|---|
| Nominal value: | ± 20 mA |
| Maximal input current: | ± 25 mA |
| Input impedance / load resistor: | 100 Ohm |



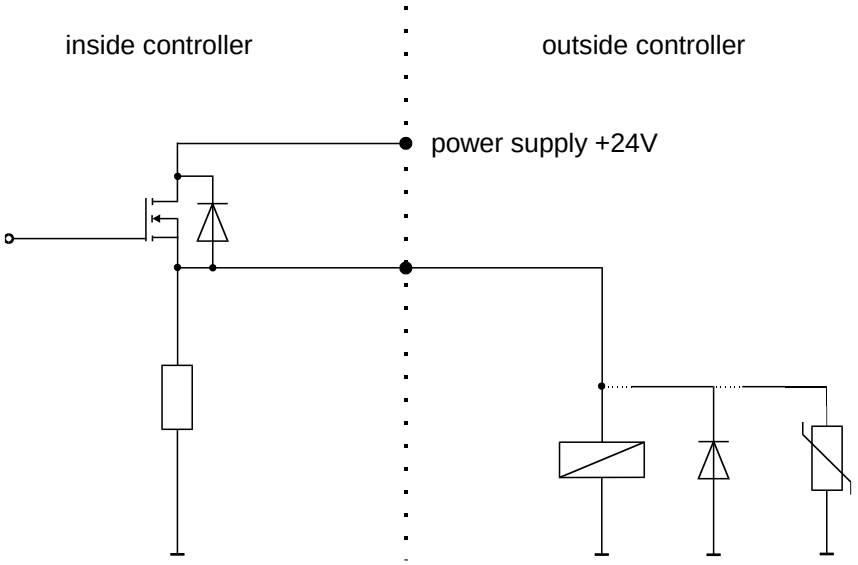
### 15.1.4 Digital inputs

| | |
|---|---|
| Number of inputs: | Up to 8 (depending on options) |
| Input voltage. | |
| Nominal value: | +24 VDC |
| Maximal voltage: | +30 VDC |
| Threshold High-Level: | +12 VDC |
| Threshold low-level: | + 3 VDC |
| Minimal input voltage: | 0 VDC |
| Input current at nominal voltage: | typ.: 1,5 mA |
| Delay time inputs: | typ.: 10 µsec |
| Low->High: | typ.: 10 µsec |
| High->Low: | |
| Galvanic isolation: | No |



### 15.1.5 Digital outputs: 24 VDC

| | |
|---|---|
| Number of outputs: | 1 ... 4 (depending on options) |
| Type of output: | Transistor output (MOS-FET) |
| Output current at high signal: | |
| Nominal value: | 0,5 A |
| Residual current at low signal: | 250 µA |
| Power supply: | +24V logic supply |
| Short circuit protection: | yes, self-reactivating |
| Current limitation: | 1 A |
| Trip current: | 1,0 A |
| External protection with inductive loads: | Varistor (e.g. Murrelektronik VG-A/24) or quenching diode (e.g. Murrelektronik LG-A01) recommended |
| Parallel connection admissible: | No |
| Delay time outputs: | typ.: 100 µsec |
| Low->High: | typ.: 100 µsec |
| High->Low: | |
| Galvanic isolation: | No |



## 15.2 Ballast circuit

Threshold: 60V (default value)

Depended on operating voltage.

Internal ballast resistor:

| | |
|---|---|
| Peak power/100ms | 75W |
| Max continuous power | 9W |

It's possible to connect an external ballast resistor parallel to the internal resistor. An external ballast resistor is only required on extreme situations.

External ballast resistor:

| | |
|---|---|
| Minimal value | 10 Ohm |
| Peak power / 100ms /10 Ohm | 422W |
| Max continuous power | 42W |

**Attention**

The ballast circuit is not short circuit proof. Wrong wiring of a ballast resistor could destroy the ballast circuit and the motor regulator.

