

Betriebsanleitung

**servoTEC S2
CANopen-Handbuch**

Ausgabe: Januar 2012
Artikel-Nr.: 1086954

**IEF Werner GmbH
Wendelhofstraße 6
78120 Furtwangen
Telefon: 07723/925-0
Telefax: 07723/925-100
www.IEF-Werner.de**

Änderungshistorie

Dokumentencode	Datum	Änderung
MAN_DE_1086954_CANopen_Handbuch_servoTEC_S2_R1a.doc	Januar 2012	Neuerstellung dieses deutschen Dokuments (Release R1a)

Warenzeichen und Warennamen sind ohne Gewährleistung der freien Verwendbarkeit benutzt. Bei der Erstellung der Texte und Beispiele wurde mit großer Sorgfalt vorgegangen. Trotzdem können Fehler nicht ausgeschlossen werden. Die IEF Werner GmbH kann für fehlende oder fehlerhafte Angaben und deren Folgen weder eine juristische Verantwortung noch irgendeine Haftung übernehmen.

Die IEF Werner GmbH behält sich das Recht vor, ohne Ankündigung die Software oder Hardware oder Teile davon, sowie die mitgelieferten Druckschriften oder Teile davon zu verändern oder zu verbessern.

Alle Rechte der Vervielfältigung, der fotomechanischen Wiedergabe, auch auszugsweise sind ausdrücklich der IEF Werner GmbH vorbehalten.

Für Verbesserungsvorschläge und Hinweise auf Fehler sind wir jederzeit dankbar.

© Januar 2012, IEF Werner GmbH

Inhaltsverzeichnis

1	Allgemeines	12
1.1	Dokumentation	12
1.2	CANopen.....	13
2	Sicherheitshinweise für elektrische Antriebe und Steuerungen	14
2.1	Verwendete Symbole	14
2.2	Allgemeine Hinweise.....	15
2.3	Hinweise vor der ersten Inbetriebnahme	16
2.4	Gefahren durch falschen Gebrauch.....	17
2.5	Sicherheitshinweise	18
2.5.1	Allgemeine Sicherheitshinweise	18
2.5.2	Sicherheitshinweise bei Montage und Wartung	19
2.5.3	Schutz gegen Berühren elektrischer Teile	21
2.5.4	Schutz durch Schutzkleinspannung (PELV) gegen elektrischen Schlag	22
2.5.5	Schutz vor gefährlichen Bewegungen	23
2.5.6	Schutz gegen Berühren heißer Teile.....	24
2.5.7	Schutz bei Handhabung und Montage	25
3	Verkabelung und Steckerbelegung	26
3.1	Anschlussbelegungen.....	26
3.2	Verkabelungs-Hinweise	27
4	Aktivierung von CANopen	28
4.1	Übersicht.....	28
5	Zugriffsverfahren.....	30
5.1	Einleitung	30
5.2	SDO-Zugriff	31
5.2.1	SDO-Sequenzen zum Lesen und Schreiben	32
5.2.2	SDO-Fehlermeldungen.....	33
5.2.3	Simulation von SDO-Zugriffen über RS232	34
5.3	PDO-Message.....	35
5.3.1	Beschreibung der Objekte	36
5.3.2	Objekte zur PDO-Parametrierung	39
5.3.3	Aktivierung der PDOs	44
5.4	SYNC-Message	45
5.5	EMERGENCY-Message	46
5.5.1	Übersicht.....	46
5.5.2	Aufbau der EMERGENCY-Message	47
5.5.3	Beschreibung der Objekte	50
5.5.3.1	Objekt 1003 _H : pre_defined_error_field	50
5.6	Netzwerkmanagement (NMT-Service)	52
5.7	Bootup.....	54
5.7.1	Übersicht.....	54
5.7.2	Aufbau der Bootup-Nachricht	54
5.8	Heartbeat (Error Control Protocol).....	54
5.8.1	Übersicht.....	54
5.8.2	Aufbau der Heartbeat-Nachricht.....	55
5.8.3	Beschreibung der Objekte	56

5.8.3.1	Objekt 1017 _h : producer_heartbeat_time	56
5.9	Nodeguarding (Error Control Protocol)	57
5.9.1	Übersicht	57
5.9.2	Aufbau der Nodeguarding-Nachrichten	57
5.9.3	Beschreibung der Objekte	59
5.9.3.1	Objekt 100C _h : guard_time	59
5.9.3.2	Objekt 100D _h : life_time_factor	59
5.10	Tabelle der Identifier	60
6	Parameter einstellen	61
6.1	Parametersätze laden und speichern	61
6.1.1	Übersicht	61
6.1.2	Beschreibung der Objekte	63
6.1.2.1	Objekt 1011 _h : restore_default_parameters	63
6.1.2.2	Objekt 1010 _h : store_parameters	64
6.2	Kompatibilitäts-Einstellungen	65
6.2.1	Übersicht	65
6.2.2	Beschreibung der Objekte	65
6.2.2.1	In diesem Kapitel behandelte Objekte	65
6.2.2.2	Objekt 6510 _h _F0 _h : compatibility_control	65
6.3	Umrechnungsfaktoren (Factor Group)	68
6.3.1	Übersicht	68
6.3.2	Beschreibung der Objekte	69
6.3.2.1	In diesem Kapitel behandelte Objekte	69
6.3.2.2	Objekt 6093 _h : position_factor	69
6.3.2.3	Objekt 6094 _h : velocity_encoder_factor	72
6.3.2.4	Objekt 6097 _h : acceleration_factor	74
6.3.2.5	Objekt 607E _h : polarity	76
6.4	Endstufenparameter	77
6.4.1	Übersicht	77
6.4.2	Beschreibung der Objekte	78
6.4.2.1	Objekt 6510 _h _10 _h : enable_logic	78
6.4.2.2	Objekt 6510 _h _30 _h : pwm_frequency	79
6.4.2.3	Objekt 6510 _h _3A _h : enable_enhanced_modulation	79
6.4.2.4	Objekt 6510 _h _31 _h : power_stage_temperature	80
6.4.2.5	Objekt 6510 _h _32 _h : max_power_stage_temperature	81
6.4.2.6	Objekt 6510 _h _33 _h : nominal_dc_link_circuit_voltage	81
6.4.2.7	Objekt 6510 _h _34 _h : actual_dc_link_circuit_voltage	82
6.4.2.8	Objekt 6510 _h _35 _h : max_dc_link_circuit_voltage	82
6.4.2.9	Objekt 6510 _h _36 _h : min_dc_link_circuit_voltage	83
6.4.2.10	Objekt 6510 _h _37 _h : enable_dc_link_undervoltage_error	83
6.4.2.11	Objekt 6510 _h _40 _h : nominal_current	84
6.4.2.12	Objekt 6510 _h _41 _h : peak_current	85
6.5	Stromregler und Motoranpassung	86
6.5.1	Übersicht	86
6.5.2	Beschreibung der Objekte	87
6.5.2.1	Objekt 6075 _h : motorRatedCurrent	87
6.5.2.2	Objekt 6073 _h : max_current	88
6.5.2.3	Objekt 604D _h : pole_number	88
6.5.2.4	Objekt 6410 _h _03 _h : iit_time_motor	89
6.5.2.5	Objekt 6410 _h _04 _h : iit_ratio_motor	89
6.5.2.6	Objekt 6510 _h _38 _h : iit_error_enable	90

6.5.2.7	Objekt 6410h_10h: phase_order	90
6.5.2.8	Objekt 6410h_11h: encoder_offset_angle	91
6.5.2.9	Objekt 6410h_14h: motor_temperature_sensor_polarity	92
6.5.2.10	Objekt 6510h_2Eh: motor_temperature	92
6.5.2.11	Objekt 6510h_2Fh: max_motor_temperature	93
6.5.2.12	Objekt 60F6h: torque_control_parameters	94
6.6	Drehzahlregler	95
6.6.1	Übersicht	95
6.6.2	Beschreibung der Objekte	95
6.6.2.1	Objekt 60F9h: velocity_control_parameters	95
6.6.2.2	Objekt 2073h: velocity_display_filter_time	97
6.7	Lageregler (Position Control Function)	98
6.7.1	Übersicht	98
6.7.2	Beschreibung der Objekte	100
6.7.2.1	In diesem Kapitel behandelte Objekte	100
6.7.2.2	Betroffene Objekte aus anderen Abschnitten	101
6.7.2.3	Objekt 60FBh: position_control_parameter_set	101
6.7.2.4	Objekt 6062h: position_demand_value	103
6.7.2.5	Objekt 202Dh: position_demand_sync_value	103
6.7.2.6	Objekt 6064h: position_actual_value	104
6.7.2.7	Objekt 6065h: following_error_window	104
6.7.2.8	Objekt 6066h: following_error_time_out	105
6.7.2.9	Objekt 60FAh: control_effort	105
6.7.2.10	Objekt 6067h: position_window	106
6.7.2.11	Objekt 6068h: position_window_time	106
6.7.2.12	Objekt 6510h_22h: position_error_switch_off_limit	107
6.7.2.13	Objekt 607Bh: position_range_limit	108
6.7.2.14	Objekt 6510h_20h: position_range_limit_enable	109
6.7.2.15	Objekt 2030h: set_position_absolute	110
6.8	Sollwert-Begrenzung	111
6.8.1	Beschreibung der Objekte	111
6.8.1.1	In diesem Kapitel behandelte Objekte	111
6.8.1.2	Objekt 2415h: current_limitation	111
6.8.1.3	Objekt 2416h: speed_limitation	113
6.9	Geberanpassungen	114
6.9.1	Übersicht	114
6.9.2	Beschreibung der Objekte	114
6.9.2.1	In diesem Kapitel behandelte Objekte	114
6.9.2.2	Objekt 2024h: encoder_x2a_data_field	115
6.9.2.3	Objekt 2026h: encoder_x2b_data_field	116
6.9.2.4	Objekt 2025h: encoder_x10_data_field	118
6.10	Inkrementalgeberemulation	120
6.10.1	Übersicht	120
6.10.2	Beschreibung der Objekte	120
6.10.2.1	In diesem Kapitel behandelte Objekte	120
6.10.2.2	Objekt 201Ah: encoder_emulation_data	120
6.10.2.3	Objekt 2028h: encoder_emulation_resolution	121
6.11	Soll-/Istwertaufschaltung	122
6.11.1	Übersicht	122
6.11.2	Beschreibung der Objekte	122
6.11.2.1	In diesem Kapitel behandelte Objekte	122

6.11.2.2	Objekt 201F _h : commutation_encoder_select	123
6.11.2.3	Objekt 2021 _h : position_encoder_selection	124
6.11.2.4	Objekt 2022 _h : synchronisation_encoder_selection.....	125
6.11.2.5	Objekt 202F _h : synchronisation_selector_data.....	126
6.11.2.6	Objekt 2023 _h : synchronisation_filter_time	127
6.12	Analoge Eingänge.....	128
6.12.1	Übersicht.....	128
6.12.2	Beschreibung der Objekte	128
6.12.2.1	2400 _h : analog_input_voltage (Eingangsspannung).....	128
6.12.2.2	Objekt 2401 _h : analog_input_offset (Offset Analogeingänge)	129
6.13	Digitale Ein- und Ausgänge	130
6.13.1	Übersicht.....	130
6.13.2	Beschreibung der Objekte	130
6.13.2.1	In diesem Kapitel behandelte Objekte	130
6.13.2.2	Objekt 60FD _h : digital_inputs.....	131
6.13.2.3	Objekt 60FE _h : digital_outputs.....	132
6.13.2.4	Objekt 2420 _h : digital_output_state_mapping.....	133
6.14	Endschalter / Referenzschalter.....	135
6.14.1	Übersicht.....	135
6.14.2	Beschreibung der Objekte	135
6.14.2.1	Objekt 6510 _h _11 _h : limit_switch_polarity.....	135
6.14.2.2	Objekt 6510 _h _12 _h : limit_switch_selector.....	136
6.14.2.3	Objekt 6510 _h _14 _h : homing_switch_polarity	136
6.14.2.4	Objekt 6510 _h _13 _h : homing_switch_selector	137
6.14.2.5	Objekt 6510 _h _15 _h : limit_switch_deceleration.....	137
6.15	Sampling von Positionen	138
6.15.1	Übersicht.....	138
6.15.2	Beschreibung der Objekte	138
6.15.2.1	In diesem Kapitel behandelte Objekte	138
6.15.2.2	Objekt 204A _h : sample_data	139
6.16	Bremsen-Ansteuerung	142
6.16.1	Übersicht.....	142
6.16.2	Beschreibung der Objekte	143
6.16.2.1	Objekt 6510 _h _18 _h : brake_delay_time	143
6.17	Geräteinformationen	144
6.17.1	Beschreibung der Objekte	144
6.17.1.1	Objekt 1018 _h : identity_object	144
6.17.1.2	Objekt 6510 _h _A0 _h : drive_serial_number.....	146
6.17.1.3	Objekt 6510 _h _A1 _h : drive_type	146
6.17.1.4	Objekt 6510 _h _A9 _h : firmware_main_version	147
6.17.1.5	Objekt 6510 _h _AA _h : firmware_custom_version.....	147
6.17.1.6	Objekt 6510 _h _AD _h : km_release	147
6.17.1.7	Objekt 6510 _h _AC _h : firmware_type.....	148
6.17.1.8	Objekt 6510 _h _B0 _h : cycletime_current_controller	148
6.17.1.9	Objekt 6510 _h _B1 _h : cycletime_velocity_controller.....	149
6.17.1.10	Objekt 6510 _h _B2 _h : cycletime_position_controller	149
6.17.1.11	Objekt 6510 _h _B3 _h : cycletime_trajectory_generator	149
6.17.1.12	Objekt 6510 _h _C0 _h : commissioning_state	150
6.18	Fehlermanagement.....	151
6.18.1	Übersicht.....	151
6.18.2	Beschreibung der Objekte	151

6.18.2.1	In diesem Kapitel behandelte Objekte	151
6.18.2.2	Objekt 2100 _H : error_management.....	152
6.18.2.3	Objekt 200F _H : last_warning_code	153
7	Gerätesteuerung (Device Control).....	154
7.1	Zustandsdiagramm (State Machine).....	154
7.1.1	Übersicht.....	154
7.1.2	Das Zustandsdiagramm des Reglers (State Machine)	155
7.1.2.1	Zustandsdiagramm: Zustände	157
7.1.2.2	Zustandsdiagramm: Zustandsübergänge	157
7.1.3	controlword (Steuerwort)	159
7.1.3.1	Objekt 6040 _H : controlword.....	159
7.1.4	Auslesen des Reglerzustands	163
7.1.5	statuswords (Statusworte)	164
7.1.5.1	Objekt 6041 _H : statusword	164
7.1.5.2	Objekt 2000 _H : manufacturer_statuswords	168
7.1.5.3	Objekt 2005 _H : manufacturer_status_masks	171
7.1.5.4	Objekt 200A _H : manufacturer_status_invert.....	172
7.1.6	Beschreibung der weiteren Objekte	173
7.1.6.1	In diesem Kapitel behandelte Objekte	173
7.1.6.2	Objekt 605B _H : shutdown_option_code	173
7.1.6.3	Objekt 605C _H : disable_operation_option_code	174
7.1.6.4	Objekt 605A _H : quick_stop_option_code	174
7.1.6.5	Objekt 605E _H : fault_reaction_option_code	175
8	Betriebsarten	176
8.1	Einstellen der Betriebsart.....	176
8.1.1	Übersicht.....	176
8.1.2	Beschreibung der Objekte	176
8.1.2.1	In diesem Kapitel behandelte Objekte	176
8.1.2.2	Objekt 6060 _H : modes_of_operation.....	177
8.1.2.3	Objekt 6061 _H : modes_of_operation_display.....	178
8.2	Betriebsart Referenzfahrt (Homing Mode).....	179
8.2.1	Übersicht.....	179
8.2.2	Beschreibung der Objekte	180
8.2.2.1	In diesem Kapitel behandelte Objekte	180
8.2.2.2	Betroffene Objekte aus anderen Abschnitten	180
8.2.2.3	Objekt 607C _H : home_offset.....	180
8.2.2.4	Objekt 6098 _H : homing_method.....	181
8.2.2.5	Objekt 6099 _H : homing_speeds	182
8.2.2.6	Objekt 609A _H : homing_acceleration.....	183
8.2.2.7	Objekt 2045 _H : homing_timeout.....	183
8.2.3	Referenzfahrt-Abläufe.....	184
8.2.3.1	Methode 1: Negativer Endschalter mit Nullimpulsauswertung	184
8.2.3.2	Methode 2: Positiver Endschalter mit Nullimpulsauswertung	184
8.2.3.3	Methoden 7 u. 11: Referenzschalter und Nullimpulsauswertung	185
8.2.3.4	Methode 17: Referenzfahrt auf den negativen Endschalter.....	186
8.2.3.5	Methode 18: Referenzfahrt auf den positiven Endschalter	186
8.2.3.6	Methoden 23 und 27: Referenzfahrt auf den Referenzschalter	187
8.2.3.7	Methode -1: negativer Anschlag mit Nullimpulsauswertung.....	188
8.2.3.8	Methode -2: positiver Anschlag mit Nullimpulsauswertung	188
8.2.3.9	Methode -17: Referenzfahrt auf den negativen Anschlag	189

8.2.3.10	Methode –18: Referenzfahrt auf den positiven Anschlag	189
8.2.3.11	Methoden 32 und 33: Referenzfahrt auf den Nullimpuls.....	189
8.2.3.12	Methode 34: Referenzfahrt auf die aktuelle Position	190
8.2.4	Steuerung der Referenzfahrt	190
8.3	Betriebsart Positionieren (Profile Position Mode)	191
8.3.1	Übersicht.....	191
8.3.2	Beschreibung der Objekte	192
8.3.2.1	In diesem Kapitel behandelte Objekte	192
8.3.2.2	Betroffene Objekte aus anderen Abschnitten	192
8.3.2.3	Objekt 607A _h : target_position	192
8.3.2.4	Objekt 6081 _h : profile_velocity.....	193
8.3.2.5	Objekt 6082 _h : end_velocity	193
8.3.2.6	Objekt 6083 _h : profile_acceleration	194
8.3.2.7	Objekt 6084 _h : profile_deceleration.....	194
8.3.2.8	Objekt 6085 _h : quick_stop_deceleration.....	195
8.3.2.9	Objekt 6086 _h : motion_profile_type	195
8.3.3	Funktionsbeschreibung	196
8.4	Interpolated Position Mode	198
8.4.1	Übersicht.....	198
8.4.2	Beschreibung der Objekte	199
8.4.2.1	In diesem Kapitel behandelte Objekte	199
8.4.2.2	Betroffene Objekte aus anderen Kapiteln.....	199
8.4.2.3	Objekt 60C0 _h : interpolation_submode_select	199
8.4.2.4	Objekt 60C1 _h : interpolation_data_record	200
8.4.2.5	Objekt 60C2 _h : interpolation_time_period.....	201
8.4.2.6	Objekt 60C3 _h : interpolation_sync_definition.....	202
8.4.2.7	Objekt 60C4 _h : interpolation_data_configuration.....	203
8.4.3	Funktionsbeschreibung	205
8.4.3.1	Vorbereitende Parametrierung	205
8.4.3.2	Aktivierung des Interpolated Position Mode und Aufsynchronisation.....	205
8.4.3.3	Unterbrechung der Interpolation im Fehlerfall.....	207
8.5	Betriebsart Drehzahlregelung (Profile Velocity Mode).....	208
8.5.1	Übersicht.....	208
8.5.2	Beschreibung der Objekte	210
8.5.2.1	In diesem Kapitel behandelte Objekte	210
8.5.2.2	Betroffene Objekte aus anderen Abschnitten	210
8.5.2.3	Objekt 6069 _h : velocity_sensor_actual_value.....	211
8.5.2.4	Objekt 606A _h : sensor_selection_code	211
8.5.2.5	Objekt 606B _h : velocity_demand_value.....	212
8.5.2.6	Objekt 202E _h : velocity_demand_sync_value	212
8.5.2.7	Objekt 606C _h : velocity_actual_value.....	213
8.5.2.8	Objekt 2074 _h : velocity_actual_value_filtered.....	214
8.5.2.9	Objekt 606D _h : velocity_window	215
8.5.2.10	Objekt 606E _h : velocity_window_time	215
8.5.2.11	Objekt 606F _h : velocity_threshold	216
8.5.2.12	Objekt 6070 _h : velocity_threshold_time	216
8.5.2.13	Objekt 6080 _h : max_motor_speed.....	217
8.5.2.14	Objekt 60FF _h : target_velocity.....	217
8.6	Drehzahl-Rampen	218
8.7	Betriebsart Momentenregelung (Profile Torque Mode)	220
8.7.1	Übersicht.....	220

8.7.2	Beschreibung der Objekte	221
8.7.2.1	In diesem Kapitel behandelte Objekte	221
8.7.2.2	Betroffene Objekte aus anderen Kapiteln	221
8.7.2.3	Objekt 6071 _h : target_torque	222
8.7.2.4	Objekt 6072 _h : max_torque	222
8.7.2.5	Objekt 6074 _h : torque_demand_value	223
8.7.2.6	Objekt 6076 _h : motorRatedTorque	223
8.7.2.7	Objekt 6077 _h : torqueActualValue	224
8.7.2.8	Objekt 6078 _h : currentActualValue	224
8.7.2.9	Objekt 6079 _h : dcLinkCircuitVoltage	225
8.7.2.10	Objekt 6087 _h : torqueSlope	225
8.7.2.11	Objekt 6088 _h : torqueProfileType	226
9	Anhang	227
9.1	Kenndaten des CAN-Interface	227
9.2	Definitionsdatei	227
10	Stichwortverzeichnis	234