**Attention**

The surface of the external brake resistor can reach temperatures of up to 450°C. Mount the resistor only in areas where heat can't cause a fire. Take care that no flammable or heat-sensitive materials are close to the resistor.

## 15.3 Standstill brake (Option)

Via a digital input the optional standstill brake of the motor can be switched in.

The supply of the brake is the motor supply. The voltage for the brake is limited internal to a maximum of 24V. The maximum current is $I_{max}$= 0,5A.

## 15.4 RS232-Interface

The parameters of the motor regulator are set via the serial interface. There is PC-Software "Servo Link" 019.060.xx available to set up the parameters. See chapter 18 on page 95.

**Attention**

For continuous operation, the interface must be protected with an external galvanic isolation.

## 15.5 Motors

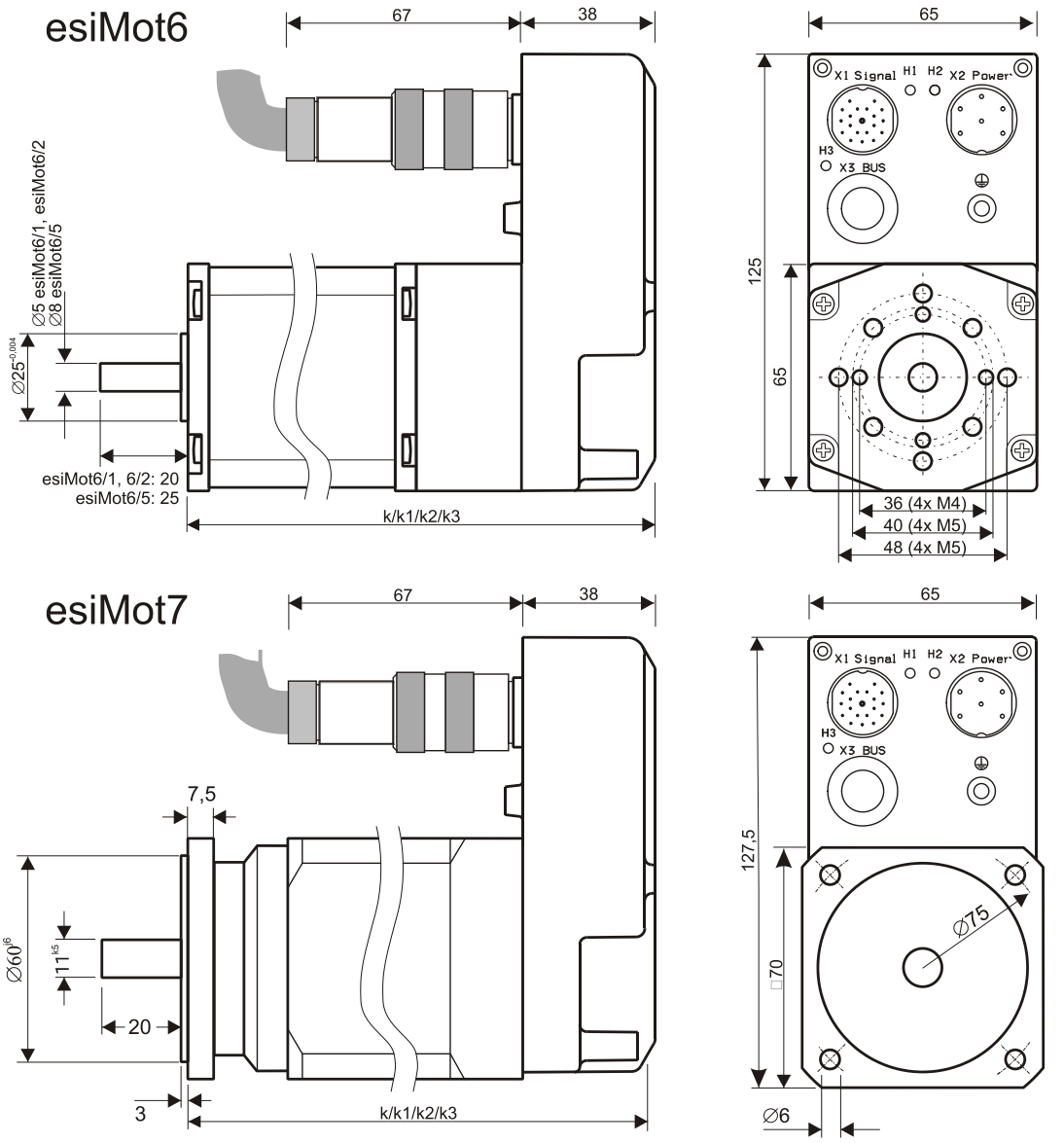
| esiMot Size | | 6/1 | 6/2 | 6/5 | 7/1 | 7/1 | 7/2 | 7/2 |
|---|---|---|---|---|---|---|---|---|
| Motor power (S3) | [W] | 120 | 180 | 260 | 500 | 600 | 400 | 480 |
| Rated power | [W] | 70 | 110 | 200 | 340 | 430 | 300 | 380 |
| Nominal torque | [Nm] | 0,22 | 0,35 | 0,65 | 1,1 | 1,1 | 1,8 | 1,8 |
| Impulse torque | [Nm] | 0,50 | 0,70 | 1,20 | 1,6 | 1,6 | 2,3 | 2,3 |
| Rated speed | [$min^{-1}$] | 3000 | 3000 | 3000 | 3000 | 3750 | 1600 | 2000 |
| Inertia rotative | [$gcm^2$] | 71,6 | 128 | 172 | 530 | 530 | 530 | 530 |
| Motor supply | [V] | 24 | 24 | 48 | 48 | 60 | 48 | 60 |
| Nominal current | [A] | 4,0 | 5,6 | 3,5 | 9,0 | 9,0 | 8,0 | 8,0 |
| Peak current | [A] | 14 | 14 | 14 | 14 | 14 | 14 | 14 |
| Logic supply | [V] | 24 | 24 | 24 | 24 | 24 | 24 | 24 |
| Axle resolution without incremental encoder | [lpr] | 30 | 30 | 30 | - | - | - | - |
| Axle resolution with incremental encoder (Positioning mode) | [lpr] | 2048 | 2048 | 2048 | 4096 | 4096 | 4096 | 4096 |
| Axle resolution with absolute encoder (Positioning mode) | [lpr] | 4096 x 4096 | 4096 x 4096 | 4096 x 4096 | 4096 x 4096 | 4096 x 4096 | 4096 x 4096 | 4096 x 4096 |
| Protection class* (mounting specific) | IP | 64 | 64 | 64 | 64 | 64 | 64 | 64 |
| Length k | [mm] | 148 | 173 | 198 | 186 | 186 | 186 | 186 |
| Length k1 (with brake) | [mm] | ✧ | ✧ | ✧ | 236 | 236 | 236 | 236 |
| Length k2 (with absolute encoder) | [mm] | 179 | 204 | 229 | 199 | 199 | 199 | 199 |
| Weight ca. | [kg] | 1,37 | 1,8 | 2,3 | 2,3 | 2,3 | 2,3 | 2,3 |
| Weight with brake ca. | [kg] | ✧ | ✧ | ✧ | 2,48 | 2,48 | 2,48 | 2,48 |

✧ On Request

**Attention**

* All connectors have to be plugged in and tightened. Use a protection cap, if no connector is required.
IP40 at motor axle. With optional axle seal IP64. Mounted gear-boxes provide no additional protection unless specially sealed. For protection class of the gear-box consult gear-box data sheet.