Abbildungsverzeichnis

Abbildung 3.1: CAN-Steckverbinder für SERVOTEC S2	26
Abbildung 3.2: Verkabelungsbeispiel	27
Abbildung 4.1: CAN-Bus-Fenster S2 Commander	28
Abbildung 5.2: Zugriffsverfahren	30
Abbildung 5.3: Identifier	31
Abbildung 5.4: SDO-Sequenzen zum Lesen und Schreiben	32
Abbildung 5.5: SDO-Fehlermeldungen	33
Abbildung 5.6: Simulation von SDO-Zugriffen über RS232	34
Abbildung 5.7: Identifier SYNC-Message	45
Abbildung 5.8: Emergency-Message	46
Abbildung 5.9: Aufbau der EMERGENCY-Message	47
Abbildung 5.10: Aufbau der NMT-Nachricht	52
Abbildung 5.11: NMT-State Machine	52
Abbildung 5.12: Aufbau der Bootup-Nachricht	54
Abbildung 5.13: Aufbau der Heartbeat-Nachricht	55
Abbildung 5.14: Aufbau der Nodeguarding-Nachrichten	57
Abbildung 5.15: Aufbau der Nodeguarding-Nachrichten (Antwort Regler)	57
Abbildung 6.16: Default-/Applikations-Parametersatz	62
Abbildung 6.17: Umrechnungsfaktoren (Factor Group)	68
Abbildung 6.18: Übersicht: Factor Group	69
Abbildung 6.19: Schleppfehler – Funktionsübersicht	98
Abbildung 6.20: Schleppfehler	99
Abbildung 6.21: Position erreicht – Funktionsübersicht	99
Abbildung 6.22: Position erreicht	100
Abbildung 6.23: Funktion der Bremsverzögerung (bei Drehzahlregelung / Positionieren)	142
Abbildung 7.24: Zustandsdiagramm des Reglers	155
Abbildung 7.25: Wichtigste Zustandsübergänge des Reglers	156
Abbildung 7.26: Manufacturer Statuswords	170
Abbildung 8.1: Die Referenzfahrt	179
Abbildung 8.2: Home Offset	180
Abbildung 8.3: Referenzfahrt auf den negativen Endschalter mit Auswertung des Nullimpulses	184
Abbildung 8.4: Referenzfahrt auf den positiven Endschalter mit Auswertung des Nullimpulses	184
Abbildung 8.5: Referenzfahrt auf den Referenzschalter mit Auswertung des Nullimpulses bei positiver Anfangsbewegung	185
Abbildung 8.6: Referenzfahrt auf den Referenzschalter mit Auswertung des Nullimpulses bei negativer Anfangsbewegung	185
Abbildung 8.7: Referenzfahrt auf den negativen Endschalter	186
Abbildung 8.8: Referenzfahrt auf den positiven Endschalter	186
Abbildung 8.9: Referenzfahrt auf den Referenzschalter bei positiver Anfangsbewegung	187
Abbildung 8.10: Referenzfahrt auf den Referenzschalter bei negativer Anfangsbewegung	187
Abbildung 8.11: Referenzfahrt auf den negativen Anschlag mit Auswertung des Nullimpulses	188
Abbildung 8.12: Referenzfahrt auf den positiven Anschlag mit Auswertung des Nullimpulses	188
Abbildung 8.13: Referenzfahrt auf den negativen Anschlag	189
Abbildung 8.14: Referenzfahrt auf den positiven Anschlag	189
Abbildung 8.15: Referenzfahrt nur auf den Nullimpuls bezogen	189
Abbildung 8.16: Fahrkurven-Generator und Lageregler	191

Abbildung 8.17: Fahrauftrag-Übertragung von einem Host	196
Abbildung 8.18: Einfacher Fahrauftrag	197
Abbildung 8.19: Lückenlose Folge von Fahraufträgen	197
Abbildung 8.20: Fahrauftrag Lineare Interpolation zwischen zwei Datenwerten	198
Abbildung 8.21: Aufsynchronisation und Datenfreigabe	206
Abbildung 8.22: Struktur des drehzahlgeregelten Betriebs (Profile Velocity Mode)	209
Abbildung 8.23: Ermittlung von velocity_actual_value und velocity_actual_value_filtered	214
Abbildung 8.24: Drehzahlrampen	218
Abbildung 8.25: Struktur des drehmomentgeregelten Betriebs	220

1 Allgemeines

1.1 Dokumentation

Das vorliegende Handbuch beschreibt, wie die Servopositionierregler der servoTEC S2 in eine CANopen-Netzwerkumgebung einbezogen werden können. Es wird die Einstellung der physikalischen Parameter, die Aktivierung des CANopen-Protokolls, die Einbindung in das CAN-Netzwerk und die Kommunikation mit dem Servopositionierregler beschrieben. Es richtet sich an Personen, die bereits mit dieser Servopositionierregler-Reihe vertraut sind.

Das Handbuch enthält Sicherheitshinweise, die beachtet werden müssen.

Weitergehende Informationen finden sich in folgenden Handbüchern zur servoTEC S2 Produktfamilie:

- **Softwarehandbuch “MAN_DE_1078649_LV-servoTEC_S2_Softwarehandbuch”:** Beschreibung der Gerätefunktionalität und der Softwarefunktionen der Firmware einschließlich der RS232-Kommunikation. Beschreibung des Parametrierprogramms IEF Werner GmbH ServoCommander™ mit einer Anleitung bei der Erstinbetriebnahme eines Servopositionierreglers servoTEC S2.
- **Produkthandbuch “MAN_DE_1076777_LV-servoTEC_S2_3xx”:** Beschreibung der Technischen Daten und der Gerätefunktionalität sowie Hinweise zur Installation und Betrieb des Servopositionierreglers servoTEC S2 Typ 302 und 305.
- **Produkthandbuch “MAN_DE_1076775_LV-servoTEC_S2_1xx”:** Beschreibung der Technischen Daten und der Gerätefunktionalität sowie Hinweise zur Installation und Betrieb des Servopositionierreglers servoTEC S2 Typ 102 und 105.

1.2 CANopen

CANopen ist ein von der Vereinigung „CAN in Automation“ erarbeiteter Standard. In diesem Verbund sind eine Vielzahl von Geräteherstellern organisiert. Dieser Standard hat die bisherigen, herstellerspezifischen CAN-Protokolle weitgehend ersetzt. Somit steht dem Endanwender ein herstellerunabhängiges Kommunikations-Interface zur Verfügung.

Von diesem Verbund sind unter anderem folgende Handbücher beziehbar:

- **CiA Draft Standard 201-207:**
In diesen Werken werden die allgemeinen Grundlagen und die Einbettung von CANopen in das OSI-Schichtenmodell behandelt. Die relevanten Punkte dieses Buches werden im vorliegenden CANopen-Handbuch vorgestellt, so dass der Erwerb der DS201..207 im allgemeinen nicht notwendig ist.
- **CiA Draft Standard 301:**
In diesem Werk wird der grundsätzliche Aufbau des Objektverzeichnisses eines CANopen-Gerätes und der Zugriff auf dieses beschrieben. Außerdem werden die Aussagen der DS201..207 konkretisiert. Die für die Reglerfamilien servoTEC S2 benötigten Elemente des Objektverzeichnisses und die zugehörigen Zugriffsmethoden sind im vorliegenden Handbuch beschrieben. Der Erwerb der DS301 ist ratsam aber nicht unbedingt notwendig.
- **CiA Draft Standard 402:**
Dieses Buch befasst sich mit der konkreten Implementation von CANopen in Antriebsregler. Obwohl alle implementierten Objekte auch im vorliegenden CANopen-Handbuch in kurzer Form dokumentiert und beschrieben sind, sollte der Anwender über dieses Werk verfügen.

Bezugsadresse:

CAN in Automation (CiA) International Headquarter

Am Weichselgarten 26

D-91058 Erlangen

Telefon: 09131-601091

Telefax: 09131-601092

Web-Adresse: www.can-cia.de

Der CANopen-Implementierung des Reglers liegen folgende Normen zugrunde:

- CiA Draft Standard 301, Version 4.02, 13. Februar 2002
- CiA Draft Standard Proposal 402, Version 2.0, 26. Juli 2002

2 Sicherheitshinweise für elektrische Antriebe und Steuerungen

2.1 Verwendete Symbole



Information

Wichtige Informationen und Hinweise.



Vorsicht

Die Nichtbeachtung kann hohe Sachschäden zur Folge haben.



GEFAHR!

Die Nichtbeachtung kann **Sachschäden** und **Personenschäden** zur Folge haben.



Vorsicht! Lebensgefährliche Spannung.

Der Sicherheitshinweis enthält einen Hinweis auf eine eventuell auftretende lebensgefährliche Spannung.



Die mit diesem Symbol gekennzeichneten Abschnitte stellen Beispiele dar, die das Verständnis und die Anwendung einzelner Objekte und Parameter erleichtern.

2.2 Allgemeine Hinweise

Bei Schäden infolge von Nichtbeachtung der Warnhinweise in dieser Betriebsanleitung übernimmt die IEF Werner GmbH keine Haftung.



Vor der Inbetriebnahme sind die *Sicherheitshinweise für elektrische Antriebe und Steuerungen ab Seite 14* durchzulesen.

Wenn die Dokumentation in der vorliegenden Sprache nicht einwandfrei verstanden wird, bitte beim Lieferant anfragen und diesen informieren.

Der einwandfreie und sichere Betrieb des Servoantriebsreglers setzt den sachgemäßen und fachgerechten Transport, die Lagerung, die Montage und die Installation sowie die sorgfältige Bedienung und die Instandhaltung voraus. Für den Umgang mit elektrischen Anlagen ist ausschließlich ausgebildetes und qualifiziertes Personal einsetzen:

Ausgebildetes und qualifiziertes Personal

Ausgebildetes und qualifiziertes Personal im Sinne dieses Produkthandbuches bzw. der Warnhinweise auf dem Produkt selbst sind Personen, die mit der Aufstellung, der Montage, der Inbetriebsetzung und dem Betrieb des Produktes sowie mit allen Warnungen und Vorsichtsmaßnahmen gemäß dieser Betriebsanleitung in diesem Produkthandbuch ausreichend vertraut sind und über die ihrer Tätigkeit entsprechenden Qualifikationen verfügen:

- Ausbildung und Unterweisung bzw. Berechtigung, Geräte/Systeme gemäß den Standards der Sicherheitstechnik ein- und auszuschalten, zu erden und gemäß den Arbeitsanforderungen zweckmäßig zu kennzeichnen.
- Ausbildung oder Unterweisung gemäß den Standards der Sicherheitstechnik in Pflege und Gebrauch angemessener Sicherheitsausrüstung.
- Schulung in Erster Hilfe.

2.3 Hinweise vor der ersten Inbetriebnahme

Die nachfolgenden Hinweise sind vor der ersten Inbetriebnahme der Anlage zur Vermeidung von Körperverletzungen und/oder Sachschäden zu lesen:



Diese Sicherheitshinweise sind jederzeit einzuhalten.



Versuchen Sie nicht, den Servoantriebsregler zu installieren oder in Betrieb zu nehmen, bevor Sie nicht alle Sicherheitshinweise für elektrische Antriebe und Steuerungen in diesem Dokument sorgfältig durchgelesen haben. Diese Sicherheitsinstruktionen und alle anderen Benutzerhinweise sind vor jeder Arbeit mit dem Servoantriebsregler durchzulesen.



Sollten Ihnen keine Benutzerhinweise für den Servoantriebsregler zur Verfügung stehen, wenden Sie sich an Ihren zuständigen Vertriebsrepräsentanten. Verlangen Sie die unverzügliche Übersendung dieser Unterlagen an den oder die Verantwortlichen für den sicheren Betrieb des Servoantriebsreglers.



Bei Verkauf, Verleih und/oder anderweitiger Weitergabe des Servoantriebsreglers sind diese Sicherheitshinweise ebenfalls mitzugeben.



Ein Öffnen des Servoantriebsreglers durch den Betreiber ist aus Sicherheits- und Gewährleistungsgründen nicht zulässig.



Die Voraussetzung für eine einwandfreie Funktion des Servoantriebsreglers ist eine fachgerechte Projektierung!



GEFAHR!

Unsachgemäßer Umgang mit dem Servoantriebsregler und Nichtbeachten der hier angegebenen Warnhinweise sowie unsachgemäße Eingriffe in die Sicherheitseinrichtung können zu Sachschaden, Körperverletzung, elektrischem Schlag oder im Extremfall zum Tod führen.

2.4 Gefahren durch falschen Gebrauch



GEFAHR!

Hohe elektrische Spannung und hoher Arbeitsstrom!

Lebensgefahr oder schwere Körperverletzung durch elektrischen Schlag!



GEFAHR!

Hohe elektrische Spannung durch falschen Anschluss!

Lebensgefahr oder Körperverletzung durch elektrischen Schlag!



GEFAHR!

Heiße Oberflächen des Gerätegehäuses möglich!

Verletzungsgefahr! Verbrennungsgefahr!



GEFAHR!

Gefahrbringende Bewegungen!

Lebensgefahr, schwere Körperverletzung oder Sachschaden durch unbeabsichtigte Bewegungen der Motoren!

2.5 Sicherheitshinweise

2.5.1 Allgemeine Sicherheitshinweise



Der Servoantriebsregler entspricht der Schutzklasse IP20, sowie der Verschmutzungsstufe 1. Es ist darauf zu achten, dass die Umgebung dieser Schutz- bzw. Verschmutzungsstufe entspricht.



Nur vom Hersteller zugelassene Zubehör- und Ersatzteile verwenden.



Die Servoantriebsregler müssen entsprechend den EN-Normen und VDE-Vorschriften so an das Netz angeschlossen werden, dass sie mit geeigneten Freischaltmitteln (z.B. Hauptschalter, Schütz, Leistungsschalter) vom Netz getrennt werden können.



Der Servoantriebsregler kann mit einem allstromsensitiven FI-Schutzschalter (RCD = Residual Current protective Device) 300mA abgesichert werden.



Zum Schalten der Steuerkontakte sollten vergoldete Kontakte oder Kontakte mit hohem Kontaktdruck verwendet werden.



Vorsorglich müssen Entstörungsmaßnahmen für Schaltanlagen getroffen werden, wie z.B. Schütze und Relais mit RC-Gliedern bzw. Dioden beschalten.



Es sind die Sicherheitsvorschriften und -bestimmungen des Landes, in dem das Gerät zur Anwendung kommt, zu beachten.



Die in der Produktdokumentation angegebenen Umgebungsbedingungen müssen eingehalten werden. Sicherheitskritische Anwendungen sind nicht zugelassen, sofern sie nicht ausdrücklich vom Hersteller freigegeben werden.



Die Hinweise für eine EMV-gerechte Installation sind aus dem Produkthandbuch Servopositionierregler servoTEC S2 zu entnehmen. Die Einhaltung der durch die nationalen Vorschriften geforderten Grenzwerte liegt in der Verantwortung der Hersteller der Anlage oder Maschine.



Die technischen Daten, die Anschluss- und Installationsbedingungen für den Servoantriebsregler sind aus diesem Produkthandbuch zu entnehmen und unbedingt einzuhalten.



GEFAHR!

Es sind die Allgemeinen Errichtungs- und Sicherheitsvorschriften für das Arbeiten an Starkstromanlagen (z.B. DIN, VDE, EN, IEC oder andere nationale und internationale Vorschriften) zu beachten.

Nichtbeachtung können Tod, Körperverletzung oder erheblichen Sachschaden zur Folge haben.



Ohne Anspruch auf Vollständigkeit gelten unter anderem folgende Vorschriften:

VDE 0100: Bestimmung für das Errichten von Starkstromanlagen bis 1000 Volt

EN 60204: Elektrische Ausrüstung von Maschinen

EN 50178: Ausrüstung von Starkstromanlagen mit elektronischen Betriebsmitteln

2.5.2 Sicherheitshinweise bei Montage und Wartung

Für die Montage und Wartung der Anlage gelten in jedem Fall die einschlägigen DIN, VDE, EN und IEC-Vorschriften, sowie alle staatlichen und örtlichen Sicherheits- und Unfallverhütungsvorschriften. Der Anlagenbauer bzw. der Betreiber hat für die Einhaltung dieser Vorschriften zu sorgen:



Die Bedienung, Wartung und/oder Instandsetzung des Servoantriebsreglers darf nur durch für die Arbeit an oder mit elektrischen Geräten ausgebildetes und qualifiziertes Personal erfolgen.

Vermeidung von Unfällen, Körperverletzung und/oder Sachschaden:



Vertikale Achsen gegen Herabfallen oder Absinken nach Abschalten des Motors zusätzlich sichern, wie durch:

- mechanische Verriegelung der vertikalen Achse
- externe Brems-/ Fang-/ Klemmeinrichtung oder
- ausreichenden Gewichtsausgleich der Achse.



Die serienmäßig gelieferte Motor-Haltebremse oder eine externe, vom Antriebsregelgerät angesteuerte Motor-Haltebremse alleine ist nicht für den Personenschutz geeignet!



Die elektrische Ausrüstung über den Hauptschalter spannungsfrei schalten und gegen Wiedereinschalten sichern, warten bis der Zwischenkreis entladen ist, bei:

- Wartungsarbeiten und Instandsetzung
- Reinigungsarbeiten
- langen Betriebsunterbrechungen



Vor der Durchführung von Wartungsarbeiten ist sicherzustellen, dass die Stromversorgung abgeschaltet, verriegelt und der Zwischenkreis entladen ist.



Der externe oder interne Bremswiderstand führt im Betrieb und kann bis ca. 5 Minuten nach dem Abschalten des Servoantriebsreglers gefährliche Zwischenkreisspannung führen, diese kann bei Berührung den Tod oder schwere Körperverletzungen hervorrufen.

Fortsetzung: Vermeidung von Unfällen, Körperverletzung und/oder Sachschaden:



Bei der Montage ist sorgfältig vorzugehen. Es ist sicherzustellen, dass sowohl bei Montage als auch während des späteren Betriebes des Antriebs keine Bohrspäne, Metallstaub oder Montageteile (Schrauben, Muttern, Leitungsabschnitte) in den Servoantriebsregler fallen.



Ebenfalls ist sicherzustellen, dass die externe Spannungsversorgung des Reglers (24V) abgeschaltet ist.



Ein Abschalten des Zwischenkreises oder der Netzspannung muss immer vor dem Abschalten der 24V Reglerversorgung erfolgen.



Die Arbeiten im Maschinenbereich sind nur bei abgeschalteter und verriegelter Wechselstrom- bzw. Gleichstromversorgung durchzuführen. Abgeschaltete Endstufen oder abgeschaltete Reglerfreigabe sind keine geeigneten Verriegelungen. Hier kann es im Störfall zum unbeabsichtigten Verfahren des Antriebes kommen.



Die Inbetriebnahme mit leerlaufenden Motoren durchführen, um mechanische Beschädigungen, z.B. durch falsche Drehrichtung zu vermeiden.



Elektronische Geräte sind grundsätzlich nicht ausfallsicher. Der Anwender ist dafür verantwortlich, dass bei Ausfall des elektrischen Geräts seine Anlage in einen sicheren Zustand geführt wird.



Der Servoantriebsregler und insbesondere der Bremswiderstand, extern oder intern, können hohe Temperaturen annehmen, die bei Berührung schwere körperliche Verbrennungen verursachen können.

2.5.3 Schutz gegen Berühren elektrischer Teile

Dieser Abschnitt betrifft nur Geräte und Antriebskomponenten mit Spannungen über 50 Volt. Werden Teile mit Spannungen größer 50 Volt berührt, können diese für Personen gefährlich werden und zu elektrischem Schlag führen. Beim Betrieb elektrischer Geräte stehen zwangsläufig bestimmte Teile dieser Geräte unter gefährlicher Spannung.



GEFAHR!

Hohe elektrische Spannung!

Lebensgefahr, Verletzungsgefahr durch elektrischen Schlag oder schwere Körperverletzung!

Für den Betrieb gelten in jedem Fall die einschlägigen DIN, VDE, EN und IEC-Vorschriften, sowie alle staatlichen und örtlichen Sicherheits- und Unfallverhütungsvorschriften. Der Anlagenbauer bzw. der Betreiber hat für die Einhaltung dieser Vorschriften zu sorgen:



Vor dem Einschalten die dafür vorgesehenen Abdeckungen und Schutzvorrichtungen für den Berührungsschutz an den Geräten anbringen. Für Einbaugeräte ist der Schutz gegen direktes Berühren elektrischer Teile durch ein äußeres Gehäuse, wie beispielsweise einen Schaltschrank, sicherzustellen. Die Vorschriften VBG 4 sind zu beachten!



Den Schutzleiter der elektrischen Ausrüstung und der Geräte stets fest an das Versorgungsnetz anschließen. Der Ableitstrom ist aufgrund der integrierten Netzfilter größer als 3,5 mA!



Nach der Norm EN60617 den vorgeschriebenen Mindest-Kupfer-Querschnitt für die Schutzleiterverbindung in seinem ganzen Verlauf beachten!



Vor Inbetriebnahme, auch für kurzzeitige Mess- und Prüfzwecke, stets den Schutzleiter an allen elektrischen Geräten entsprechend dem Anschlussplan anschließen oder mit Erdleiter verbinden. Auf dem Gehäuse können sonst hohe Spannungen auftreten, die einen elektrischen Schlag verursachen.



Elektrische Anschlussstellen der Komponenten im eingeschalteten Zustand nicht berühren.



Vor dem Zugriff zu elektrischen Teilen mit Spannungen größer 50 Volt das Gerät vom Netz oder von der Spannungsquelle trennen. Gegen Wiedereinschalten sichern.



Bei der Installation ist besonders in Bezug auf Isolation und Schutzmaßnahmen die Höhe der Zwischenkreisspannung zu berücksichtigen. Es muss für ordnungsgemäße Erdung, Leiterdimensionierung und entsprechenden Kurzschlusschutz gesorgt werden.



Das Gerät verfügt über eine Zwischenkreisschnellentladeschaltung gemäß EN60204 Abschnitt 6.2.4. In bestimmten Gerätekonstellationen, vor allem bei der Parallelschaltung mehrerer Servoantriebsregler im Zwischenkreis oder bei einem nicht angeschlossenen Bremswiderstand, kann die Schnellentladung allerdings unwirksam sein. Die Servoantriebsregler können dann nach dem Abschalten bis zu 5 Minuten unter gefährlicher Spannung stehen (Kondensatorrestladung).

2.5.4 Schutz durch Schutzkleinspannung (PELV) gegen elektrischen Schlag

Alle Anschlüsse und Klemmen mit Spannungen von 5 bis 50 Volt an dem Servoantriebsregler sind Schutzkleinspannungen, die entsprechend folgender Normen berührungssicher ausgeführt sind:

- international: IEC 60364-4-41
- Europäische Länder in der EU: EN 50178/1998, Abschnitt 5.2.8.1.



GEFAHR!

Hohe elektrische Spannung durch falschen Anschluss!

Lebensgefahr, Verletzungsgefahr durch elektrischen Schlag!

An alle Anschlüsse und Klemmen mit Spannungen von 0 bis 50 Volt dürfen nur Geräte, elektrische Komponenten und Leitungen angeschlossen werden, die eine Schutzkleinspannung (PELV = Protective Extra Low Voltage) aufweisen.

Nur Spannungen und Stromkreise, die sichere Trennung zu gefährlichen Spannungen haben, anschließen. Sichere Trennung wird beispielsweise durch Trenntransformatoren, sichere Optokoppler oder netzfreien Batteriebetrieb erreicht.

2.5.5 Schutz vor gefährlichen Bewegungen

Gefährliche Bewegungen können durch fehlerhafte Ansteuerung von angeschlossenen Motoren verursacht werden. Die Ursachen können verschiedenster Art sein:

- unsaubere oder fehlerhafte Verdrahtung oder Verkabelung
- Fehler bei der Bedienung der Komponenten
- Fehler in den Messwert- und Signalgebern
- defekte oder nicht EMV-gerechte Komponenten
- Fehler in der Software im übergeordneten Steuerungssystem

Diese Fehler können unmittelbar nach dem Einschalten oder nach einer unbestimmten Zeitdauer im Betrieb auftreten.