## 15.6 Dimensions and position map




072390_2

| Size | | 6/1 | 6/2 | 6/5 | 7/1 | 7/2 |
|---|---|---|---|---|---|---|
| Length k | [mm] | 148 | 173 | 198 | 186 | 186 |
| Length k1 (Brake) | [mm] | On request | | | 236 | 236 |
| Length k2 (absolute encoder) | [mm] | 179 | 204 | 229 | 199 | 199 |
| Weight ca. | [kg] | 1,4 | 1,8 | 2,3 | 2,3 | 2,3 |
| Weight with brake ca. | [kg] | On request | | | 2,5 | 2,5 |

# 16. Connections

## 16.1 Signal connector X1

| X1 Pin: | Function | Remarks |
|---|---|---|
| A | Digital Output 1 | |
| B | Digital output 2 | |
| C | Optional: Digital input 7 / Digital Output 3 | Option |
| D | Optional: Digital input 8 / Digital Output 4 | Option |
| E | Digital input 1 | |
| F | Digital input 2 | |
| G | Digital input 3 | |
| H | Digital input 4 | |
| I | Analogue input + (Option) | 0-10V; +/- 10V; 0/4 – 20mA |
| K | Analogue input - (Option) | 0-10V; +/- 10V; 0/4 – 20mA |
| L | Sensor supply +24V | |
| M | Sensor supply 0V | (GND) |
| N | Do not connect | |
| O | Do not connect | |
| P | Do not connect | |
| R | Do not connect | |
| S | RS232 – Interface TxD | |
| T | RS232 – Interface RxD | |
| U | RS232 – Interface reference potential | Don't connect to GND |

Signal connector: male cable solder-connector 630.01409-1

## 16.2 Power supply connector X2

| X2 Pin-No.: | Function | Remarks |
|---|---|---|
| 1 | Power supply motor +24V / +48V | Power supply for |
| 2 | Power supply motor +24V / +48V | logic and motor |
| 3 | Power supply logic +24V | |
| 4 | GND (reference for logic and motor) | |
| 5 | GND (reference for logic and motor) | |
| 6 | Output for external ballast resistor | (connect other side of resistor to motor +) |

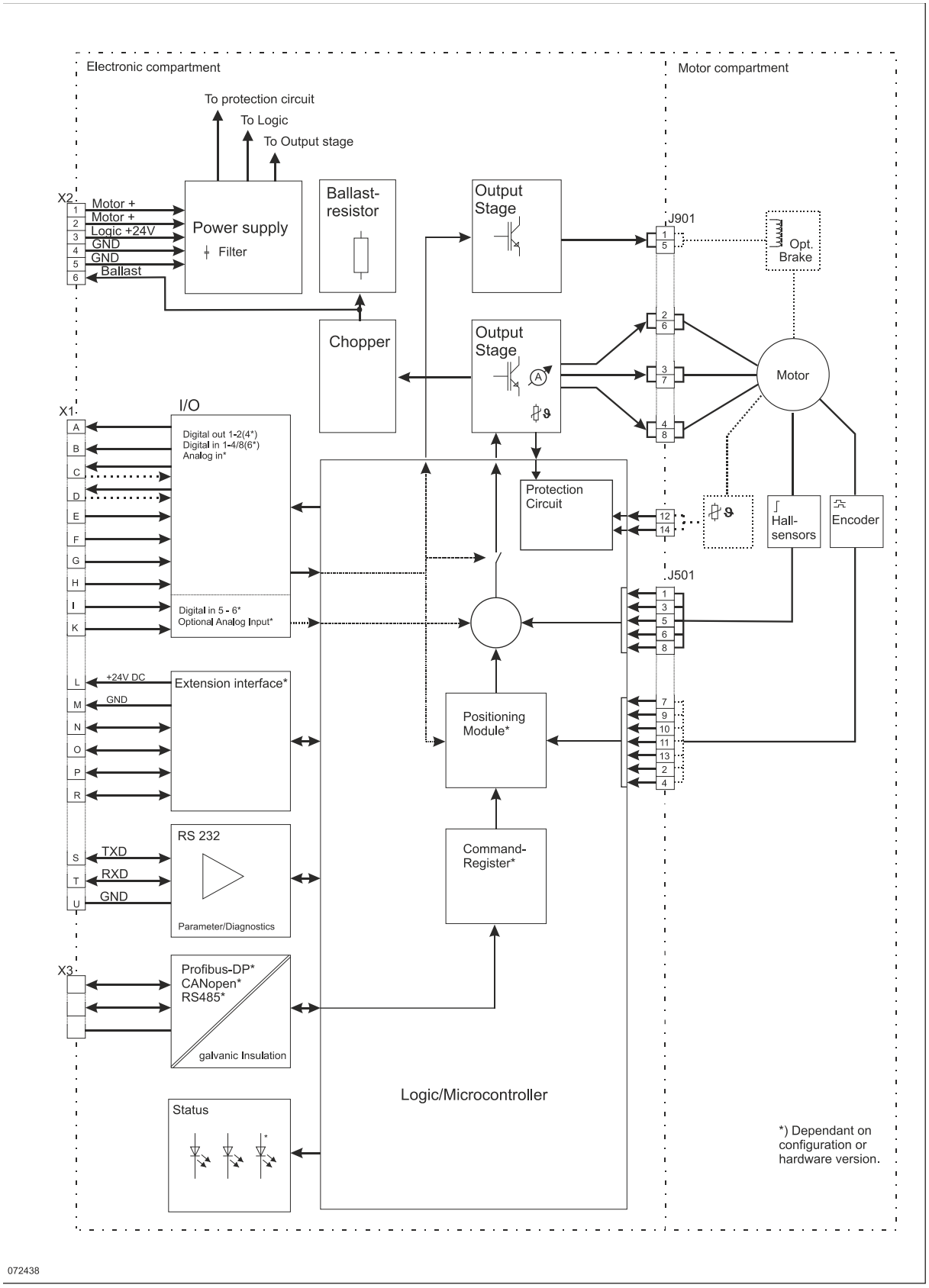
Power connector: female cable solder-connector 630.01409-0

## 16.3 Fieldbus connector X3

| X3 Pin-No.: | Function | Remarks |
|---|---|---|
| 1 | Shield | CANopen |
| 2 | Not used | |
| 3 | CAN-GND | |
| 4 | CANH | |
| 5 | CANL | |

CAN-Bus Connector: female cable connector 630.01409-3

# 17. Block diagram



# 18. Accessories

## 18.1 Programming kit Servo Link

Consisting of:

- PC-Software for parameter adjustment, online-diagnostics and programming.
- Programming adapter and RS232 cable.

## 18.2 Assembled feed lines

- Standard and drag-chain suitable cables
- Signal, Bus and Power-cable
- With plug on one or both sides

Signal cable:

| Cable colour | Function | |
|---|---|---|
| Violet | Digital Output 1 | |
| Blue | Digital Output 2 | |
| Red | Digital Output 3 / Digital Input 7 | |
| Green | Digital Output 4 / Digital Input 8 | |
| Yellow | Digital Input 1 | |
| White-Green | Digital Input 2 | |
| Brown-Green | Digital Input 3 | |
| White | Digital Input 4 | |
| Red-Blue | Digital Input 5 / Analogue Input + | |
| Grey-Pink | Digital Input 6 / Analogue Input - | |
| Yellow-Brown | Sensor supply +24V (output) | |
| White-Yellow | Sensor supply GND (output) | |
| Grey | RS232 - Interface TxD | |
| Pink | RS232 - Interface RxD | |
| Black | RS232 - Interface Reference Potential (Don't connect to GND) | |
| Brown | Don't connect | |
| White - Grey | Don't connect | Spare on some cables |
| Grey - Brown | Don't connect | |

Power cable

| Cable colour | Function |
|---|---|
| Yellow | Motor supply +24V ... +60V |
| Green | Motor supply +24V ... +60V |
| Brown | Logic supply +24V |
| White | GND (Reference potential for Motor and Logic) |
| Grey | GND (Reference potential for Motor and Logic) |
| Pink | Output for external ballast resistor |