Die Überwachungen in den Antriebskomponenten schließen eine Fehlfunktion in den angeschlossenen Antrieben weitestgehend aus. Im Hinblick auf den Personenschutz, insbesondere der Gefahr der Körperverletzung und/oder Sachschaden, darf auf diesen Sachverhalt nicht allein vertraut werden. Bis zum Wirksamwerden der eingebauten Überwachungen ist auf jeden Fall mit einer fehlerhaften Antriebsbewegung zu rechnen, deren Maß von der Art der Steuerung und des Betriebszustandes abhängen.



GEFAHR!

Gefahrbringende Bewegungen!

Lebensgefahr, Verletzungsgefahr, schwere Körperverletzung oder Sachschaden!

Der Personenschutz ist aus den oben genannten Gründen durch Überwachungen oder Maßnahmen, die anlagenseitig übergeordnet sind, sicherzustellen. Diese werden nach den spezifischen Gegebenheiten der Anlage einer Gefahren- und Fehleranalyse vom Anlagenbauer vorgesehen. Die für die Anlage geltenden Sicherheitsbestimmungen werden hierbei mit einbezogen. Durch Ausschalten, Umgehen oder fehlendes Aktivieren von Sicherheitseinrichtungen können willkürliche Bewegungen der Maschine oder andere Fehlfunktionen auftreten.

2.5.6 Schutz gegen Berühren heißer Teile



GEFAHR!

Heiße Oberflächen des Gerätegehäuses möglich!

Verletzungsgefahr! Verbrennungsgefahr!



Gehäuseoberfläche in der Nähe von heißen Wärmequellen nicht berühren!
Verbrennungsgefahr!



Vor dem Zugriff Geräte nach dem Abschalten erst 10 Minuten abkühlen lassen.



Werden heiße Teile der Ausrüstung wie Gerätegehäuse, in denen sich Kühlkörper und Widerstände befinden, berührt, kann das zu Verbrennungen führen!

2.5.7 Schutz bei Handhabung und Montage

Die Handhabung und Montage bestimmter Teile und Komponenten in ungeeigneter Art und Weise kann unter ungünstigen Bedingungen zu Verletzungen führen.



GEFAHR!

Verletzungsgefahr durch unsachgemäße Handhabung!

Körperverletzung durch Quetschen, Scheren, Schneiden, Stoßen!

Hierfür gelten allgemeine Sicherhinweise:



Die allgemeinen Errichtungs- und Sicherheitsvorschriften zu Handhabung und Montage beachten.



Geeignete Montage- und Transporteinrichtungen verwenden.



Einklemmungen und Quetschungen durch geeignete Vorkehrungen vorbeugen.



Nur geeignetes Werkzeug verwenden. Sofern vorgeschrieben, Spezialwerkzeug benutzen.



Hebeeinrichtungen und Werkzeuge fachgerecht einsetzen.



Wenn erforderlich, geeignete Schutzausstattungen (zum Beispiel Schutzbrillen, Sicherheitsschuhe, Schutzhandschuhe) benutzen.



Nicht unter hängenden Lasten aufhalten.



Auslaufende Flüssigkeiten am Boden sofort wegen Rutschgefahr beseitigen.

3 Verkabelung und Steckerbelegung

3.1 Anschlussbelegungen

Das CAN-Interface ist bei der Gerätefamilie SERVOTEC S2 bereits im Servoregler integriert und somit immer verfügbar.

Der CAN-Bus-Anschluss ist normgemäß als 9-poliger DSUB-Stecker (reglerseitig) ausgeführt.

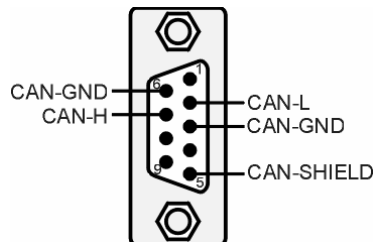


Abbildung 3.1: CAN-Steckverbinder für SERVOTEC S2



CAN-Bus-Verkabelung

Bei der Verkabelung der Regler über den CAN-Bus sollten sie unbedingt die nachfolgenden Informationen und Hinweise beachten, um ein stabiles, störungsfreies System zu erhalten. Bei einer nicht sachgemäßen Verkabelung können während des Betriebs Störungen auf dem CAN-Bus auftreten, die dazu führen, dass der Regler aus Sicherheitsgründen mit einem Fehler abschaltet.



120Ω-Abschlusswiderstand

In den Geräten der SERVOTEC S2-Reihe ist kein Abschlusswiderstand integriert.

3.2 Verkabelungs-Hinweise

Der CAN-Bus bietet eine einfache und störungssichere Möglichkeit alle Komponenten einer Anlage miteinander zu vernetzen. Voraussetzung dafür ist allerdings, dass alle nachfolgenden Hinweise für die Verkabelung beachtet werden.

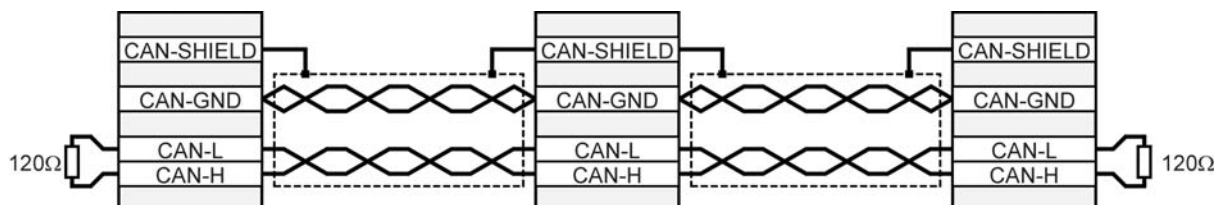


Abbildung 3.2: Verkabelungsbeispiel

- Die einzelnen Knoten des Netzwerkes werden grundsätzlich linienförmig miteinander verbunden, so dass das CAN-Kabel von Regler zu Regler durchgeschleift wird (siehe *Abbildung 3.2*).
- An beiden Enden des CAN-Kabels muss jeweils genau ein Abschlusswiderstand von $120\Omega \pm 5\%$ vorhanden sein. Häufig ist in CAN-Karten oder in einer SPS bereits ein solcher Abschlusswiderstand eingebaut, der entsprechend berücksichtigt werden muss.
- Für die Verkabelung muss **geschirmtes** Kabel mit genau zwei **verdrillten** Aderpaaren verwendet werden.
 - Ein verdrilltes Aderpaar wird für den Anschluss von CAN-H und CAN-L verwendet.
 - Die Adern des anderen Paares werden **gemeinsam** für CAN-GND verwendet.
 - Der Schirm des Kabels wird bei allen Knoten an die CAN-Shield-Anschlüsse geführt.
- Eine Tabelle mit den technischen Daten von verwendbaren Kabeln befindet sich am Ende dieses Kapitels, geeignete und von IEF Werner GmbH empfohlene Kabel finden sie im Produkthandbuch
- Von der Verwendung von Zwischensteckern bei der CAN-Bus-Verkabelung wird abgeraten. Sollte dies dennoch notwendig sein, ist zu beachten, dass metallische Steckergehäuse verwendet werden, um den Kabelschirm zu verbinden.
- Um die Störeinkopplung so gering wie möglich zu halten, sollten grundsätzlich
 - Motorkabel nicht parallel zu Signalleitungen verlegt werden.
 - Motorkabel gemäß der Spezifikation von IEF Werner GmbH ausgeführt sein.
 - Motorkabel ordnungsgemäß geschirmt und geerdet sein.
- Für weitere Informationen zum Aufbau einer störungsfreien CAN-Bus-Verkabelung verweisen wir auf die Controller Area Network protocol specification, Version 2.0 der Robert Bosch GmbH, 1991.
- Technische Daten CAN-Bus-Kabel:

2 Paare á 2 verdrillten Adern, $d \geq 0,22 \text{ mm}^2$ Geschirmt	Schleifenwiderstand $< 0,2/\text{m}$ Wellenwiderstand 100-120 Ω
--	---

4 Aktivierung von CANopen

4.1 Übersicht

Die Aktivierung des CAN-Interface mit dem Protokoll CANopen erfolgt einmalig über die serielle Schnittstelle des Servoreglers. Das CAN-Protokoll wird über das CAN-Bus-Fenster des S2Commander aktiviert.

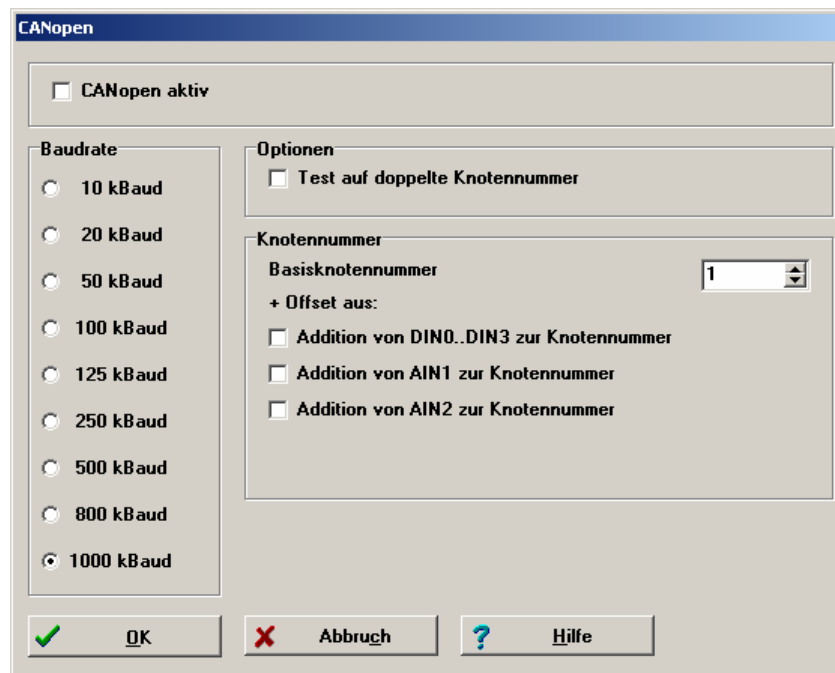


Abbildung 4.1: CAN-Bus-Fenster S2 Commander

Es müssen insgesamt 3 verschiedene Parameter eingestellt werden:

- Basis-Knotennummer**
 Zur eindeutigen Identifizierung im Netzwerk muss jedem Teilnehmer eine Knotennummer zugeteilt werden, die nur einmal im Netzwerk vorkommen darf. Über diese Knotennummer wird das Gerät adressiert.
 Als zusätzliche Option besteht die Möglichkeit die Knotennummer des Antriebsreglers von der äußeren Beschaltung abhängig zu machen. Zur Basis-Knotennummer wird einmalig nach dem Reset die Eingangskombination der digitalen Eingänge DIN0...DIN3 oder der analogen Eingänge AIN1 und AIN2 addiert. Dabei wird AIN1 mit einer Wertigkeit von 32 und AIN2 mit einer Wertigkeit von 64 hinzuaddiert, wenn der jeweilige Eingang auf $V_{ref} = 10V$ gebrückt ist.
- Baudrate**
 Dieser Parameter bestimmt die auf dem CAN-Bus verwendete Baudrate in kBaud. Beachten Sie, dass hohe Baudraten eine niedrige maximale Kabellänge erfordern.
- Optionen**
 Alle in einem CANopen-Netzwerk vorhandenen Geräte senden eine Einschaltmeldung (Bootup-Message) über den Bus, die die Knotennummer des Senders enthält. Empfängt der Regler eine solche Einschaltmeldung, die seiner eigenen Knotennummer entspricht, wird der Fehler 12-0 ausgelöst.

Letztlich kann das CANopen-Protokoll im Regler aktiviert werden. Beachten Sie, dass Sie die genannten Parameter nur ändern können, wenn das Protokoll deaktiviert ist.



Beachten Sie, dass die Parametrierung der CANopen-Funktionalität nach einem Reset nur erhalten bleibt, wenn der Parametersatz des Reglers gesichert wurde.

5 Zugriffsverfahren

5.1 Einleitung

CANopen stellt eine einfache und standardisierte Möglichkeit bereit, auf die Parameter des Servoreglers (z.B. den maximalen Motorstrom) zuzugreifen. Dazu ist jedem Parameter (*CAN-Objekt*) eine eindeutige Nummer (*Index und Subindex*) zugeordnet. Die Gesamtheit aller einstellbaren Parameter wird als *Objektverzeichnis* bezeichnet.

Für den Zugriff auf die CAN-Objekte über den CAN-Bus sind im Wesentlichen zwei Methoden verfügbar: Eine bestätigte Zugriffsart, bei der der Regler jeden Parameterzugriff quittiert (über sog. SDOs) und eine unbestätigte Zugriffsart, bei der keine Quittierung erfolgt (über sog. PDOs).

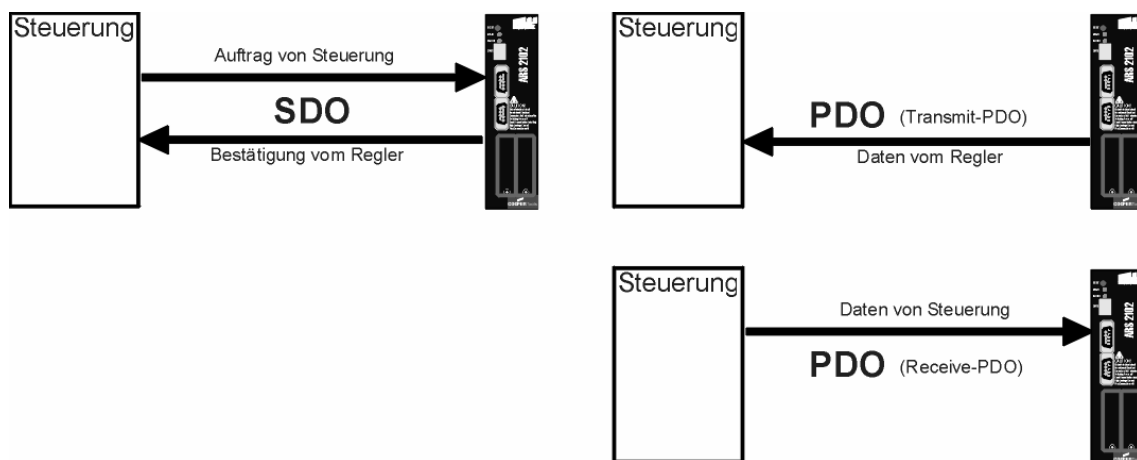


Abbildung 5.2: Zugriffsverfahren

In der Regel wird der Regler über SDO-Zugriffe sowohl parametrieren als auch steuern. Für spezielle Anwendungsfälle sind darüber hinaus noch weitere Arten von Nachrichten (sog. Kommunikations-Objekte) definiert, die entweder vom Regler oder der übergeordneten Steuerung gesendet werden:

SDO	Service Data Object	Werden zur normalen Parametrierung des Reglers verwendet.
PDO	Process Data Object	Schneller Austausch von Prozessdaten (z.B. Ist-Drehzahl) möglich.
SYNC	Synchronization Message	Synchronisierung mehrerer CAN-Knoten
EMCY	Emergency Message	Übermittlung von Fehlermeldungen.
NMT	Network Management	Netzwerkdienst: Es kann z.B. auf alle CAN-Knoten gleichzeitig eingewirkt werden.
HEARTBEAT	Error Control Protocol	Überwachung der Kommunikationsteilnehmer durch regelmäßige Nachrichten.

Jede Nachricht, die auf dem CAN-Bus verschickt wird, enthält eine Art Adresse, mit deren Hilfe festgestellt werden kann, für welchen Bus-Teilnehmer die Nachricht gedacht ist. Diese Nummer wird als *Identifier* bezeichnet. Je niedriger der Identifier, desto größer ist die Priorität der Nachricht. Für die oben genannten Kommunikationsobjekte sind jeweils Identifier festgelegt. Die folgende Skizze zeigt den prinzipiellen Aufbau einer CANopen-Nachricht:

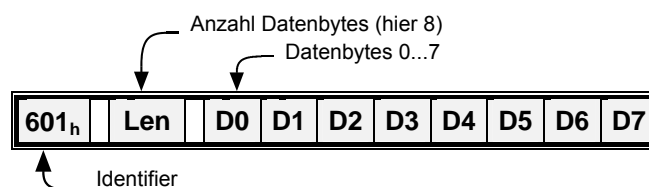


Abbildung 5.3: Identifier

5.2 SDO-Zugriff

Über die **Service-Data-Objekte** (SDO) kann auf das Objektverzeichnis des Reglers zugegriffen werden. Dieser Zugriff ist besonders einfach und übersichtlich. Es wird daher empfohlen, die Applikation zunächst nur mit SDOs aufzubauen und erst später einige Objektzugriffe auf die zwar schnelleren, aber auch komplizierteren **Process-Data-Objekte** (PDOs) umzustellen.

SDO-Zugriffe gehen immer von der übergeordneten Steuerung (Host) aus. Dieser sendet an den Regler entweder einen Schreibbefehl, um einen Parameter des Objektverzeichnisses zu ändern, oder einen Lesebefehl, um einen Parameter auszulesen. Zu jedem Befehl erhält der Host eine Antwort, die entweder den ausgelesenen Wert enthält oder – im Falle eines Schreibbefehls – als Quittung dient.

Damit der Regler erkennt, dass der Befehl für ihn bestimmt ist, muss der Host den Befehl mit einem bestimmten Identifier senden. **Dieser setzt sich aus der Basis 600_h + Knotennummer des betreffenden Reglers zusammen. Der Regler antwortet entsprechend mit dem Identifier 580_h + Knotennummer.**

Der Aufbau der Befehle bzw. der Antworten hängt vom Datentyp des zu lesenden oder schreibenden Objekts ab, da entweder 1, 2 oder 4 Datenbytes gesendet bzw. empfangen werden müssen.

Folgende Datentypen werden unterstützt:

UINT8	8-Bit-Wert ohne Vorzeichen	0 ... 255
INT8	8-Bit-Wert mit Vorzeichen	-128 ... 127
UINT16	16-Bit-Wert ohne Vorzeichen	0 ... 65535
INT16	16-Bit-Wert mit Vorzeichen	-32768 ... 32767
UINT32	32-Bit-Wert ohne Vorzeichen	0 ... (2 ³² -1)
INT32	32-Bit-Wert mit Vorzeichen	-(2 ³¹) ... (2 ³¹ -1)

5.2.1 SDO-Sequenzen zum Lesen und Schreiben

Um Objekte dieser Zahlentypen auszulesen oder zu beschreiben sind die nachfolgend aufgeführten Sequenzen zu verwenden. Die Kommandos, um einen Wert in den Regler zu schreiben, beginnen je nach Datentyp mit einer unterschiedlichen Kennung. Die Antwort-Kennung ist hingegen stets die gleiche. Lesebefehle beginnen immer mit der gleichen Kennung und der Regler antwortet je nach zurückgegebenem Datentyp unterschiedlich. Alle Zahlen sind in hexadezimaler Schreibweise gehalten.

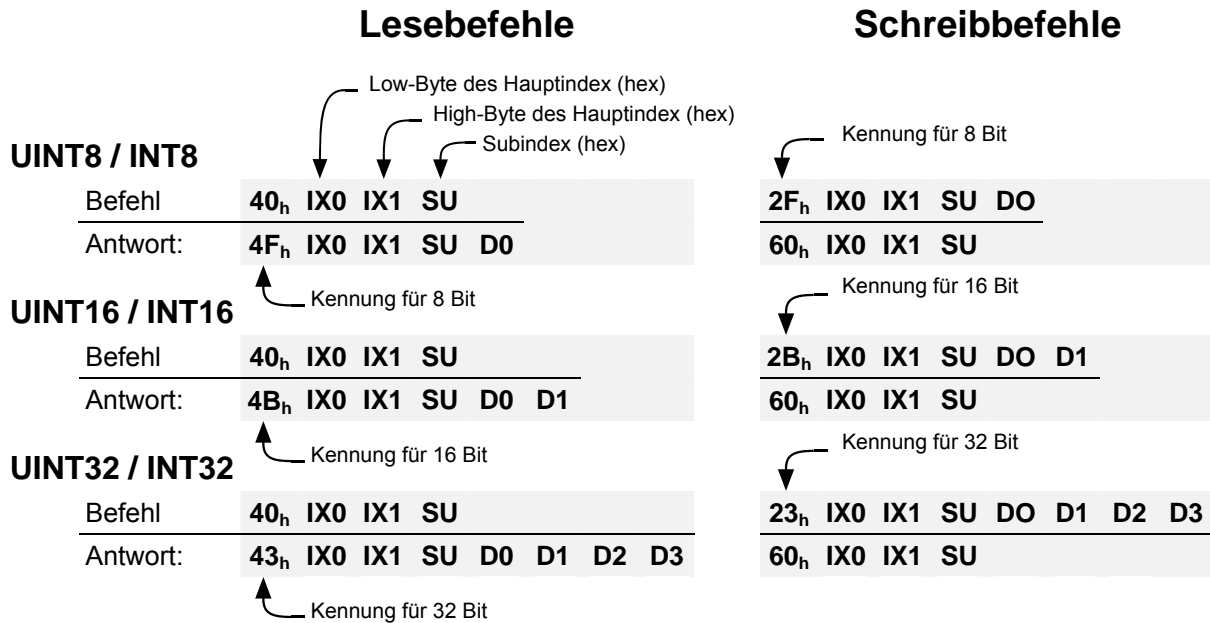


Abbildung 5.4: SDO-Sequenzen zum Lesen und Schreiben

BEISPIEL

<p>UINT8 / INT8</p> <p style="text-align: center;">Lesen von Obj. 6061_00_h Rückgabe-Daten: 01_h</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td style="padding: 2px;">Befehl</td><td style="padding: 2px;">40_h 61_h 60_h 00_h</td></tr> <tr><td style="padding: 2px;">Antwort:</td><td style="padding: 2px;">4F_h 61_h 60_h 00_h 01_h</td></tr> </table>	Befehl	40 _h 61 _h 60 _h 00 _h	Antwort:	4F _h 61 _h 60 _h 00 _h 01 _h	<p style="text-align: center;">Schreiben von Obj. 1401_02_h Daten: EF_h</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td style="padding: 2px;">2F_h 01_h 14_h 02_h EF_h</td></tr> <tr><td style="padding: 2px;">60_h 01_h 14_h 02_h</td></tr> </table>	2F _h 01 _h 14 _h 02 _h EF _h	60 _h 01 _h 14 _h 02 _h
Befehl	40 _h 61 _h 60 _h 00 _h						
Antwort:	4F _h 61 _h 60 _h 00 _h 01 _h						
2F _h 01 _h 14 _h 02 _h EF _h							
60 _h 01 _h 14 _h 02 _h							
<p>UINT16 / INT16</p> <p style="text-align: center;">Lesen von Obj. 6041_00_h Rückgabe-Daten: 1234_h</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td style="padding: 2px;">Befehl</td><td style="padding: 2px;">40_h 41_h 60_h 00_h</td></tr> <tr><td style="padding: 2px;">Antwort:</td><td style="padding: 2px;">4B_h 41_h 60_h 00_h 34_h 12_h</td></tr> </table>	Befehl	40 _h 41 _h 60 _h 00 _h	Antwort:	4B _h 41 _h 60 _h 00 _h 34 _h 12 _h	<p style="text-align: center;">Schreiben von Obj. 6040_00_h Daten: 03E8_h</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td style="padding: 2px;">2B_h 40_h 60_h 00_h E8_h 03_h</td></tr> <tr><td style="padding: 2px;">60_h 40_h 60_h 00_h</td></tr> </table>	2B _h 40 _h 60 _h 00 _h E8 _h 03 _h	60 _h 40 _h 60 _h 00 _h
Befehl	40 _h 41 _h 60 _h 00 _h						
Antwort:	4B _h 41 _h 60 _h 00 _h 34 _h 12 _h						
2B _h 40 _h 60 _h 00 _h E8 _h 03 _h							
60 _h 40 _h 60 _h 00 _h							
<p>UINT32 / INT32</p> <p style="text-align: center;">Lesen von Obj. 6093_01_h Rückgabe-Daten: 12345678_h</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td style="padding: 2px;">Befehl</td><td style="padding: 2px;">40_h 93_h 60_h 01_h</td></tr> <tr><td style="padding: 2px;">Antwort:</td><td style="padding: 2px;">43_h 93_h 60_h 01_h 78_h 56_h 34_h 12_h</td></tr> </table>	Befehl	40 _h 93 _h 60 _h 01 _h	Antwort:	43 _h 93 _h 60 _h 01 _h 78 _h 56 _h 34 _h 12 _h	<p style="text-align: center;">Schreiben von Obj. 6093_01_h Daten: 12345678_h</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td style="padding: 2px;">23_h 93_h 60_h 01_h 78_h 56_h 34_h 12_h</td></tr> <tr><td style="padding: 2px;">60_h 93_h 60_h 01_h</td></tr> </table>	23 _h 93 _h 60 _h 01 _h 78 _h 56 _h 34 _h 12 _h	60 _h 93 _h 60 _h 01 _h
Befehl	40 _h 93 _h 60 _h 01 _h						
Antwort:	43 _h 93 _h 60 _h 01 _h 78 _h 56 _h 34 _h 12 _h						
23 _h 93 _h 60 _h 01 _h 78 _h 56 _h 34 _h 12 _h							
60 _h 93 _h 60 _h 01 _h							

Die Quittierung vom Regler muss in jedem Fall abgewartet werden !
Erst wenn der Regler die Anforderung quittiert hat, dürfen weitere
Anforderungen gesendet werden.