Power cable (Alternative)

| Cable No. | Function |
|---|---|
| 1 | Motor supply +24V ... +60V |
| 2 | Motor supply +24V ... +60V |
| 3 | Logic supply +24V |
| 4 | GND (Reference potential for Motor and Logic) |
| 5 | GND (Reference potential for Motor and Logic) |
| 6 | Output for external ballast resistor |

CAN - bus Cable

| Cable colour | Function |
|---|---|
| Shield | Shield |
| Red | Not used |
| Black | CAN-GND |
| White | CANH |
| Blue | CANL |

## 18.3 Plugs

- Signal, Bus and Power connector

Enable the customer to assemble the feed-line to suit his needs best.

## 18.4 Bus accessories

- Termination resistors
- Y-Adapter

## 18.5 Signal Converter

Converts 24V PLC-Signals to 5V RS422

## 18.6 Power supplies

Various voltages and current capabilities for DIN-Rail mounting.

## 18.7 Planetary Gear boxes

Different sizes and quality grades from cost effective to high precision.

One to three stages with output torque's up to 100Nm

# 19. Models

## 19.1 esiMot

esiMotT/S.ABC.DE.FFF

T/S: Type series and size of motor
T/S = SC for Servo-controller (only electronics, no motor)

A: Basic Version

1 = Amplifier mode, Analogue set value input, 4x digital In, 1x digital Out, Encoder out
2 = PLC interface, 8x digital In, 2x digital Out
3 = PROFIBUS-DP, 4x digital In, 2x digital Out
4 = CANopen, 4x digital In, 2x digital Out
5 = RS485, 4x digital In, 2x digital Out
6 = EtherCAT CoE, 4x digital In, 2x digital Out
7 = PROFINET-IO, 4x digital In, 2x digital Out

B: Position acquisition
1 = Hall sensor (not available for esiMot7)
2 = Incremental encoder (esiMot6=2048 incr./rev; esiMot7=4096 incr./rev)
3 = Absolute encoder

C: Standstill brake
0 = No brake
1 = Brake

D: Optional I/O Extension (only on fieldbus-versions A = 3-7)
0 = 4/2 dig. I/O(standard)
1 = 6/2 dig. I/O (two additional inputs)
2 = 4/2 dig. I/O + Analogue input
3 = 4/1 dig. I/O + Analogue input, incremental encoder outputs
4 = 4/4 dig. I/O (two additional outputs)

E: Optional address switches (only on fieldbus-versions A = 3-5)
0 = No switch
1 = Address switches

F: Special options
3 digits specify custom versions
To be omitted on standard types.

## 19.2 Gearboxes

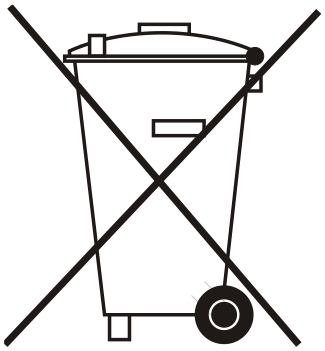
Fitted gearboxes have their own label and are not coded into the esiMot identifier. For types and sizes, please consult separate gearbox brochure.

## 20. Disposal

This product consists of various materials. The following table shows the correct end-of-life treatment.

| **Material** | **Recycle** | **Dispose** |
|---|---|---|
| Metal | Yes | No |
| Plastics material | Yes | No |
| Printed circuit board | No | Yes |

Dispose the appropriate materials in accordance to the valid environmental protection laws.



Don’t throw hazardous substances in domestic waste!

## 21. Bibliography

[1] CiA DR303-3 Indicator specification by CAN in Automation international users and manufacturers group e.V.
[2] CiA DR303-3 Indicator specification by CAN in Automation international users and manufacturers group e.V.
[3] CiA Draft Standard 301 CANopen Application Layer and Communication Profile by CAN in Automation international users and manufacturers group e.V.
[4] CiA Draft Standard Proposal 402 Device Profile Drives and Motion Control by CAN in Automation international users and manufacturers group e.V.