5.2.2 SDO-Fehlermeldungen

Im Falle eines Fehlers beim Lesen oder Schreiben (z.B. weil der geschriebene Wert zu groß ist), antwortet der Regler mit einer Fehlermeldung anstelle der Quittierung:

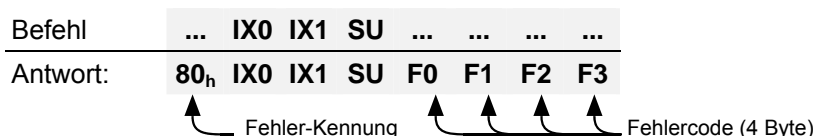


Abbildung 5.5: SDO-Fehlermeldungen

Fehlercode F3 F2 F1 F0	Bedeutung
05 03 00 00 _h	Protokollfehler: Toggle Bit wurde nicht geändert
05 04 00 01 _h	Protokollfehler: client / server command specifier ungültig oder unbekannt
06 06 00 00 _h	Zugriff fehlerhaft aufgrund eine Hardware-Problems * ¹⁾
06 01 00 00 _h	Zugriffsart wird nicht unterstützt.
06 01 00 01 _h	Lesezugriff auf ein Objekt, dass nur geschrieben werden kann
06 01 00 02 _h	Schreibzugriff auf ein Objekt, dass nur gelesen werden kann
06 02 00 00 _h	Das angesprochene Objekt existiert nicht im Objektverzeichnis
06 04 00 41 _h	Das Objekt darf nicht in ein PDO eingetragen werden (z.B. ro-Objekt in RPDO)
06 04 00 42 _h	Die Länge der in das PDO eingetragenen Objekte überschreitet die PDO-Länge
06 04 00 43 _h	Allgemeiner Parameterfehler
06 04 00 47 _h	Überlauf einer internen Größe / Genereller Fehler
06 07 00 10 _h	Protokollfehler: Länge des Service-Parameters stimmt nicht überein
06 07 00 12 _h	Protokollfehler: Länge des Service-Parameters zu groß
06 07 00 13 _h	Protokollfehler: Länge des Service-Parameters zu klein
06 09 00 11 _h	Der angesprochene Subindex existiert nicht
06 09 00 30 _h	Die Daten überschreiten den Wertebereich des Objekts
06 09 00 31 _h	Die Daten sind zu groß für das Objekt
06 09 00 32 _h	Die Daten sind zu klein für das Objekt
06 09 00 36 _h	Obere Grenze ist kleiner als untere Grenze
08 00 00 20 _h	Daten können nicht übertragen oder gespeichert werden * ¹⁾
08 00 00 21 _h	Daten können nicht übertragen oder gespeichert werden, da der Regler lokal arbeitet
08 00 00 22 _h	Daten können nicht übertragen oder gespeichert werden, da sich der Regler dafür nicht im richtigen Zustand befindet * ³⁾
08 00 00 23 _h	Es ist kein Object Dictionary vorhanden * ²⁾

*¹⁾ Werden gemäß DS301 bei fehlerhaftem Zugriff auf store_parameters / restore_parameters zurückgegeben.

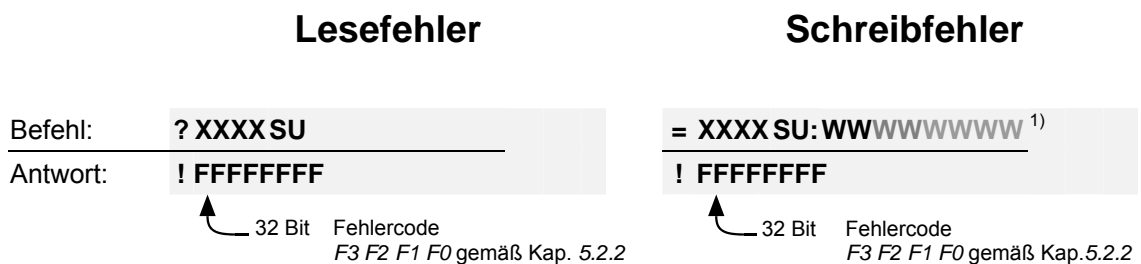
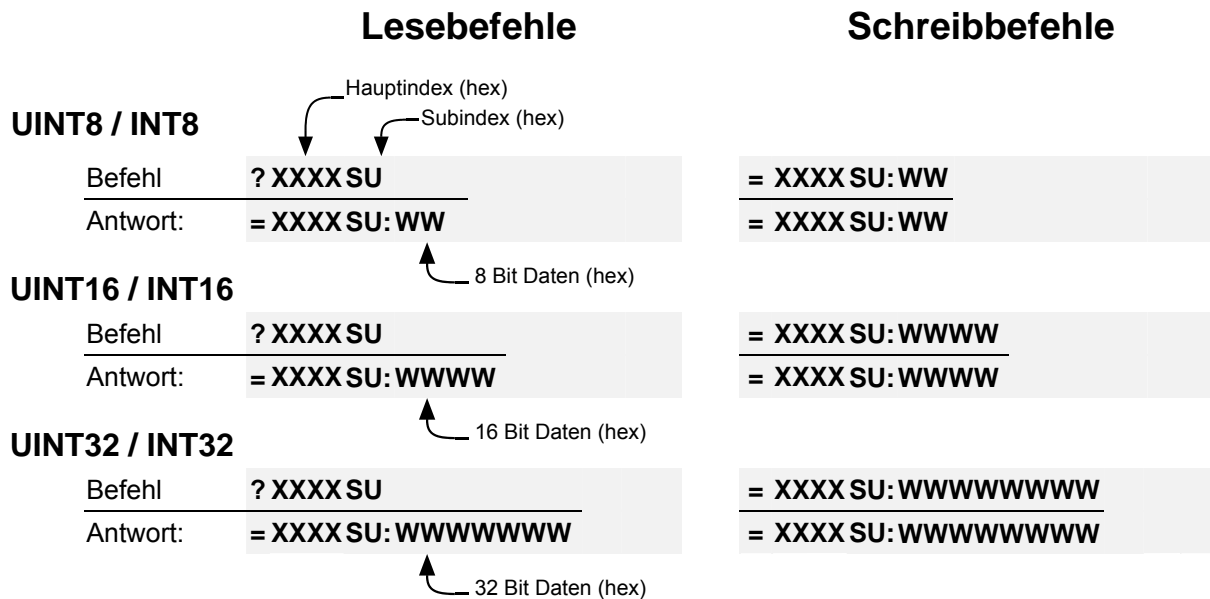
*²⁾ Dieser Fehler wird z.B. zurückgegeben, wenn ein anderes Bussystem den Regler kontrolliert oder der Parameterzugriff nicht erlaubt ist.

*³⁾ „Zustand“ ist hier allgemein zu verstehen: Es kann sich dabei sowohl um die falsche Betriebsart handeln, als auch um ein nicht vorhandenes Technologie-Modul o.ä.

5.2.3 Simulation von SDO-Zugriffen über RS232

Die Firmware der Servoregler bietet die Möglichkeit, SDO-Zugriffe über die RS232-Schnittstelle zu simulieren. So können in der Testphase Objekte nach dem Einschreiben über den CAN-Bus über die RS232-Schnittstelle gelesen und kontrolliert werden. Durch Verwendung des Transfer-Fensters des S2Commander (unter *Datei/Transfer*) wird so die Applikationserstellung erleichtert.


Die Syntax der Befehle lautet:



¹⁾ Die Antwort ist im Fehlerfall für alle 3 Schreibbefehle (8, 16, 32 Bit) gleich aufgebaut.

Abbildung 5.6: Simulation von SDO-Zugriffen über RS232

Die Befehle werden als Zeichen ohne jegliche Leerzeichen eingegeben.



Verwenden Sie diese Testbefehle niemals in Applikationen !

Der Zugriff über RS232 dient lediglich zu Testzwecken und ist nicht für eine echtzeitfähige Kommunikation geeignet.

Darüber hinaus kann die Syntax der Testbefehle jederzeit geändert werden.

5.3 PDO-Message

Mit **Process-Data-Objekten** (PDOs) können Daten ereignisgesteuert übertragen werden. Das PDO überträgt dabei einen oder mehrere vorher festgelegte Parameter. Anders als bei einem SDO erfolgt bei der Übertragung eines PDOs keine Quittierung. Nach der PDO-Aktivierung müssen daher alle Empfänger jederzeit eventuell ankommende PDOs verarbeiten können. Dies bedeutet meistens einen erheblichen Softwareaufwand im Host-Rechner. Diesem Nachteil steht der Vorteil gegenüber, dass der Host-Rechner die durch ein PDO übertragenen Parameter nicht zyklisch abzufragen braucht, was zu einer starken Verminderung der CAN-Busauslastung führt.

BEISPIEL



Der Host-Rechner möchte wissen, wann der Regler eine Positionierung von A nach B abgeschlossen hat.

Bei der Verwendung von SDOs muss er hierzu ständig, beispielsweise jede Millisekunde, das Objekt **statusword** abfragen, womit er die Buskapazität stark auslastet.

Bei der Verwendung eines PDOs wird der Regler schon beim Start der Applikation so parametrieren, dass er bei jeder Veränderung des Objektes **statusword** ein PDO absetzt, in dem das Objekt **statusword** enthalten ist.

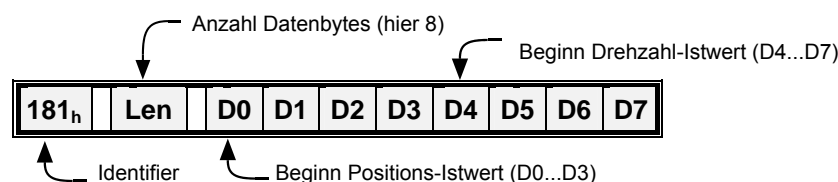
Statt ständig nachzufragen, wird dem Host-Rechner somit automatisch eine entsprechende Meldung zugestellt, sobald das Ereignis eingetreten ist.

Folgende Typen von PDOs werden unterschieden:

Transmit-PDO (T-PDO)	Regler <input type="checkbox"/> Host	Regler sendet PDO bei Auftreten eines bestimmten Ereignisses
Receive-PDO (R-PDO)	Host <input type="checkbox"/> Regler	Regler wertet PDO bei Auftreten eines bestimmten Ereignisses aus

Der Regler verfügt über vier Transmit- und vier Receive-PDOs.

In die PDOs können nahezu alle Objekte des Objektverzeichnisses eingetragen (gemappt) werden, d.h. das PDO enthält als Daten z.B. den Drehzahl-Istwert, den Positions-Istwert o.ä. Welche Daten übertragen werden, muss dem Regler vorher mitgeteilt werden, da das PDO lediglich Nutzdaten und keine Information über die Art des Parameters enthält. In der unteren Beispiel würde in den Datenbytes 0...3 des PDOs der Positions-Istwert und in den Bytes 4...7 der Drehzahl-Istwert übertragen.



Auf diese Art können nahezu beliebige Datentelegramme definiert werden. Die folgenden Kapitel beschreiben die dazu nötigen Einstellungen.

5.3.1 Beschreibung der Objekte

Identifizier des PDOs **COB_ID_used_by_PDO**

In dem Objekt **COB_ID_used_by_PDO** ist der Identifizier einzutragen, auf dem das jeweilige PDO gesendet bzw. empfangen werden soll. Ist Bit 31 gesetzt, ist das jeweilige PDO deaktiviert. Dies ist die Voreinstellung für alle PDOs.

Die COB-ID darf nur geändert werden, wenn das PDO deaktiviert, d.h. Bit 31 gesetzt ist. Ein anderer Identifizier als aktuell im Regler eingestellt darf daher nur geschrieben werden, wenn gleichzeitig Bit 31 gesetzt ist.

Das gesetzte Bit 30 beim Lesen des Identifiziers zeigt an, dass das Objekt nicht durch ein Remoteframe abgefragt werden kann. Dieses Bit wird beim Schreiben ignoriert und ist beim Lesen immer gesetzt.

Anzahl zu übertragender Objekte

number_of_mapped_objects

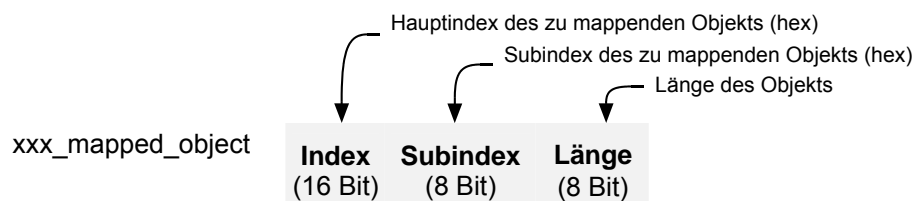
Dieses Objekt gibt an, wie viele Objekte in das entsprechende PDO gemappt werden sollen. Folgende Einschränkungen sind zu beachten:

- Es können pro PDO maximal 4 Objekte gemappt werden
- Ein PDO darf über maximal 64 Bit (8 Byte) verfügen.

Zu übertragende Objekte

first_mapped_object ... fourth_mapped_object

Für jedes Objekt, das im PDO enthalten sein soll muss dem Regler der entsprechende Index, der Subindex und die Länge mitgeteilt werden. Die Längenangabe muss mit der Längenangabe im Object Dictionary übereinstimmen. Teile eines Objekts können nicht gemappt werden. Die Mapping-Informationen besitzen folgendes Format:



Zur Vereinfachung des Mappings ist folgendes Vorgehen vorgeschrieben:

- 1.) Die Anzahl der gemappten Objekte wird auf 0 gesetzt.
- 2.) Die Parameter **first_mapped_object...fourth_mapped_object** dürfen beschrieben werden (Die Gesamtlänge aller Objekte ist in dieser Zeit nicht relevant).
- 3.) Die Anzahl der gemappten Objekte wird auf einen Wert zwischen 1...4 gesetzt. Die Länge all dieser Objekte darf jetzt 64 Bit nicht überschreiten.

Übertragungsart

transmission_type und inhibit_time

Für jedes PDO kann festgelegt werden, welches Ereignis zum Aussenden (Transmit-PDO) bzw. Auswerten (Receive-PDO) einer Nachricht führt:

Wert	Bedeutung	Erlaubt bei
01 _h –F0 _h	SYNC-Message Der Zahlenwert gibt an, wie viele SYNC-Nachrichten eingetroffen sein müssen, bevor das PDO - gesendet (T-PDO) bzw. - ausgewertet (R-PDO) wird.	TPDOs RPDOs
FE _h	Zyklisch Das Transfer-PDO wird vom Regler zyklisch aktualisiert und gesendet. Die Zeitspanne wird durch das Objekt inhibit_time festgelegt. Receive-PDOs werden hingegen unmittelbar nach Empfang ausgewertet.	TPDOs (RPDOs)
FF _h	Änderung Das Transfer-PDO wird gesendet, wenn sich in den Daten des PDOs mindestens 1 Bit geändert hat. Mit inhibit_time kann zusätzlich der minimale Abstand zwischen dem Absenden zweier PDOs in 100µs-Schritten festgelegt werden.	TPDOs

Die Verwendung aller anderen Werte ist nicht zulässig.

Maskierung

transmit_mask_high und transmit_mask_low

Wird als **transmission_type** „Änderung“ gewählt, wird das TPDO immer gesendet, wenn sich mindestens 1 Bit des TPDOs ändert. Häufig wird es aber benötigt, dass das TPDO nur gesendet wird, wenn sich bestimmte Bits geändert haben. Daher kann das TPDO mit einer Maske versehen werden: Nur die Bits des TPDOs, die in der Maske auf „1“ gesetzt sind, werden zur Auswertung, ob sich das PDO geändert hat herangezogen. Da diese Funktion herstellerspezifisch ist, sind als Defaultwert alle Bits der Masken gesetzt.



BEISPIEL

Folgende Objekte sollen zusammen in einem PDO übertragen werden:

Name des Objekts	Index_Subindex	Bedeutung
statusword	6041 _h _00 _h	Reglersteuerung
modes_of_operation_display	6061 _h _00 _h	Betriebsart
digital_inputs	60FD _h _00 _h	Digitale Eingänge

Es soll das erste Transmit-PDO (TPDO 1) verwendet werden, welches immer gesendet werden soll, wenn sich eines der digitalen Eingänge ändert, allerdings maximal alle 10 ms. Als Identifier für dieses PDO soll 187_h verwendet werden.

1.) PDO deaktivieren

Falls das PDO aktiv ist, muss es zuerst deaktiviert werden.

Schreiben des Identifiers mit gesetztem Bit 31 (PDO ist deaktiviert):

⇒ **cob_id_used_by_pdo = C0000187_h**

2.) Anzahl der Objekte löschen

Damit das Objektmapping geändert werden darf, Anzahl der Objekte auf Null setzen.

⇒ **number_of_mapped_objects = 0**

3.) Objekte, die gemappt werden sollen, parametrieren

Die oben aufgeführten Objekte müssen jeweils zu einem 32 Bit-Wert zusammengesetzt werden:

Index = 6041_h Subin. = 00_h Länge = 10_h ⇒ **first_mapped_object = 60410010_h**

Index = 6061_h Subin. = 00_h Länge = 08_h ⇒ **second_mapped_object = 60610008_h**

Index = 60FD_h Subin. = 00_h Länge = 20_h ⇒ **third_mapped_object = 60FD0020_h**

4.) Anzahl der Objekte parametrieren

Es sollen 3 Objekte im PDO enthalten sein

⇒ **number_of_mapped_objects = 3_h**

5.) Übertragungsart parametrieren

Das PDO soll bei Änderung (der digitalen Eingänge) gesendet werden.

⇒ **transmission_type = FF_h**

Damit nur die Änderung der digitalen Eingänge zum Senden führt, wird das PDO maskiert, so dass nur die 16 Bits des Objekts 60FD_h „durchkommen“.

⇒ **transmit_mask_high = 00FFFF00_h**

⇒ **transmit_mask_low = 00000000_h**

Das PDO soll höchstens alle 10 ms (100×100µs) gesendet werden.

⇒ **inhibit_time = 64_h**

6.) Identifier parametrieren

Das PDO soll mit Identifier 187_h gesendet werden.

Schreiben des neuen Identifier und Aktivieren des PDOs durch Löschen von Bit 31:

⇒ **cob_id_used_by_pdo = 40000187_h**



Beachten Sie, dass die Parametrierung der PDOs generell nur geändert werden darf, wenn der Netzwerkstatus (NMT) nicht **operational** ist. Siehe hierzu auch Abschnitt 5.6.

5.3.2 Objekte zur PDO-Parametrierung

In den Reglern der SERVOTEC S2 -Reihe sind insgesamt 4 Transmit und 4 Receive-PDOs verfügbar. Die einzelnen Objekte, um diese PDOs zu parametrieren sind jeweils für alle 4 TPDOs und alle 4 RPDOs gleich. Daher ist im Folgenden nur die Parameterbeschreibung des ersten TPDOs explizit aufgeführt. Sie ist sinngemäß auch für die anderen PDOs zu verwenden, die im Anschluss tabellarisch aufgeführt sind:

Index	1800_h
Name	transmit_pdo_parameter_tpdo1
Object Code	RECORD
No. of Elements	3

Sub-Index	01_h
Description	cob_id_used_by_pdo_tpdo1
Data Type	UINT32
Access	rw
PDO Mapping	no
Units	--
Value Range	181 _h ...1FF _h , Bit 30 und 31 dürfen gesetzt sein
Default Value	C0000181 _h

Sub-Index	02_h
Description	transmission_type_tpdo1
Data Type	UINT8
Access	rw
PDO Mapping	no
Units	--
Value Range	0...8C _h , FE _h , FF _h
Default Value	FF _h

Sub-Index	03_h
Description	inhibit_time_tpdo1
Data Type	UINT16
Access	rw
PDO Mapping	no
Units	100µs (i.e. 10 = 1ms)
Value Range	--
Default Value	0

Index	1A00_h
Name	transmit_pdo_mapping_tpdo1
Object Code	RECORD
No. of Elements	2

Sub-Index	00_h
Description	number_of_mapped_objects_tpdo1
Data Type	UINT8
Access	rw
PDO Mapping	no
Units	--
Value Range	0...4
Default Value	siehe Tabelle

Sub-Index	01_h
Description	first_mapped_object_tpdo1
Data Type	UINT32
Access	rw
PDO Mapping	no
Units	--
Value Range	--
Default Value	siehe Tabelle

Sub-Index	02_h
Description	second_mapped_object_tpdo1
Data Type	UINT32
Access	rw
PDO Mapping	no
Units	--
Value Range	--
Default Value	siehe Tabelle

Sub-Index	03_h
Description	third_mapped_object_tpdo1
Data Type	UINT32
Access	rw
PDO Mapping	no
Units	--
Value Range	--
Default Value	siehe Tabelle

Sub-Index	04_h
Description	fourth_mapped_object_tpdo1
Data Type	UINT32
Access	rw
PDO Mapping	no
Units	--
Value Range	--
Default Value	siehe Tabelle



Beachten Sie, dass die Objekt- Gruppen **transmit_pdo_parameter_XXX** und **transmit_pdo_mapping_XXX** nur beschrieben werden können, wenn das PDO deaktiviert ist (Bit 31 in **cob_id_used_by_pdo_XXX** gesetzt).

1. Transmit-PDO

Index	Comment	Type	Acc.	Default Value
1800 _{h_00h}	number of entries	UINT8	ro	03 _h
1800 _{h_01h}	COB-ID used by PDO	UINT32	rw	C0000181 _h
1800 _{h_02h}	transmission type	UINT8	rw	FF _h
1800 _{h_03h}	inhibit time (100 µs)	UINT16	rw	0000 _h
1A00 _{h_00h}	number of mapped objects	UINT8	rw	01 _h
1A00 _{h_01h}	first mapped object	UINT32	rw	60410010 _h
1A00 _{h_02h}	second mapped object	UINT32	rw	00000000 _h
1A00 _{h_03h}	third mapped object	UINT32	rw	00000000 _h
1A00 _{h_04h}	fourth mapped object	UINT32	rw	00000000 _h

2. Transmit-PDO

Index	Comment	Type	Acc.	Default Value
1801 _{h_00h}	number of entries	UINT8	ro	03 _h
1801 _{h_01h}	COB-ID used by PDO	UINT32	rw	C0000281 _h
1801 _{h_02h}	transmission type	UINT8	rw	FF _h
1801 _{h_03h}	inhibit time (100 µs)	UINT16	rw	0000 _h
1A01 _{h_00h}	number of mapped objects	UINT8	rw	02 _h
1A01 _{h_01h}	first mapped object	UINT32	rw	60410010 _h
1A01 _{h_02h}	second mapped object	UINT32	rw	60610008 _h
1A01 _{h_03h}	third mapped object	UINT32	rw	00000000 _h
1A01 _{h_04h}	fourth mapped object	UINT32	rw	00000000 _h

3. Transmit-PDO

Index	Comment	Type	Acc.	Default Value
1802 _{h_00h}	number of entries	UINT8	ro	03 _h
1802 _{h_01h}	COB-ID used by PDO	UINT32	rw	C0000381 _h
1802 _{h_02h}	transmission type	UINT8	rw	FF _h
1802 _{h_03h}	inhibit time (100 µs)	UINT16	rw	0000 _h
1A02 _{h_00h}	number of mapped objects	UINT8	rw	02 _h
1A02 _{h_01h}	first mapped object	UINT32	rw	60410010 _h
1A02 _{h_02h}	second mapped object	UINT32	rw	60640020 _h
1A02 _{h_03h}	third mapped object	UINT32	rw	00000000 _h
1A02 _{h_04h}	fourth mapped object	UINT32	rw	00000000 _h

4. Transmit-PDO

Index	Comment	Type	Acc.	Default Value
1803 _h _00 _h	number of entries	UINT8	ro	03 _h
1803 _h _01 _h	COB-ID used by PDO	UINT32	rw	C0000481 _h
1803 _h _02 _h	transmission type	UINT8	rw	FF _h
1803 _h _03 _h	inhibit time (100 µs)	UINT16	rw	0000 _h
1A03 _h _00 _h	number of mapped objects	UINT8	rw	02 _h
1A03 _h _01 _h	first mapped object	UINT32	rw	60410010 _h
1A03 _h _02 _h	second mapped object	UINT32	rw	606C0020 _h
1A03 _h _03 _h	third mapped object	UINT32	rw	00000000 _h
1A03 _h _04 _h	fourth mapped object	UINT32	rw	00000000 _h

tpdo_1_transmit_mask

Index	Comment	Type	Acc.	Default Value
2014 _h _00 _h	number of entries	UINT8	ro	02 _h
2014 _h _01 _h	tpdo_1_transmit_mask_low	UINT32	rw	FFFFFFFF _h
2014 _h _02 _h	tpdo_1_transmit_mask_high	UINT32	rw	FFFFFFFF _h

tpdo_2_transmit_mask

Index	Comment	Type	Acc.	Default Value
2015 _h _00 _h	number of entries	UINT8	ro	02 _h
2015 _h _01 _h	tpdo_2_transmit_mask_low	UINT32	rw	FFFFFFFF _h
2015 _h _02 _h	tpdo_2_transmit_mask_high	UINT32	rw	FFFFFFFF _h

tpdo_3_transmit_mask

Index	Comment	Type	Acc.	Default Value
2016 _h _00 _h	number of entries	UINT8	ro	02 _h
2016 _h _01 _h	tpdo_3_transmit_mask_low	UINT32	rw	FFFFFFFF _h
2016 _h _02 _h	tpdo_3_transmit_mask_high	UINT32	rw	FFFFFFFF _h

tpdo_4_transmit_mask

Index	Comment	Type	Acc.	Default Value
2017 _h _00 _h	number of entries	UINT8	ro	02 _h
2017 _h _01 _h	tpdo_4_transmit_mask_low	UINT32	rw	FFFFFFFF _h
2017 _h _02 _h	tpdo_4_transmit_mask_high	UINT32	rw	FFFFFFFF _h

1. Receive PDO

Index	Comment	Type	Acc.	Default Value
1400 _h _00 _h	number of entries	UINT8	ro	02 _h
1400 _h _01 _h	COB-ID used by PDO	UINT32	rw	C0000201 _h
1400 _h _02 _h	transmission type	UINT8	rw	FF _h
1600 _h _00 _h	number of mapped objects	UINT8	rw	01 _h
1600 _h _01 _h	first mapped object	UINT32	rw	60400010 _h
1600 _h _02 _h	second mapped object	UINT32	rw	00000000 _h
1600 _h _03 _h	third mapped object	UINT32	rw	00000000 _h
1600 _h _04 _h	fourth mapped object	UINT32	rw	00000000 _h

2. Receive PDO

Index	Comment	Type	Acc.	Default Value
1401 _h _00 _h	number of entries	UINT8	ro	02 _h
1401 _h _01 _h	COB-ID used by PDO	UINT32	rw	C0000301 _h
1401 _h _02 _h	transmission type	UINT8	rw	FF _h
1601 _h _00 _h	number of mapped objects	UINT8	rw	02 _h
1601 _h _01 _h	first mapped object	UINT32	rw	60400010 _h
1601 _h _02 _h	second mapped object	UINT32	rw	60600008 _h
1601 _h _03 _h	third mapped object	UINT32	rw	00000000 _h
1601 _h _04 _h	fourth mapped object	UINT32	rw	00000000 _h

3. Receive PDO

Index	Comment	Type	Acc.	Default Value
1402 _h _00 _h	number of entries	UINT8	ro	02 _h
1402 _h _01 _h	COB-ID used by PDO	UINT32	rw	C0000401 _h
1402 _h _02 _h	transmission type	UINT8	rw	FF _h
1602 _h _00 _h	number of mapped objects	UINT8	rw	02 _h
1602 _h _01 _h	first mapped object	UINT32	rw	60400010 _h
1602 _h _02 _h	second mapped object	UINT32	rw	607A0020 _h
1602 _h _03 _h	third mapped object	UINT32	rw	00000000 _h
1602 _h _04 _h	fourth mapped object	UINT32	rw	00000000 _h

4. Receive PDO

Index	Comment	Type	Acc.	Default Value
1403 _h _00 _h	number of entries	UINT8	ro	02 _h
1403 _h _01 _h	COB-ID used by PDO	UINT32	rw	C0000501 _h
1403 _h _02 _h	transmission type	UINT8	rw	FF _h
1603 _h _00 _h	number of mapped objects	UINT8	rw	02 _h
1603 _h _01 _h	first mapped object	UINT32	rw	60400010 _h
1603 _h _02 _h	second mapped object	UINT32	rw	60FF0020 _h
1603 _h _03 _h	third mapped object	UINT32	rw	00000000 _h
1603 _h _04 _h	fourth mapped object	UINT32	rw	00000000 _h

5.3.3 Aktivierung der PDOs

Damit der Regler PDOs sendet oder empfängt müssen folgende Punkte erfüllt sein:

- Das Objekt **number_of_mapped_objects** muss ungleich Null sein.
- Im Objekt **cob_id_used_for_pdos** muss das Bit 31 gelöscht sein.
- Der Kommunikationsstatus des Reglers muss **operational** sein (siehe Abschnitt 5.6: *Netzwerkmanagement (NMT-Service)* *Netzwerkmanagement (NMT-Service)*, ab Seite 52)

Damit PDOs parametrieren werden können, müssen folgende Punkte erfüllt sein:

- Der Kommunikationsstatus des Reglers darf nicht **operational** sein.

5.4 SYNC-Message

Mehrere Geräte einer Anlage können miteinander synchronisiert werden. Hierzu sendet eines der Geräte (meistens die übergeordnete Steuerung) periodisch Synchronisations-Nachrichten aus. Alle angeschlossenen Regler empfangen diese Nachrichten und verwenden sie für die Behandlung der PDOs (siehe Abschnitt 5.3: *PDO-Message*, ab Seite 35).

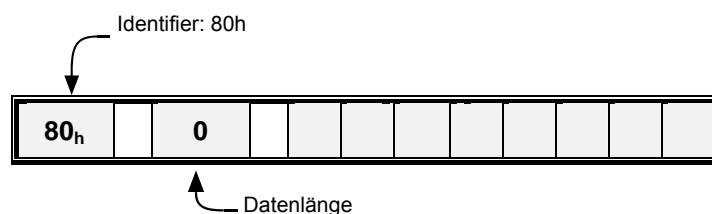


Abbildung 5.7: Identifizier SYNC-Message

Der Identifizier, auf dem der Regler die SYNC-Message empfängt, ist fest auf 080h eingestellt. Der Identifizier kann über das Objekt **cob_id_sync** ausgelesen werden.

Index	1005 _h
Name	cob_id_sync
Object Code	VAR
Data Type	UINT32

Access	rw
PDO Mapping	no
Units	--
Value Range	80000080 _h , 00000080 _h
Default Value	00000080 _h

5.5 EMERGENCY-Message

Der Antriebsregler überwacht die Funktion seiner wesentlichen Baugruppen. Hierzu zählen die Spannungsversorgung, die Endstufe, die Winkelgeberauswertung und die Technologiesteckplätze. Außerdem werden laufend der Motor (Temperatur, Winkelgeber) und die Endschalter überprüft. Auch Fehlparametrierungen können zu Fehlermeldungen führen (Division durch Null etc.).

Beim Auftreten eines Fehlers wird in der Anzeige des Reglers die Fehlernummer angezeigt. Wenn mehrere Fehlermeldungen gleichzeitig auftreten, so wird in der Anzeige immer die Nachricht mit der höchsten Priorität (der geringsten Nummer) angezeigt.

5.5.1 Übersicht

Der Regler sendet beim Auftreten eines Fehlers, oder wenn eine Fehlerquittierung durchgeführt wird, eine EMERGENCY-Message. Der Identifier dieser Nachricht wird aus dem Identifier **80_h** und der **Knotennummer** des betroffenen Reglers zusammengesetzt.

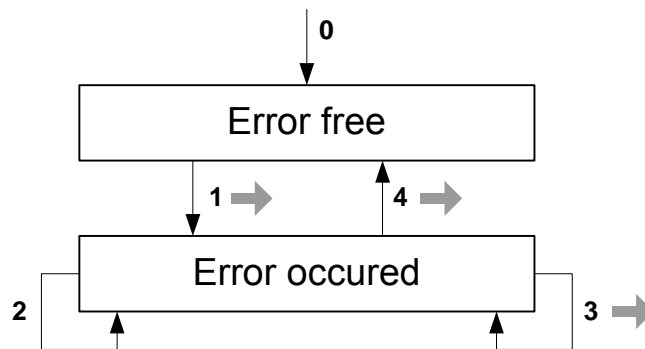


Abbildung 5.8: Emergency-Message

Nach einem Reset befindet sich der Regler im Zustand *Error free* (den er ggf. sofort wieder verlässt, weil von Anfang an ein Fehler vorhanden ist). Folgende Zustandsübergänge sind möglich:

Nr.	Ursache	Bedeutung
0	Initialisierung abgeschlossen	
1	Fehler tritt auf	Es lag kein Fehler vor und ein Fehler tritt auf. Ein EMERGENCY-Telegramm mit dem Fehlercode des aufgetretenen Fehlers wird gesendet.
2	Fehlerquittierung	Eine Fehlerquittierung (siehe Abschnitt 7.1.3.1: <i>Objekt 6040h: controlword, ab Seite 159</i>) wird versucht, aber nicht alle Ursachen sind behoben.
3	Fehler tritt auf	Es liegt schon ein Fehler vor und ein weiterer Fehler tritt auf. Ein EMERGENCY-Telegramm mit dem Fehlercode des neuen Fehlers wird gesendet.
4	Fehlerquittierung	Eine Fehlerquittierung wird versucht und alle Ursachen sind behoben. Es wird ein EMERGENCY-Telegramm mit dem Fehlercode 0000 gesendet.

5.5.2 Aufbau der EMERGENCY-Message

Die EMERGENCY-Message besteht aus acht Datenbytes, wobei in den ersten beiden Bytes ein **error_code** steht, die in folgender Tabelle aufgeführt sind. Im dritten Byte steht ein weiterer Fehlercode (Objekt 1001_h). Die restlichen fünf Bytes enthalten Nullen.

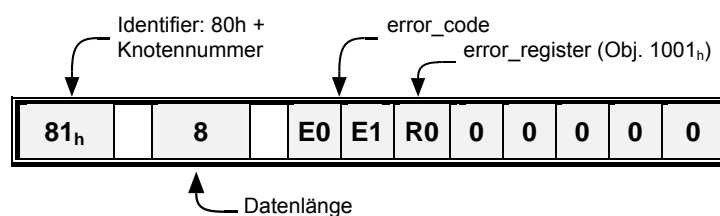


Abbildung 5.9: Aufbau der EMERGENCY-Message

Folgende Fehlercodes können auftreten:

error_code (hex)	Anzeige	Bedeutung
0000	--	Regler ist fehlerfrei
6180	E 01 0	Stack Overflow
3220	E 02 0	Unterspannung Zwischenkreis
4310	E 03 x	Übertemperatur Motor
4210	E 04 0	Übertemperatur Leistungsteil
4280	E 04 1	Übertemperatur Zwischenkreis
5114	E 05 0	Ausfall interne Spannung 1
5115	E 05 1	Ausfall interne Spannung 2
5116	E 05 2	Ausfall Treiberversorgung
5410	E 05 3	Unterspannung digitale I/O
5410	E 05 4	Überstrom digitale I/O
2320	E 06 x	Kurzschluss Endstufe
3210	E 07 0	Überspannung
7380	E 08 0	Winkelgeberfehler Resolver
7382	E 08 2	Fehler Spursignale Z0 Inkrementalgeber
7383	E 08 3	Fehler Spursignale Z1 Inkrementalgeber
7384	E 08 4	Fehler Spursignale digitaler Inkrementalgeber
7385	E 08 5	Fehler Spursignale Hallgebersignale Inkrementalgeber
7386	E 08 6	Kommunikationsfehler Winkelgeber
7387	E 08 7	Signalamplitude Inkrementalspur fehlerhaft
7388	E 08 8	Interner Winkelgeberfehler
7389	E 08 9	Winkelgeber an X2b wird nicht unterstützt
73A1	E 09 0	Winkelgeberparametersatz Typ ARS
73A2	E 09 1	Winkelgeberparametersatz kann nicht decodiert werden
73A3	E 09 2	Winkelgeberparametersatz: Version unbekannt
73A4	E 09 3	Winkelgeberparametersatz: Datenstruktur defekt
73A5	E 09 7	EEPROM Winkelgeber schreibgeschützt
73A6	E 09 9	EEPROM Winkelgeber zu klein
8A80	E 11 0	Referenzfahrt: Fehler beim Start
8A81	E 11 1	Fehler während einer Referenzfahrt
8A82	E 11 2	Referenzfahrt: Nullimpulsfehler
8A83	E 11 3	Referenzfahrt: Zeitüberschreitung
8A84	E 11 4	Referenzfahrt: Falscher / ungültiger Endschalter
8A85	E 11 5	Referenzfahrt: I ² t / Schleppfehler
8A86	E 11 6	Referenzfahrt: Ende der Suchstrecke
8180	E 12 0	CAN-Bus: Doppelte Knotennummer
8120	E 12 1	Kommunikationsfehler CAN: BUS OFF
8181	E 12 2	Kommunikationsfehler CAN beim Senden
8182	E 12 3	Kommunikationsfehler CAN beim Empfangen
6185	E 15 0	Division durch 0
6186	E 15 1	Bereichüberschreitung (Über-/Unterlauf)
6181	E 16 0	Programmausführung fehlerhaft
6182	E 16 1	Illegaler Interrupt
6187	E 16 2	Initialisierungsfehler
6183	E 16 3	Unerwarteter Zustand
8611	E 17 x	Überschreitung Grenzwert Schleppfehler
5280	E 21 1	Fehler 1 Strommessung U
5281	E 21 1	Fehler 1 Strommessung V
5282	E 21 2	Fehler 2 Strommessung U
5283	E 21 3	Fehler 2 Strommessung V
6080	E 25 0	Ungültiger Gerätetyp
6081	E 25 1	Nicht unterstützter Gerätetyp
6082	E 25 2	Nicht unterstützte HW- Revision
6083	E 25 3	Gerätefunktion beschränkt

error_code (hex)	Anzeige	Bedeutung
5580	E 26 0	Fehlender User-Parametersatz
5581	E 26 1	Checksummenfehler
5582	E 26 2	Flash: Fehler beim Schreiben
5583	E 26 3	Flash: Fehler beim Löschen
5584	E 26 4	Flash: Fehler im internen Flash
5585	E 26 5	Fehlende Kalibrierdaten
5586	E 26 6	Fehlende User- Positionsdatensätze
8611	E 27 0	Warnschwelle Schleppefehler
FF01	E 28 0	Betriebsstundenzähler fehlt
FF02	E 28 1	Betriebsstundenzähler: Schreibfehler
FF03	E 28 2	Betriebsstundenzähler korrigiert
FF04	E 28 3	Betriebsstundenzähler konvertiert
6380	E 30 0	Interner Umrechnungsfehler
2312	E 31 0	I ² T – Motor
2311	E 31 1	I ² T – Servoregler
2313	E 31 2	I ² T – PFC
2314	E 31 3	I ² T – Bremswiderstand
3280	E 32 0	Ladezeit Zwischenkreis überschritten
3281	E 32 1	Unterspannung für aktive PFC
3282	E 32 5	Überlast Bremschopper
3283	E 32 6	Entladezeit Zwischenkreis überschritten
3284	E 32 7	Leistungsversorgung fehlt für Reglerfreigabe
3285	E 32 8	Ausfall Leistungsversorgung bei Reglerfreigabe
3286	E 32 9	Phasenausfall
8A87	E 33 0	Schleppefehler Encoder-Emulation
8780	E 34 0	Synchronisationsfehler (Aufsynchronisierung)
8781	E 34 1	Synchronisationsfehler (Synchronisierung ausgefallen)
8480	E 35 0	Durchdrehschutz Linearmotor
6320	E 36 x	Parameter wurde limitiert
8612	E 40 x	SW-Endschalter erreicht
8680	E 42 0	Positionierung: Antrieb stoppt aufgrund fehlender Anschlusspositionierung
8681	E 42 1	Positionierung: Antrieb stoppt weil Drehrichtungsumkehr nicht erlaubt
8682	E 42 2	Positionierung: Unerlaubte Drehrichtungsumkehr nach HALT
8081	E 43 0	Endschalter: Negativer Sollwert gesperrt
8082	E 43 1	Endschalter: Positiver Sollwert gesperrt
8083	E 43 2	Endschalter: Positionierung unterdrückt
8084	E 45 0	Treiberversorgung nicht abschaltbar
8085	E 45 1	Treiberversorgung nicht aktivierbar
8086	E 45 2	Treiberversorgung wurde aktiviert
7580	E 60 0	Ethernet I
7581	E 61 0	Ethernet II
F080	E 80 0	Überlauf Stromregler- IRQ
F081	E 80 1	Überlauf Drehzahlregler- IRQ
F082	E 80 2	Überlauf Lageregler- IRQ
F083	E 80 3	Überlauf Interpolator- IRQ
F084	E 81 4	Überlauf Low Level- IRQ
F085	E 81 5	Überlauf MDC- IRQ
5080	E 90 x	Hardwarefehler
6000	E 91 0	Interner Initialisierungsfehler

5.5.3 Beschreibung der Objekte

5.5.3.1 Objekt 1003_h: pre_defined_error_field

Der jeweilige **error_code** der Fehlermeldungen wird zusätzlich in einem vierstufigen Fehlerspeicher abgelegt. Dieser ist wie ein Schieberegister strukturiert, so dass immer der zuletzt aufgetretene Fehler im Objekt **1003_h_01_h (standard_error_field_0)** abgelegt ist. Durch einen Lesezugriff auf das Objekt **1003_h_00_h (pre_defined_error_field)** kann festgestellt werden, wie viele Fehlermeldungen zur Zeit im Fehlerspeicher abgelegt sind. Der Fehlerspeicher wird durch das Einschreiben des Wertes 00_h in das Objekt **1003_h_00_h (pre_defined_error_field)** gelöscht. Um nach einem Fehler die Endstufe des Reglers wieder aktivieren zu können, muss zusätzlich eine **Fehlerquittierung** (siehe Abschnitt 7.1: Zustandsdiagramm (State Machine), ab Seite 154) durchgeführt werden.

Index	1003_h
Name	pre_defined_error_field
Object Code	ARRAY
No. of Elements	4
Data Type	UINT32

Sub-Index	01_h
Description	standard_error_field_0
Access	ro
PDO Mapping	no
Units	--
Value Range	--
Default Value	--

Sub-Index	02_h
Description	standard_error_field_1
Access	ro
PDO Mapping	no
Units	--
Value Range	--
Default Value	--

Sub-Index	03_h
Description	standard_error_field_2
Access	ro
PDO Mapping	no
Units	--
Value Range	--
Default Value	--

Sub-Index	04_h
Description	standard_error_field_3
Access	ro
PDO Mapping	no
Units	--
Value Range	--
Default Value	--

5.6 Netzwerkmanagement (NMT-Service)

Alle CANopen-Geräte können über das Netzwerkmanagement angesteuert werden. Hierfür ist der Identifier mit der höchsten Priorität (000_h) reserviert.

Mittels NMT können Befehle an einen oder alle Regler gesendet werden. Jeder Befehl besteht aus zwei Bytes, wobei das erste Byte den Befehlscode (command specifier, **CS**) und das zweite Byte die Knotenadresse (node id, **NI**) des angesprochenen Reglers beinhaltet. Über die Knotenadresse Null können gleichzeitig alle im Netzwerk befindlichen Knoten angesprochen werden. Es ist somit möglich, dass z.B. in allen Geräten gleichzeitig ein Reset ausgelöst wird. Die Regler quittieren die NMT-Befehle nicht. Es kann nur indirekt (z.B. durch die Einschaltmeldung nach einem Reset) auf die erfolgreiche Durchführung geschlossen werden.

Aufbau der NMT-Nachricht:

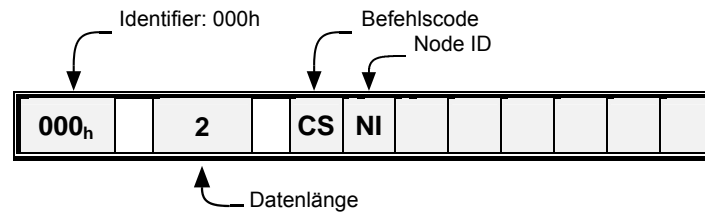
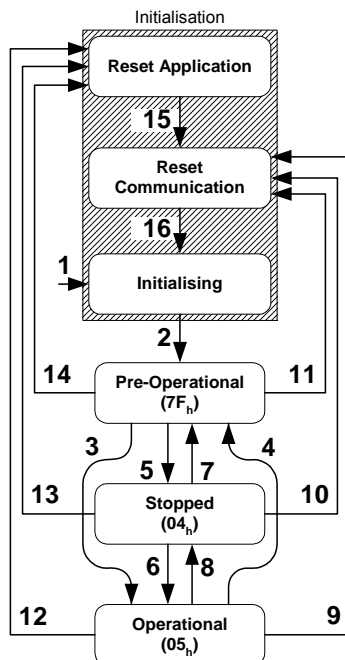


Abbildung 5.10: Aufbau der NMT-Nachricht

Für den NMT-Status des CANopen-Knotens sind Zustände in einem Zustandsdiagramm festgelegt. Über das Byte **CS** in der NMT-Nachricht können Zustandsänderungen ausgelöst werden. Diese sind im Wesentlichen am Ziel-Zustand orientiert.



	Bedeutung	CS	Ziel-Zustand	
2	Bootup	--	Pre-Operational	7F _h
3	Start Remote Node	01 _h	Operational	05 _h
4	Enter Pre-Operational	80 _h	Pre-Operational	7F _h
5	Stop Remote Node	02 _h	Stopped	04 _h
6	Start Remote Node	01 _h	Operational	05 _h
7	Enter Pre-Operational	80 _h	Pre-Operational	7F _h
8	Stop Remote Node	02 _h	Stopped	04 _h
9	Reset Communication	82 _h	Reset Communication ^{*1)}	
10	Reset Communication	82 _h	Reset Communication ^{*1)}	
11	Reset Communication	82 _h	Reset Communication ^{*1)}	
12	Reset Application	81 _h	Reset Application ^{*1)}	
13	Reset Application	81 _h	Reset Application ^{*1)}	
14	Reset Application	81 _h	Reset Application ^{*1)}	

^{*1)} Endgültiger Zielzustand ist Pre-Operational (7F_h), da die Übergänge 15, 16 und 2 vom Regler automatisch durchgeführt werden.

Abbildung 5.11: NMT-State Machine

Alle anderen Zustands-Übergänge werden vom Regler selbsttätig ausgeführt, z.B. weil die Initialisierung abgeschlossen ist.

Im Parameter **NI** muss die Knotennummer des Reglers angegeben werden oder Null, wenn alle im Netzwerk befindlichen Knoten adressiert werden sollen (Broadcast). Je nach NMT-Status können bestimmte Kommunikationsobjekte nicht benutzt werden: So ist es z.B. unbedingt notwendig den NMT-Status auf **Operational** zu stellen, damit der Regler PDOs sendet.

Name	Bedeutung	SDO	PDO	NMT
Reset Application	Keine Kommunikation. Alle CAN-Objekte werden auf ihre Resetwerte (Applikations-Parametersatz) zurückgesetzt	-	-	-
Reset Communication	Keine Kommunikation Der CAN-Controller wird neu initialisiert.	-	-	-
Initialising	Zustand nach Hardware-Reset. Zurücksetzen des CAN-Knotens, Senden der Bootup-Message	-	-	-
Pre-Operational	Kommunikation über SDOs möglich PDOs nicht aktiv (Kein Senden / Auswerten)	X	-	X
Operational	Kommunikation über SDOs möglich Alle PDOs aktiv (Senden / Auswerten)	X	X	X
Stopped	Keine Kommunikation außer Heartbeating	-	-	X



NMT- Telegramme dürfen nicht in einem Burst (unmittelbar hintereinander) gesendet werden!
Zwischen zwei aufeinander folgenden NMT-Nachrichten auf dem Bus (auch für verschiedene Knoten!) muss mindestens die doppelte Lagereglerzykluszeit liegen, damit der Regler die NMT-Nachrichten korrekt verarbeitet.



Der NMT Befehl „Reset Application“ wird gegebenenfalls so lange verzögert, bis ein laufender Speichervorgang abgeschlossen ist, da ansonsten der Speichervorgang unvollständig bleiben würde (Defekter Parametersatz). Die Verzögerung kann im Bereich einiger Sekunden liegen.



Der Kommunikationsstatus muss auf **operational** eingestellt werden, damit der Regler PDOs sendet und empfängt.

5.7 Bootup

5.7.1 Übersicht

Nach dem Einschalten der Spannungsversorgung oder nach einem Reset, meldet der Regler über eine Bootup-Nachricht, dass die Initialisierungsphase beendet ist. Der Regler ist dann im NMT-Status **preoperational**.

5.7.2 Aufbau der Bootup-Nachricht

Die Bootup-Nachricht ist nahezu identisch zur folgenden Heartbeat-Nachricht aufgebaut. Lediglich wird statt des NMT-Status eine Null gesendet.

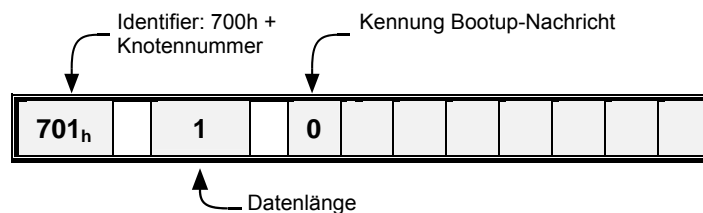


Abbildung 5.12: Aufbau der Bootup-Nachricht

5.8 Heartbeat (Error Control Protocol)

5.8.1 Übersicht

Zur Überwachung der Kommunikation zwischen Slave (Antrieb) und Master kann das sogenannte Heartbeat-Protokoll aktiviert werden: Hierbei sendet der Antrieb zyklisch Nachrichten an den Master. Der Master kann das zyklische Auftreten dieser Nachrichten überprüfen und entsprechende Maßnahmen einleiten, wenn diese ausbleiben. Da sowohl Heartbeat- als auch Nodeguarding-Telegramme (siehe Abschnitt 5.9: *Nodeguarding (Error Control Protocol)*, ab Seite 57) mit dem Identifier **700_h + Knotennummer** gesendet werden, können nicht beide Protokolle gleichzeitig aktiv sein. Werden beide Protokolle gleichzeitig aktiviert, ist nur das Heartbeat-Protokoll aktiv.

5.8.2 Aufbau der Heartbeat-Nachricht

Das Heartbeat-Telegramm wird mit dem Identifier **700_h + Knotennummer** gesendet. Es enthält nur 1 Byte Nutzdaten, den NMT-Status des Reglers (siehe Abschnitt 5.8: *Heartbeat (Error Control Protocol)*, ab Seite 54).

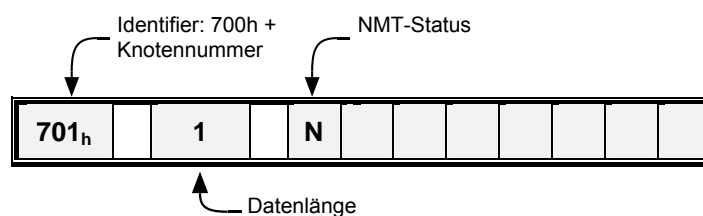


Abbildung 5.13: Aufbau der Heartbeat-Nachricht

N	Bedeutung
04 _h	Stopped
05 _h	Operational
7F _h	Pre-Operational

5.8.3 Beschreibung der Objekte

5.8.3.1 Objekt 1017_h: producer_heartbeat_time

Zur Aktivierung der Heartbeat- Funktionalität kann die Zeit zwischen zwei Heartbeat-Telegrammen über das Object **producer_heartbeat_time** festgelegt werden.

Index	1017_h
Name	producer_heartbeat_time
Object Code	VAR
Data Type	UINT16

Access	rw
PDO Mapping	no
Units	ms
Value Range	0...65535
Default Value	0

Die **producer_heartbeat_time** kann im Parametersatz gespeichert werden. Startet der Regler mit einer **producer_heartbeat_time** ungleich Null, gilt die Bootup-Nachricht als erstes Heartbeat.

Der Regler kann nur als sog. Heartbeat Producer verwendet werden. Das Objekt **1016_h** (**consumer_heartbeat_time**) ist daher nur aus Kompatibilitätsgründen implementiert und liefert immer 0 zurück.

5.9 Nodeguarding (Error Control Protocol)

5.9.1 Übersicht

Ebenfalls zur Überwachung der Kommunikation zwischen Slave (Antrieb) und Master kann das sogenannte Nodeguarding-Protokoll verwendet werden. Im Gegensatz zum Heartbeat-Protokoll überwachen sich hierbei Master und Slave gegenseitig:

Der Master fragt den Antrieb zyklisch nach seinem NMT-Status. Dabei wird in jeder Antwort des Reglers ein bestimmtes Bit invertiert (getoggelt). Bleiben diese Antworten aus oder antwortet der Regler immer mit dem gleichen Togglebit kann der Master entsprechend reagieren. Ebenso überwacht der Antrieb das regelmäßige Eintreffen der Nodeguarding-Anfragen des Masters: Bleiben die Nachrichten über einen bestimmten Zeitraum aus, löst der Regler Fehler 12-4 aus. Da sowohl Heartbeat- als auch Nodeguarding-Telegramme (siehe Abschnitt 5.9: *Nodeguarding (Error Control Protocol)*, ab Seite 57) mit dem Identifier **700_h + Knotennummer** gesendet werden, können nicht beide Protokolle gleichzeitig aktiv sein. Werden beide Protokolle gleichzeitig aktiviert, ist nur das Heartbeat-Protokoll aktiv. Nodeguarding ist ab Firmware 3.5.x.1.1 verfügbar.

5.9.2 Aufbau der Nodeguarding-Nachrichten

Die Anfrage des Masters muss als sog. Remoteframe mit dem Identifier **700_h + Knotennummer** gesendet werden. Bei einem Remoteframe ist zusätzlich ein spezielles Bit im Telegramm gesetzt, das Remotebit. Remoteframes haben grundsätzlich keine Daten.

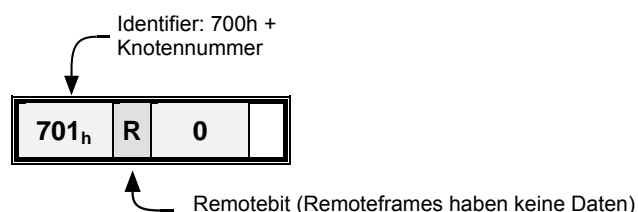


Abbildung 5.14: Aufbau der Nodeguarding-Nachrichten

Die Antwort des Reglers ist analog zur Heartbeat-Nachricht aufgebaut. Sie enthält nur 1 Byte Nutzdaten, das Togglebit und den NMT-Status des Reglers.

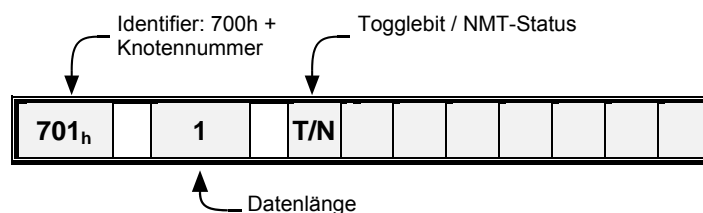


Abbildung 5.15: Aufbau der Nodeguarding-Nachrichten (Antwort Regler)

Das erste Datenbyte (**T/N**) ist folgendermaßen aufgebaut:

Bit	Wert	Name	Bedeutung
7	80 _h	toggle_bit	Ändert sich mit jedem Telegramm
0...6	7F _h	nmt_state	04_h Stopped 05_h Operational 7F_h Pre-Operational

Die Überwachungszeit für Anfragen des Masters ist parametrierbar. Die Überwachung beginnt mit der ersten empfangenen Remoteabfrage des Masters. Ab diesem Zeitpunkt müssen die Remoteabfragen vor Ablauf der eingestellten Überwachungszeit eintreffen, da anderenfalls Fehler 12-4 ausgelöst wird.

Das Togglebit wird durch das NMT-Kommando **Reset Communication** zurückgesetzt. Es ist daher in der ersten Antwort des Reglers gelöscht.

5.9.3 Beschreibung der Objekte

5.9.3.1 Objekt 100C_h: guard_time

Zur Aktivierung der Nodeguarding- Überwachung wird die Maximalzeit zwischen zwei Remoteabfragen des Masters parametrisiert. Diese Zeit wird im Regler aus dem Produkt von **guard_time** (100C_h) und **life_time_factor** (100D_h) bestimmt. Es empfiehlt sich daher den **life_time_factor** mit 1 zu beschreiben und die Zeit dann direkt über die **guard_time** in Millisekunden vorzugeben.

Index	100C_h
Name	guard_time
Object Code	VAR
Data Type	UINT16

Ab Firmware 3.5.x.1.1

Access	rw
PDO Mapping	no
Units	ms
Value Range	0...65535
Default Value	0

5.9.3.2 Objekt 100D_h: life_time_factor

Der **life_time_factor** sollte mit 1 beschrieben werden um die **guard_time** direkt vorzugeben.

Index	100D_h
Name	life_time_factor
Object Code	VAR
Data Type	UINT8

Ab Firmware 3.5.x.1.1

Access	rw
PDO Mapping	no
Units	--
Value Range	0, 1
Default Value	0

5.10 Tabelle der Identifier

Die folgende Tabelle gibt eine Übersicht über die verwendeten Identifier:

Objekt-Typ	Identifier (hexadezimal)	Bemerkung
SDO (Host an Regler)	600_h + Knotennummer	
SDO (Regler an Host)	580_h + Knotennummer	
TPDO1	181_h	Standardwerte. Können bei Bedarf geändert werden.
TPDO2	281_h	
TPDO3	381_h	
TPDO4	481_h	
RPDO1	201_h	
RPDO2	301_h	
RPDO3	401_h	
RPDO4	501_h	
SYNC	080_h	
EMCY	080_h + Knotennummer	
HEARTBEAT	700_h + Knotennummer	
NODEGUARDING	700_h + Knotennummer	
BOOTUP	700_h + Knotennummer	
NMT	000_h	

6 Parameter einstellen

Bevor der Servoregler die gewünschte Aufgabe (Momenten-, Drehzahlregelung, Positionierung) ausführen kann, müssen zahlreiche Parameter des Reglers an den verwendeten Motor und die spezifische Applikation angepasst werden. Dabei sollte in der Reihenfolge der anschließenden Kapitel vorgegangen werden. Im Anschluss an die Einstellung der Parameter wird die Gerätesteuerung und die Nutzung der jeweiligen Betriebsarten erläutert.



Das Display des Reglers zeigt ein „A“ (Attention) an, wenn der Regler noch nicht geeignet parametrierung wurde. Soll der Regler komplett über CANopen parametrierung werden, müssen Sie das Objekt 6510_h_C0_h beschreiben, um diese Anzeige zu unterdrücken. (Siehe Abschnitt 6.17.1.12: *Objekt 6510_h_C0_h: commissioning_state*, Seite 150)

Neben den hier ausführlich beschriebenen Parametern sind im Objektverzeichnis des Reglers weitere Parameter vorhanden, die gemäß CANopen implementiert werden müssen. Sie enthalten aber in der Regel keine Informationen, die beim Aufbau einer Applikation mit der SERVOTEC S2 Familie sinnvoll verwendet werden kann. Bei Bedarf ist die Spezifikation solcher Objekte in den CAN in Automation-Dokumenten (siehe Abschnitt 1.2: *CANopen*, Seite 13) nachzulesen.

6.1 Parametersätze laden und speichern

6.1.1 Übersicht

Der Regler verfügt über drei Parametersätze:

- **Aktueller Parametersatz**
Dieser Parametersatz befindet sich im flüchtigen Speicher (RAM) des Reglers. Er kann mit dem Parametrierprogramm MSC oder über den CAN-Bus beliebig gelesen und beschrieben werden. Beim Einschalten des Reglers wird der Applikations-Parametersatz in den aktuellen Parametersatz kopiert.
- **Default-Parametersatz**
Dieses ist der vom Hersteller standardmäßig vorgegebene unveränderliche Parametersatz des Antriebsreglers. Durch einen Schreibvorgang in das CANopen-Objekt 1011_h_01_h (restore_all_default_parameters) kann der Default-Parametersatz in den aktuellen Parametersatz kopiert werden. Dieser Kopiervorgang ist nur bei ausgeschalteter Endstufe möglich.
- **Applikations-Parametersatz**
Der aktuelle Parametersatz kann in den nichtflüchtigen Flash-Speicher gesichert werden. Der Speichervorgang wird mit einem Schreibzugriff auf das CANopen-Objekt 1010_h_01_h (save_all_parameters) ausgelöst. Beim Einschalten des Reglers wird automatisch der Applikations-Parametersatz in den aktuellen Parametersatz kopiert.

Die nachfolgende Grafik veranschaulicht die Zusammenhänge zwischen den einzelnen Parametersätzen.

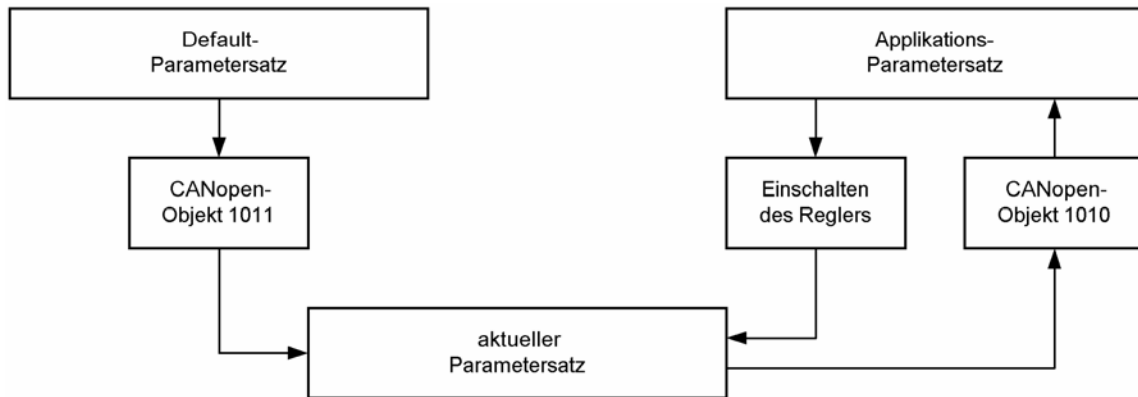


Abbildung 6.16: Default-/Applikations-Parametersatz

Es sind zwei unterschiedliche Konzepte zur Parametersatzverwaltung denkbar:

1. Der Parametersatz wird mit dem Parametrierprogramm S2Commander erstellt und ebenfalls mit dem S2Commander komplett in die einzelnen Regler übertragen. Bei diesem Verfahren müssen nur die ausschließlich via CANopen zugänglichen Objekte über den CAN-Bus eingestellt werden. **Nachteilig ist hierbei, dass für jede Inbetriebnahme einer neuen Maschine oder im Falle einer Reparatur (Regleraustausch) die Parametriersoftware benötigt wird. Dieses Verfahren ist daher nur bei Einzelstücken sinnvoll.**
2. Diese Variante basiert auf der Tatsache, dass die meisten applikationsspezifischen Parametersätze nur in wenigen Parametern vom **Default-Parametersatz** abweichen. Dadurch ist es möglich, dass der **aktuelle Parametersatz** nach jedem Einschalten der Anlage über den CAN-Bus neu aufgebaut wird. Hierzu wird von der übergeordneten Steuerung zunächst der **Default-Parametersatz** geladen (Aufruf des CANopen-Objekts **1011_h_01_h (restore_all_default_parameters)**). Danach werden nur die abweichenden Objekte übertragen. Der gesamte Vorgang dauert pro Regler unter 1 Sekunde. Vorteilhaft ist, dass dieses Verfahren auch bei unparametrierten Reglern funktioniert, so dass die Inbetriebnahme von neuen Anlagen oder der Austausch einzelner Regler unproblematisch ist und die Parametriersoftware S2Commander hierfür nicht benötigt wird. Die Verwendung dieser Methode wird empfohlen.



Stellen Sie vor dem allerersten Einschalten der Endstufe sicher, dass der Regler wirklich die von Ihnen gewünschten Parameter enthält.

Ein falsch parametrierter Regler kann unkontrolliert drehen und Personen- oder Sachschäden verursachen.

6.1.2 Beschreibung der Objekte

6.1.2.1 Objekt 1011_h: restore_default_parameters

Index	1011 _h
Name	restore_parameters
Object Code	ARRAY
No. of Elements	1
Data Type	UINT32

Sub-Index	01 _h
Description	restore_all_default_parameters
Access	rw
PDO Mapping	no
Units	--
Value Range	64616F6C _h („load“)
Default Value	1 (read access)

Das Objekt 1011_h_01_h (restore_all_default_parameters) ermöglicht, den **aktuellen Parametersatz** in einen definierten Zustand zu versetzen. Hierfür wird der **Default-Parametersatz** in den **aktuellen Parametersatz** kopiert. Der Kopiervorgang wird durch einen Schreibzugriff auf dieses Objekt ausgelöst, wobei als Datensatz der String „load“ in hexadezimaler Form zu übergeben ist.

Dieser Befehl wird nur bei deaktivierter Endstufe ausgeführt. Andernfalls wird der SDO-Fehler „Daten können nicht übertragen oder gespeichert werden, da sich der Regler dafür nicht im richtigen Zustand befindet“ erzeugt. Wird die falsche Kennung gesendet, wird der Fehler „Daten können nicht übertragen oder gespeichert werden“ erzeugt. Wird lesend auf das Objekt zugegriffen, wird eine 1 zurückgegeben, um anzuzeigen, dass das Zurücksetzen auf Defaultwerte unterstützt wird.

Die Parameter der CAN-Kommunikation (Knoten-Nr., Baudrate und Betriebsart) sowie zahlreiche Winkelgeber- Einstellungen (die zum Teil einen Reset erfordern um wirksam zu werden) bleiben hierbei unverändert.

6.1.2.2 Objekt 1010_h: store_parameters

Index	1010_h
Name	store_parameters
Object Code	ARRAY
No. of Elements	1
Data Type	UINT32

Sub-Index	01_h
Description	save_all_parameters
Access	rw
PDO Mapping	no
Units	--
Value Range	65766173 _h („save“)
Default Value	1

Soll der Default-Parametersatz auch in den Applikations-Parametersatz übernommen werden, dann muss außerdem auch das Objekt 1010_h_01_h (save_all_parameters) aufgerufen werden.

Wird das Objekt über ein SDO geschrieben, ist das Defaultverhalten, dass das SDO sofort beantwortet wird. Die Antwort spiegelt somit nicht das Ende des Speichervorgangs wider.

Das Verhalten kann jedoch über das Objekt **6510_h_F0_h (compatibility_control)** geändert werden.

6.2 Kompatibilitäts-Einstellungen

6.2.1 Übersicht

Um einerseits kompatibel zu früheren CANopen-Implementationen (z.B. auch in anderen Gerätefamilien) bleiben zu können und andererseits Änderungen und Korrekturen gegenüber der DSP402 und der DS301 ausführen zu können, wurde das Objekt **compatibility_control** eingefügt. Im Defaultparametersatz liefert dieses Objekt 0, d.h. Kompatibilität zu früheren Versionen. Für neue Applikationen empfehlen wir, die definierten Bits zu setzen, um so eine möglichst hohe Übereinstimmung mit den genannten Standards zu ermöglichen.

6.2.2 Beschreibung der Objekte

6.2.2.1 In diesem Kapitel behandelte Objekte

Index	Objekt	Name	Typ	Attr.
6510 _h _F0 _h	VAR	compatibility_control	UINT16	rw

6.2.2.2 Objekt 6510_h_F0_h: compatibility_control

Sub-Index	F0 _h
Description	compatibility_control
Data Type	UINT16
Access	rw
PDO Mapping	no
Units	--
Value Range	0...1FF _h , siehe Tabelle
Default Value	0

Ab Firmware 3.2.0.1.1

Bit	Wert	Name	
0	0001 _h	homing_method_scheme*	
1	0002 _h	reserved	
2	0004 _h	homing_method_scheme	
3	0008 _h	reserved	
4	0010 _h	response_after_save	Ab Firmware 3.4.0.1.1
5	0020 _h	reserved	Ab Firmware 3.5.0.1.1
6	0040 _h	homing_to_zero	Ab Firmware 3.5.0.1.1
7	0080 _h	device_control	Ab Firmware 3.5.0.1.1
8	0100 _h	reserved	Ab Firmware 3.5.0.1.1

Bit 0 homing_method_scheme*

Das Bit hat die gleiche Bedeutung wie Bit 2 und ist aus Kompatibilitätsgründen vorhanden. Wird Bit 2 gesetzt, wird dieses Bit auch gesetzt und umgekehrt.

Bit 1 reserved

Das Bit ist reserviert. Es darf nicht gesetzt werden.

Bit 2 homing_method_scheme

Wenn dieses Bit gesetzt ist, sind die Referenzfahrtmethoden 32... 35 gemäß DSP402 nummeriert, anderenfalls ist die Nummerierung kompatibel zu früheren Implementierungen.
(Siehe auch Abschnitt 8.2.3: *Referenzfahrt-Abläufe*, ab Seite 184). Wird dieses Bit gesetzt, wird auch Bit 0 gesetzt und umgekehrt

Bit 3 reserved

Das Bit ist reserviert. Es darf nicht gesetzt werden.

Bit 4 response_after_save

Wenn dieses Bit gesetzt ist, wird die Antwort auf **save_all_parameters** erst gesendet, wenn das Speichern abgeschlossen wurde. Dies kann mehrere Sekunden dauern, was ggf. zu einem Timeout in der Steuerung führt.
Ist das Bit gelöscht, wird sofort geantwortet, es ist allerdings zu berücksichtigen, dass der Speichervorgang noch nicht abgeschlossen ist.

Bit 5 reserved

Das Bit ist reserviert. Es darf nicht gesetzt werden.

Bit 6 homing_to_zero

Bisher besteht eine Referenzfahrt unter CANopen nur aus 2 Phasen (Suchfahrt und Kriechfahrt). Der Antrieb fährt anschließend nicht auf die ermittelte Nullposition (die z.B. durch den **homing_offset** zur gefundenen Referenzposition verschoben sein kann).

Wird dieses Bit gesetzt, wird dieses Standardverhalten geändert und der Antrieb schließt der Referenzfahrt eine Fahrt auf Null an. Siehe hierzu Abschnitt 8.2: *Betriebsart Referenzfahrt (Homing Mode)*, ab Seite 179.

Bit 7 device_control

Wenn dieses Bit gesetzt ist, wird Bit 4 des **statusword** (**voltage_enabled**) gemäß DSP 402 v2.0 ausgegeben.

Außerdem ist der Zustand **FAULT_REACTION_ACTIVE** vom Zustand **FAULT** unterscheidbar.

Siehe hierzu Abschnitt 7: *Gerätesteuerung (Device Control)*, ab Seite 154.

Bit 8 reserved

Das Bit ist reserviert. Es darf nicht gesetzt werden.

6.3 Umrechnungsfaktoren (Factor Group)

6.3.1 Übersicht

Servoregler werden in einer Vielzahl von Anwendungsfällen eingesetzt: Als Direktantrieb, mit nachgeschaltetem Getriebe, für Linearantriebe etc. Um für alle diese Anwendungsfälle eine einfache Parametrierung zu ermöglichen, kann der Regler mit Hilfe der Factor Group so parametrierbar werden, dass der Nutzer alle Größen wie z.B. die Drehzahl direkt in den gewünschten Einheiten am Abtrieb angeben bzw. auslesen kann (z.B. bei einer Linearachse Positionswerte in Millimeter und Geschwindigkeiten in Millimeter pro Sekunde). Der Regler rechnet die Eingaben dann mit Hilfe der Factor Group in seine internen Einheiten um. Für jede physikalische Größe (Position, Geschwindigkeit und Beschleunigung) ist ein Umrechnungsfaktor vorhanden, um die Nutzer-Einheiten an die eigene Applikation anzupassen. Die durch die Factor Group eingestellten Einheiten werden allgemein als **position_units**, **speed_units** oder **acceleration_units** bezeichnet. Die folgende Skizze verdeutlicht die Funktion der Factor Group:

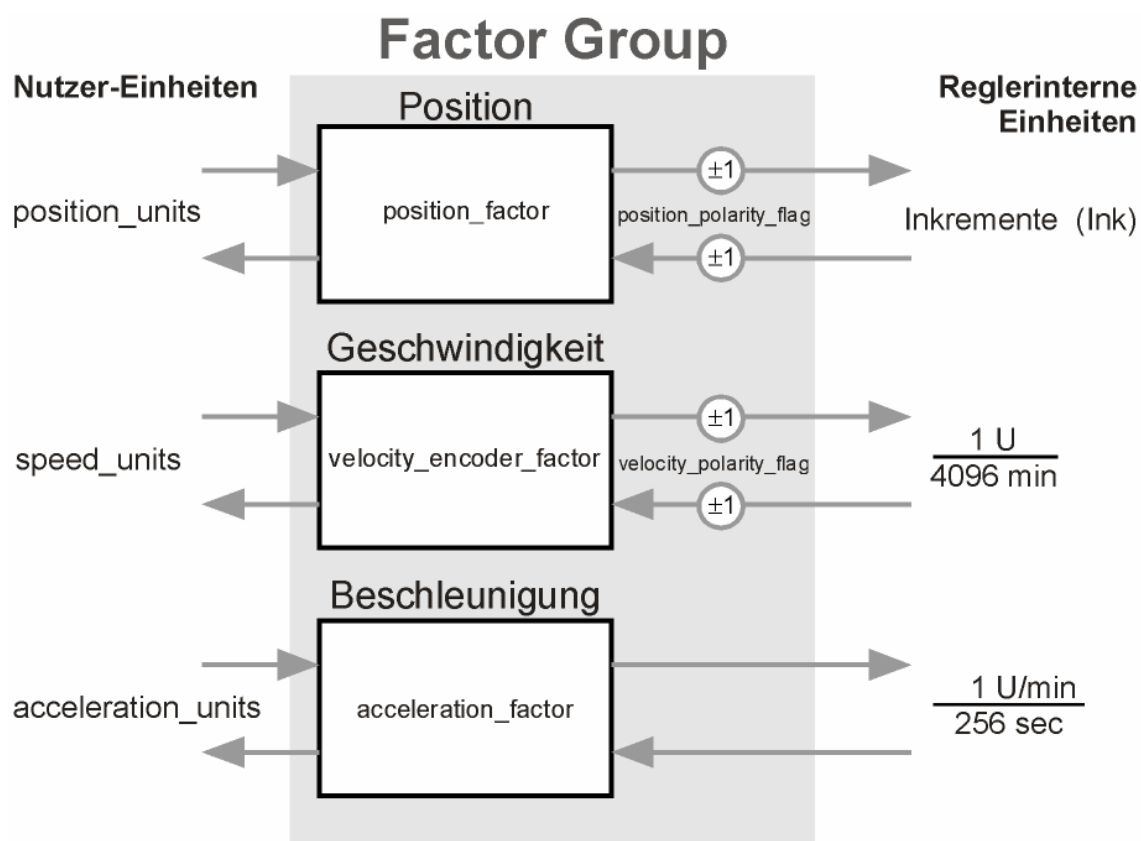


Abbildung 6.17: Umrechnungsfaktoren (Factor Group)

Alle Parameter werden im Regler grundsätzlich in seinen internen Einheiten gespeichert und erst beim Einschreiben oder Auslesen mit Hilfe der Factor Group umgerechnet.

Daher sollte die Factor Group vor der allerersten Parametrierung eingestellt werden und während einer Parametrierung nicht geändert werden.

Standardmäßig ist die Factor Group auf folgende Einheiten eingestellt:

Größe	Bezeichnung	Einheit	Erklärung
Länge	position_units	Inkrement	65536 Inkremente pro Umdrehung
Geschwindigkeit	speed_units	min⁻¹	Umdrehungen pro Minute
Beschleunigung	acceleration_units	(min⁻¹)/s	Drehzahlerhöhung pro Sekunde

6.3.2 Beschreibung der Objekte

6.3.2.1 In diesem Kapitel behandelte Objekte

Index	Objekt	Name	Typ	Attr.
6093 _h	ARRAY	position_factor	UINT32	rw
6094 _h	ARRAY	velocity_encoder_factor	UINT32	rw
6097 _h	ARRAY	acceleration_factor	UINT32	rw
607E _h	VAR	polarity	UINT8	rw

6.3.2.2 Objekt 6093_h: position_factor

Das Objekt **position_factor** dient zur Umrechnung aller Längeneinheiten der Applikation von **position_units** in die interne Einheit **Inkrement** (65536 Inkremente entsprechen 1 Umdrehung). Es besteht aus Zähler und Nenner.

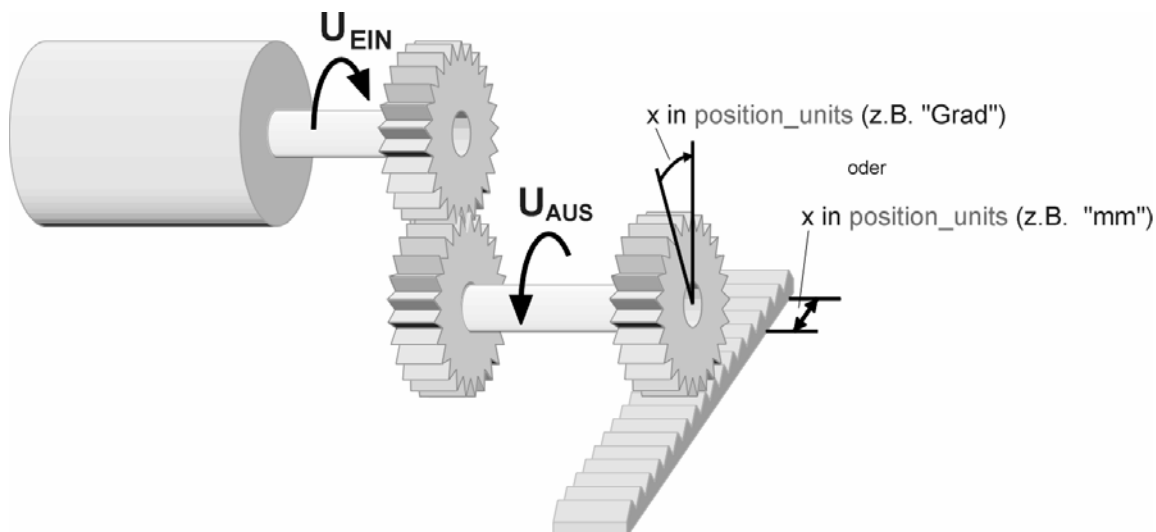


Abbildung 6.18: Übersicht: Factor Group

Index	6093_h
Name	position_factor
Object Code	ARRAY
No. of Elements	2
Data Type	UINT32

Sub-Index	01_h
Description	numerator
Access	rw
PDO Mapping	yes
Units	--
Value Range	--
Default Value	1

Sub-Index	02_h
Description	divisor
Access	rw
PDO Mapping	yes
Units	--
Value Range	--
Default Value	1

In die Berechnungsformel des **position_factor** gehen folgende Größen ein:

- gear_ratio** Getriebeverhältnis zwischen Umdrehungen am Eintrieb (U_{EIN}) und Umdrehungen am Abtrieb (U_{AUS})
- feed_constant** Verhältnis zwischen Umdrehungen am Abtrieb (U_{AUS}) und Bewegung in **position_units** (z.B. 1 U = 360° Grad)

Die Berechnung des **position_factor** erfolgt mit folgender Formel:

$$\text{position_factor} = \frac{\text{numerator}}{\text{divisor}} = \frac{\text{gear_ratio} \cdot 65536}{\text{feed_constant}}$$

Der **position_factor** muss getrennt nach Zähler und Nenner in den Regler geschrieben werden. Daher kann es notwendig sein, den Bruch durch geeignete Erweiterung auf ganze Zahlen zu bringen.



Der **position_factor** darf nicht größer als 2^{24} sein.

BEISPIEL



Zunächst muss die gewünschte Einheit (Spalte 1) und die gewünschten Nachkommastellen (NK) festgelegt, sowie der Getriebefaktor und ggf. die Vorschubkonstante der Applikation ermittelt werden. Diese Vorschubkonstante wird dann in den gewünschten Positions-Einheiten dargestellt (Spalte 2).

Letztlich können alle Werte in die Formel eingesetzt und der Bruch berechnet werden:

Grad, 1 NK 1/10 Grad (°/10)	$1 \text{ U}_{\text{AUS}} = 3600 \text{ } ^{\circ}/_{10}$	1/1	$\frac{\frac{1U}{1U} \cdot 65536 \frac{\text{Ink}}{U}}{\frac{3600 \text{ } ^{\circ}/_{10}}{1U}} = \frac{65536 \text{ Ink}}{3600 \text{ } ^{\circ}/_{10}}$	num: 4096 div: 225
--	---	-----	---	-----------------------

1. Gewünschte Einheit am Abtrieb (position_units)
2. feed_constant: Wie viel position_units sind 1 Umdrehung (U_{AUS})
3. Getriebefaktor (gear_ratio): U_{Ein} pro U_{AUS}
4. Werte in Formel einsetzen

1.	2.	3.	4.	ERGEBNIS Gekürzt
Inkremente, 0 NK Ink.	$1 \text{ U}_{\text{AUS}} = 65536 \text{ Ink}$	1/1	$\frac{\frac{1U}{1U} \cdot 65536 \frac{\text{Ink}}{U}}{\frac{65536 \text{ Ink}}{1U}} = \frac{1 \text{ Ink}}{1 \text{ Ink}}$	num: 1 div: 1
Grad, 1 NK 1/10 Grad (°/10)	$1 \text{ U}_{\text{AUS}} = 3600 \text{ } ^{\circ}/_{10}$	1/1	$\frac{\frac{1U}{1U} \cdot 65536 \frac{\text{Ink}}{U}}{\frac{3600 \text{ } ^{\circ}/_{10}}{1U}} = \frac{65536 \text{ Ink}}{3600 \text{ } ^{\circ}/_{10}}$	num: 4096 div: 225
Umdr., 2 NK 1/100 Umdr. (^U /100)	$1 \text{ U}_{\text{AUS}} = 100 \text{ } ^U/_{100}$	1/1	$\frac{\frac{1U}{1U} \cdot 65536 \frac{\text{Ink}}{U}}{\frac{100 \text{ } ^U/_{100}}{1U}} = \frac{65536 \text{ Ink}}{100 \text{ } ^U/_{100}}$	num: 16384 div: 25
	$63.15 \text{ mm}/^U \Rightarrow 1 \text{ U}_{\text{AUS}} = 631.5 \text{ mm}/_{10}$	2/3	$\frac{\frac{2U}{3U} \cdot 65536 \frac{\text{Ink}}{U}}{\frac{100 \text{ } ^U/_{100}}{1U}} = \frac{131072 \text{ Ink}}{300 \text{ } ^U/_{100}}$	num: 32768 div: 75
mm, 1 NK 1/10 mm (mm/10)	$63.15 \text{ mm}/^U \Rightarrow 1 \text{ U}_{\text{AUS}} = 631.5 \text{ mm}/_{10}$	4/5	$\frac{\frac{4U}{5U} \cdot 65536 \frac{\text{Ink}}{U}}{\frac{631.5 \text{ mm}/_{10}}{1U}} = \frac{2621440 \text{ Ink}}{31575 \text{ mm}/_{10}}$	num: 524288 div: 6315

6.3.2.3 Objekt 6094_h: velocity_encoder_factor

Das Objekt **velocity_encoder_factor** dient zur Umrechnung aller Geschwindigkeitswerte der Applikation von **speed_units** in die interne Einheit **Umdrehungen pro 4096 Minuten**. Es besteht aus Zähler und Nenner.

Index	6094_h
Name	velocity_encoder_factor
Object Code	ARRAY
No. of Elements	2
Data Type	UINT32

Sub-Index	01_h
Description	numerator
Access	rw
PDO Mapping	yes
Units	--
Value Range	--
Default Value	1000 _h

Sub-Index	02_h
Description	divisor
Access	rw
PDO Mapping	yes
Units	--
Value Range	--
Default Value	1

Die Berechnung des **velocity_encoder_factor** setzt sich im Prinzip aus zwei Teilen zusammen: Einem Umrechnungsfaktor von internen Längeneinheiten in **position_units** und einem Umrechnungsfaktor von internen Zeiteinheiten in benutzerdefinierte Zeiteinheiten (z.B. von Sekunden in Minuten). Der erste Teil entspricht der Berechnung des **position_factor** für den zweiten Teil kommt ein zusätzlicher Faktor zur Berechnung hinzu:

time_factor_v	Verhältnis zwischen interner Zeiteinheit und benutzerdefinierter Zeiteinheit. (z.B. 1 min = 1₄₀₉₆ 4096 min)
gear_ratio	Getriebeverhältnis zwischen Umdrehungen am Eintrieb (U _{EIN}) und Umdrehungen am Abtrieb (U _{AUS})
feed_constant	Verhältnis zwischen Umdrehungen am Abtrieb (U _{AUS}) und Bewegung in position_units (z.B. 1 U = 360° Grad)

Die Berechnung des **velocity_encoder_factor**s erfolgt mit folgender Formel:

$$\text{velocity_encoder_factor} = \frac{\text{numerator}}{\text{divisor}} = \frac{\text{gear_ratio} \cdot \text{time_factor_v}}{\text{feed_constant}}$$

Wie der **position_factor** wird auch der **velocity_encoder_factor** getrennt nach Zähler und Nenner in den Regler geschrieben werden. Daher kann es notwendig sein, den Bruch durch geeignete Erweiterung auf ganze Zahlen zu bringen.

BEISPIEL



Zunächst muss die gewünschte Einheit (Spalte 1) und die gewünschten Nachkommastellen (NK) festgelegt, sowie der Getriebefaktor und ggf. die Vorschubkonstante der Applikation ermittelt werden. Diese Vorschubkonstante wird dann in den gewünschten Positions-Einheiten dargestellt (Spalte 2). Anschließend wird die gewünschte Zeiteinheit in die Zeiteinheit des Servopositionierreglers umgerechnet (Spalte 3).

Letztlich können alle Werte in die Formel eingesetzt und der Bruch berechnet werden:

$\frac{\text{mm}}{\text{s}}$ 1 NK $\frac{1}{10} \frac{\text{mm}}{\text{s}}$ $(\frac{\text{mm}}{10\text{s}})$	$63.15 \frac{\text{mm}}{\text{U}} \Rightarrow$ $631.5 \frac{\text{mm}}{1}$	$\frac{1 \frac{1}{\text{s}}}{60 \frac{1}{\text{min}}} = \frac{1}{60}$ $\frac{4096 \cdot 60}{4096 \text{ min}}$	$\frac{4U \cdot 60 \cdot 4096 \frac{1}{4096 \text{ min}}}{5U \cdot 1 \frac{1}{\text{s}}} = \frac{1966080 \frac{\text{U}}{4096 \text{ min}}}{6315 \frac{\text{mm}}{10\text{s}}}$	num: 131072 div: 421
---	---	---	---	-------------------------

1. Gewünschte Einheit am Abtrieb (*speed_units*)
2. *feed_constant*: Wie viel *position_units* sind 1 Umdrehung (U_{AUS})?
3. *time_factor_v*: Gewünschte Zeiteinheit pro interner Zeiteinheit
4. Getriebefaktor (*gear_ratio*) U_{EIN} pro U_{AUS}
5. Werte in Formel einsetzen

1.	2.	3.	4.	5.	ERGEBNIS Gekürzt
$\frac{\text{U}}{\text{min}}$ 0 NK $\frac{\text{U}}{\text{min}}$	$1 U_{\text{AUS}} =$ $1 U_{\text{AUS}}$	$\frac{1 \frac{1}{\text{min}}}{4096 \frac{1}{4096 \text{ min}}}$	$1/1$	$\frac{1U \cdot 4096 \frac{1}{4096 \text{ min}}}{1U \cdot 1 \frac{1}{\text{min}}} = \frac{4096 \frac{\text{U}}{4096 \text{ min}}}{1 \frac{\text{U}}{\text{min}}}$	num: 4096 div: 1
$\frac{\text{U}}{\text{min}}$ 2 NK $\frac{1}{100} \frac{\text{U}}{\text{min}}$ $(\frac{\text{U}}{100 \text{ min}})$	$1 U_{\text{AUS}} =$ $100 \frac{\text{U}}{100}$	$\frac{1 \frac{1}{\text{min}}}{4096 \frac{1}{4096 \text{ min}}}$	$2/3$	$\frac{2U \cdot 4096 \frac{1}{4096 \text{ min}}}{3U \cdot 1 \frac{1}{\text{min}}} = \frac{8192 \frac{\text{U}}{4096 \text{ min}}}{300 \frac{\text{U}}{100 \text{ min}}}$	num: 2048 div: 75
$\frac{\text{°}}{\text{s}}$ 1 NK $\frac{1}{10} \frac{\text{°}}{\text{s}}$ $(\frac{\text{°}}{10\text{s}})$	$1 U_{\text{AUS}} =$ $3600 \frac{\text{°}}{10}$	$\frac{1 \frac{1}{\text{s}}}{60 \frac{1}{\text{min}}} = \frac{1}{60}$ $\frac{60 \cdot 4096}{4096 \text{ min}}$	$1/1$	$\frac{1U \cdot 60 \cdot 4096 \frac{1}{4096 \text{ min}}}{1U \cdot 1 \frac{1}{\text{min}}} = \frac{245760 \frac{\text{U}}{4096 \text{ min}}}{3600 \frac{\text{°}}{10\text{s}}}$	num: 1024 div: 15
$\frac{\text{mm}}{\text{s}}$ 1 NK $\frac{1}{10} \frac{\text{mm}}{\text{s}}$ $(\frac{\text{mm}}{10\text{s}})$	$63.15 \frac{\text{mm}}{\text{U}} \Rightarrow$ $1 U_{\text{AUS}} =$ $631.5 \frac{\text{mm}}{1}$	$\frac{1 \frac{1}{\text{s}}}{60 \frac{1}{\text{min}}} = \frac{1}{60}$ $\frac{60 \cdot 4096}{4096 \text{ min}}$	$4/5$	$\frac{4U \cdot 60 \cdot 4096 \frac{1}{4096 \text{ min}}}{5U \cdot 1 \frac{1}{\text{s}}} = \frac{1966080 \frac{\text{U}}{4096 \text{ min}}}{6315 \frac{\text{mm}}{10\text{s}}}$	num: 131072 div: 421

6.3.2.4 Objekt 6097_h: acceleration_factor

Das Objekt **acceleration_factor** dient zur Umrechnung aller Beschleunigungswerte der Applikation von **acceleration_units** in die interne Einheit **Umdrehungen pro Minute pro 256 Sekunden**. Es besteht aus Zähler und Nenner.

Index	6097_h
Name	acceleration_factor
Object Code	ARRAY
No. of Elements	2
Data Type	UINT32

Sub-Index	01_h
Description	numerator
Access	rw
PDO Mapping	yes
Units	--
Value Range	--
Default Value	100 _h

Sub-Index	02_h
Description	divisor
Access	rw
PDO Mapping	yes
Units	--
Value Range	--
Default Value	1

Die Berechnung des **acceleration_factor** setzt sich ebenfalls aus zwei Teilen zusammen: Einem Umrechnungsfaktor von internen Längeneinheiten in **position_units** und einem Umrechnungsfaktor von internen Zeiteinheiten zum Quadrat in benutzerdefinierte Zeiteinheiten zum Quadrat (z.B. von Sekunden² in Minuten²). Der erste Teil entspricht der Berechnung des **position_factor** für den zweiten Teil kommt ein zusätzlicher Faktor hinzu:

- time_factor_a** Verhältnis zwischen interner Zeiteinheit zum Quadrat und benutzerdefinierter Zeiteinheit zum Quadrat
(z.B. $1 \text{ min}^2 = 1 \text{ min} \cdot 1 \text{ min} = 60 \text{ s} \cdot 1 \text{ min} = \frac{60}{256} \text{ 256 min} \cdot \text{s}$)
- gear_ratio** Getriebeverhältnis zwischen Umdrehungen am Eintrieb (U_{EIN}) und Umdrehungen am Abtrieb (U_{AUS})
- feed_constant** Verhältnis zwischen Umdrehungen am Abtrieb (U_{AUS}) und Bewegung in **position_units** (z.B. 1 U = 360° Grad)

Die Berechnung des **acceleration_factor** erfolgt mit folgender Formel:

$$\text{acceleration_factor} = \frac{\text{numerator}}{\text{divisor}} = \frac{\text{gear_ratio} \cdot \text{time_factor_a}}{\text{feed_constant}}$$

Auch der **acceleration_factor** wird getrennt nach Zähler und Nenner in den Regler geschrieben werden, so dass eventuell erweitert werden muss.

BEISPIEL



Zunächst muss die gewünschte Einheit (Spalte 1) und die gewünschten Nachkommastellen (NK) festgelegt, sowie der Getriebefaktor und ggf. die Vorschubkonstante der Applikation ermittelt werden. Diese Vorschubkonstante wird dann in den gewünschten Positions-Einheiten dargestellt (Spalte 2). Anschließend wird die gewünschte Zeiteinheit² in die Zeiteinheit² des Servopositionierreglers umgerechnet (Spalte 3). Letztlich können alle Werte in die Formel eingesetzt und der Bruch berechnet werden:

$\frac{\text{mm}}{\text{s}^2}$ 1 NK $\frac{1}{10} \frac{\text{mm}}{\text{s}^2}$ ($\frac{\text{mm}}{10\text{s}^2}$)	$63.15 \frac{\text{mm}}{\text{U}} \Rightarrow$ $631.5 \frac{\text{mm}}{10}$	$1 \frac{1}{\text{s}^2} =$ $60 \frac{1}{\text{min} \cdot \text{s}} =$ $60 \cdot 256 \frac{\text{min}}{256 \cdot \text{s}}$	$4/5$	$\frac{4U \cdot 256 \cdot 60 \cdot \frac{1}{256} \frac{\text{min} \cdot \text{s}}{\text{s}^2}}{5U \cdot 1 \frac{1}{\text{s}^2}} = \frac{122880 \frac{U \cdot \text{min}}{256 \cdot \text{s}}}{631.5 \frac{\text{mm}}{10}} = \frac{122880}{6315} \frac{U \cdot \text{min}}{\text{mm} \cdot \text{s}^2}$	num: 8192 div: 421
---	--	--	-------	--	-----------------------

1. Gewünschte Einheit am Abtrieb (acceleration_units)
2. *feed_constant*: Wie viel *position_units* sind 1 Umdrehung (U_{AUS})?
3. *time_factor_a*: Gewünschte Zeiteinheit² pro interne Zeiteinheit²
4. Getriebefaktor (*gear_ratio*) U_{EIN} pro U_{AUS}
5. Werte in Formel einsetzen

1.	2.	3.	4.	5.	ERGEBNIS Gekürzt
$\frac{\text{U}}{\text{min} \cdot \text{s}}$ 0 NK $\frac{\text{U}}{\text{min} \cdot \text{s}}$	$1 U_{\text{AUS}} =$ $1 U_{\text{AUS}}$	$1 \frac{1}{\text{min} \cdot \text{s}} =$ $256 \frac{1}{256 \cdot \text{s}}$	$1/1$	$\frac{1U \cdot 256 \cdot \frac{1}{256} \frac{\text{min} \cdot \text{s}}{\text{s}^2}}{1U \cdot 1 \frac{1}{\text{min} \cdot \text{s}}} = \frac{256 \frac{U \cdot \text{min}}{256 \cdot \text{s}}}{1U} = \frac{256}{1} \frac{U \cdot \text{min}}{\text{U} \cdot \text{s}}$	num: 256 div: 1
$\frac{\circ}{\text{s}^2}$ 1 NK $\frac{1}{10} \frac{\circ}{\text{s}^2}$ ($\frac{\circ}{10\text{s}^2}$)	$1 U_{\text{AUS}} =$ $3600 \frac{\circ}{10}$	$1 \frac{1}{\text{s}^2} =$ $60 \frac{1}{\text{min} \cdot \text{s}} =$ $60 \cdot 256 \frac{1}{256 \cdot \text{s}}$	$1/1$	$\frac{1U \cdot 60 \cdot 256 \cdot \frac{1}{256} \frac{\text{min} \cdot \text{s}}{\text{s}^2}}{1U \cdot 1 \frac{1}{\text{s}^2}} = \frac{15360 \frac{U \cdot \text{min}}{256 \cdot \text{s}}}{3600 \frac{\circ}{10}} = \frac{15360}{3600} \frac{U \cdot \text{min}}{\circ \cdot \text{s}^2}$	num: 64 div: 15
$\frac{\text{U}}{\text{min}^2}$ 2 NK $\frac{1}{100} \frac{\text{U}}{\text{min}^2}$ ($\frac{\text{U}}{100 \text{min}^2}$)	$1 U_{\text{AUS}} =$ $100 \frac{\text{U}}{100}$	$1 \frac{1}{\text{min}^2} =$ $\frac{1}{60} \frac{1}{\text{s}} =$ $\frac{256}{60} \frac{1}{256 \cdot \text{s}}$	$2/3$	$\frac{2U \cdot 256 \cdot \frac{1}{256} \frac{\text{min} \cdot \text{s}}{\text{s}^2}}{3U \cdot 60 \frac{1}{\text{min}^2}} = \frac{512 \frac{U \cdot \text{min}}{256 \cdot \text{s}}}{18000 \frac{\text{U}}{100 \text{min}^2}} = \frac{512}{18000} \frac{U \cdot \text{min}}{\text{U} \cdot \text{min}^2}$	num: 32 div: 1125
$\frac{\text{mm}}{\text{s}^2}$ 1 NK $\frac{1}{10} \frac{\text{mm}}{\text{s}^2}$ ($\frac{\text{mm}}{10\text{s}^2}$)	$63.15 \frac{\text{mm}}{\text{U}} \Rightarrow$ $1 U_{\text{AUS}} =$ $631.5 \frac{\text{mm}}{10}$	$1 \frac{1}{\text{s}^2} =$ $60 \frac{1}{\text{min} \cdot \text{s}} =$ $60 \cdot 256 \frac{1}{256 \cdot \text{s}}$	$4/5$	$\frac{4U \cdot 60 \cdot 256 \cdot \frac{1}{256} \frac{\text{min} \cdot \text{s}}{\text{s}^2}}{5U \cdot 1 \frac{1}{\text{s}^2}} = \frac{122880 \frac{U \cdot \text{min}}{256 \cdot \text{s}}}{631.5 \frac{\text{mm}}{10}} = \frac{122880}{6315} \frac{U \cdot \text{min}}{\text{mm} \cdot \text{s}^2}$	num: 8192 div: 421

6.3.2.5 Objekt 607E_h: polarity

Das Vorzeichen der Positions- und Geschwindigkeitswerte des Reglers kann mit dem entsprechenden `polarity_flag` eingestellt werden. Dieses kann dazu dienen, die Drehrichtung des Motors bei gleichen Sollwerten zu invertieren.

In den meisten Applikationen ist es sinnvoll, das **position_polarity_flag** und das **velocity_polarity_flag** auf den gleichen Wert zu setzen.

Das Setzen des `polarity_flags` beeinflusst nur Parameter beim Lesen und beim Schreiben. Bereits im Regler vorhandene Parameter werden nicht verändert.

Index	607E _h
Name	polarity
Object Code	VAR
Data Type	UINT8

Access	rw
PDO Mapping	yes
Units	--
Value Range	40 _h , 80 _h , C0 _h
Default Value	0

Bit	Wert	Name	Bedeutung
6	40 _h	velocity_polarity_flag	0: multiply by 1 (default) 1: multiply by -1 (invers)
7	80 _h	position_polarity_flag	0: multiply by 1 (default) 1: multiply by -1 (invers)

6.4 Endstufenparameter

6.4.1 Übersicht

Die Netzspannung wird über eine Vorladeschaltung in die Endstufe eingespeist. Beim Einschalten der Leistungsversorgung wird der Einschaltstrom begrenzt und das Laden überwacht. Nach erfolgter Vorladung des Zwischenkreises wird die Ladeschaltung überbrückt. Dieser Zustand ist Voraussetzung für das Erteilen der Reglerfreigabe. Die gleichgerichtete Netzspannung wird mit den Kondensatoren des Zwischenkreises geglättet. Aus dem Zwischenkreis wird der Motor über die IGBTs gespeist. Die Endstufe enthält eine Reihe von Sicherheitsfunktionen, die zum Teil parametrierbar sind:

- Reglerfreigabelogik (Software- und Hardwarefreigabe)
- Über-/Unterspannungs-Überwachung des Zwischenkreises
- Überstromüberwachung
- Leistungsteilüberwachung



6.4.2 Beschreibung der Objekte

Index	Objekt	Name	Typ	Attr.
6510 _h	VAR	drive_data		

6.4.2.1 Objekt 6510_h_10_h: enable_logic

Damit die Endstufe des Antriebsreglers aktiviert werden kann, müssen die digitalen Eingänge **Endstufenfreigabe** und **Reglerfreigabe** gesetzt sein: Die **Endstufenfreigabe** wirkt direkt auf die Ansteuersignale der Leistungstransistoren und würde diese auch bei einem defekten Mikroprozessor unterbrechen können. Das Wegnehmen der Endstufenfreigabe bei laufendem Motor bewirkt somit, dass der Motor ungebremst austrudelt bzw. nur durch die eventuell vorhandene Haltebremse gestoppt wird. Die **Reglerfreigabe** wird vom Mikrokontroller des Reglers verarbeitet. Je nach Betriebsart reagiert der Regler nach der Wegnahme dieses Signals unterschiedlich:

- Positionierbetrieb und drehzahl geregelter Betrieb**
 Der Motor wird nach der Wegnahme des Signals mit einer definierten Bremsrampe abgebremst. Die Endstufe wird erst abgeschaltet, wenn die Motordrehzahl unterhalb 10 min^{-1} liegt und die eventuell vorhandene Haltebremse angezogen hat.
- Momentengeregelter Betrieb**
 Die Endstufe wird unmittelbar nach der Wegnahme des Signals abgeschaltet. Gleichzeitig wird eine eventuell vorhandene Haltebremse angezogen. Der Motor trudelt also ungebremst aus bzw. wird nur durch die eventuell vorhandene Haltebremse gestoppt

ACHTUNG !
Beide Signale garantieren nicht, dass der Motor spannungsfrei ist.

Beim Betrieb des Reglers über den CAN-Bus können die beiden digitalen Eingänge **Endstufenfreigabe** und **Reglerfreigabe** gemeinsam auf 24V gelegt und die Freigabe über den CAN-Bus gesteuert werden. Dazu muss das Objekt **6510_h_10_h (enable_logic)** auf zwei gesetzt werden. Aus Sicherheitsgründen erfolgt dies bei der Aktivierung von CANopen (auch nach einem Reset des Reglers) automatisch.

Index	6510_h
Name	drive_data
Object Code	RECORD
No. of Elements	51

Sub-Index	10_h
Description	enable_logic
Data Type	UINT16
Access	rw
PDO Mapping	no
Units	--
Value Range	0...2
Default Value	2

Wert	Bedeutung
0	Digitale Eingänge Endstufenfreigabe + Reglerfreigabe
1	Digitale Eingänge Endstufenfreigabe + Reglerfreigabe + RS232
2	Digitale Eingänge Endstufenfreigabe + Reglerfreigabe + CAN

6.4.2.2 Objekt 6510_h_30_h: pwm_frequency

Die Schaltverluste der Endstufe sind proportional zur Schaltfrequenz der Leistungs-transistoren. Aus einigen Geräten der SERVOTEC S2 -Familie kann durch Halbieren der normalen PWM-Frequenz etwas mehr Leistung entnommen werden. Dadurch steigt allerdings die durch die Endstufe verursachte Stromwelligkeit. Die Umschaltung ist nur bei ausgeschalteter Endstufe möglich.

Sub-Index	30_h
Description	pwm_frequency
Data Type	UINT16
Access	rw
PDO Mapping	no
Units	--
Value Range	0, 1
Default Value	0

Wert	Bedeutung
0	Normale Endstufenfrequenz
1	Halbe Endstufenfrequenz

6.4.2.3 Objekt 6510_h_3A_h: enable_enhanced_modulation

Mit dem Objekt **enable_enhanced_modulation** kann die erweiterte Sinusmodulation aktiviert werden. Sie erlaubt eine bessere Ausnutzung der Zwischenkreisspannung und damit um ca. 14% höhere Drehzahlen. Nachteilig ist in bestimmten Applikationen, dass das Regelverhalten und der Rundlauf des Motors bei sehr kleinen Drehzahlen geringfügig schlechter wird. Der Schreibzugriff ist nur bei ausgeschalteter Endstufe möglich.

Sub-Index	3A_h
Description	enable_enhanced_modulation
Data Type	UINT16
Access	rw
PDO Mapping	no
Units	--
Value Range	0, 1
Default Value	0

Wert	Bedeutung
0	Erweiterte Sinusmodulation AUS
1	Erweiterte Sinusmodulation EIN



Die Aktivierung der erweiterten Sinusmodulation wird erst nach einem Reset wirksam. Der Parametersatz muss somit zunächst gespeichert (**save_all_parameters**) und anschließend ein Reset durchgeführt werden.

6.4.2.4 Objekt 6510_h31_h: power_stage_temperature

Die Temperatur der Endstufe kann über das Objekt **power_stage_temperature** ausgelesen werden. Wenn die im Objekt 6510_h32_h (**max_power_stage_temperature**) angegebene Temperatur überschritten wird, schaltet die Endstufe aus und eine Fehlermeldung wird abgesetzt.

Sub-Index	31_h
Description	power_stage_temperature
Data Type	INT16
Access	ro
PDO Mapping	yes
Units	°C
Value Range	--
Default Value	--

6.4.2.5 Objekt 6510_h_32_h: max_power_stage_temperature

Die Temperatur der Endstufe kann über das Objekt 6510_h_31_h (**power_stage_temperature**) ausgelesen werden. Wenn die im Objekt **max_power_stage_temperature** angegebene Temperatur überschritten wird, schaltet die Endstufe aus und eine Fehlermeldung wird abgesetzt.

Sub-Index	32_h
Description	max_power_stage_temperature
Data Type	INT16
Access	ro
PDO Mapping	no
Units	°C
Value Range	100
Default Value	geräteabhängig

Gerätetyp	Wert
102	100°C
105	80°C
302	80°C
305	80°C
310	80°C

Gerätetyp	Wert
320	80°C
340	80°C

6.4.2.6 Objekt 6510_h_33_h: nominal_dc_link_circuit_voltage

Über das Objekt **nominal_dc_link_circuit_voltage** kann die Gerätenennspannung in Millivolt ausgelesen werden.

Sub-Index	33_h
Description	nominal_dc_link_circuit_voltage
Data Type	UINT32
Access	ro
PDO Mapping	no
Units	mV
Value Range	--
Default Value	geräteabhängig

Gerätetyp	Wert
102	360000
105	360000
302	560000
305	560000
310	560000

Gerätetyp	Wert
320	560000
340	560000

6.4.2.7 Objekt 6510_h_34_h: actual_dc_link_circuit_voltage

Über das Objekt **actual_dc_link_circuit_voltage** kann die aktuelle Spannung des Zwischenkreises in Millivolt ausgelesen werden.

Sub-Index	34_h
Description	actual_dc_link_circuit_voltage
Data Type	UINT32
Access	ro
PDO Mapping	yes
Units	mV
Value Range	--
Default Value	--

6.4.2.8 Objekt 6510_h_35_h: max_dc_link_circuit_voltage

Das Objekt **max_dc_link_circuit_voltage** gibt an, ab welcher Zwischenkreisspannung die Endstufe aus Sicherheitsgründen sofort ausgeschaltet und eine Fehlermeldung abgesetzt wird.

Sub-Index	35_h
Description	max_dc_link_circuit_voltage
Data Type	UINT32
Access	ro
PDO Mapping	no
Units	mV
Value Range	--
Default Value	geräteabhängig

Gerätetyp	Wert
102	460000
105	460000
302	800000
305	800000
310	800000

Gerätetyp	Wert
320	800000
340	800000

6.4.2.9 Objekt 6510h_36h: min_dc_link_circuit_voltage

Der Regler verfügt über eine Unterspannungsüberwachung. Diese kann über das Objekt **6510_h_37_h (enable_dc_link_undervoltage_error)** aktiviert werden. Das Objekt **6510_h_36_h (min_dc_link_circuit_voltage)** gibt an, bis zu welcher unteren Zwischenkreisspannung der Regler arbeiten soll. Unterhalb dieser Spannung wird der Fehler E 02 0 ausgelöst, wenn dieses mit dem nachfolgenden Objekt aktiviert wurde.

Sub-Index	36_h
Description	min_dc_link_circuit_voltage
Data Type	UINT32
Access	rw
PDO Mapping	no
Units	mV
Value Range	0...1000000
Default Value	0

6.4.2.10 Objekt 6510_h_37_h: enable_dc_link_undervoltage_error

Mit dem Objekt **enable_dc_link_undervoltage_error** kann die Unterspannungsüberwachung aktiviert werden. Im Objekt **6510_h_36_h (min_dc_link_circuit_voltage)** ist anzugeben, bis zu welcher unteren Zwischenkreisspannung der Regler arbeiten soll.

Sub-Index	37_h
Description	enable_dc_link_undervoltage_error
Data Type	UINT16
Access	rw
PDO Mapping	no
Units	--
Value Range	0, 1
Default Value	0

Wert	Bedeutung
0	Unterspannungsfehler AUS (Reaktion WARNUNG)
1	Unterspannungsfehler EIN (Reaktion REGLERFREIGABE AUS)

Die Aktivierung des Fehlers 02-0 erfolgt durch Änderung der Fehlerreaktion. Reaktionen, die zum Stillsetzen des Antriebs führen, werden als **EIN**, alle anderen als **AUS** zurückgegeben. Beim Beschreiben mit 0 wird die Fehlerreaktion WARNUNG gesetzt, beim Beschreiben mit 1 die Fehlerreaktion REGLERFREIGABE AUS.

Siehe hierzu auch Abschnitt 6.18: *Fehlermanagement*, ab Seite 151.

6.4.2.11 Objekt 6510_h_40_h: nominal_current

Mit dem Objekt **nominal_current** kann der Gerätenennstrom ausgelesen werden. Es handelt sich gleichzeitig um den oberen Grenzwert, der in das Objekt 6075_h (**motor_rated_current**) eingeschrieben werden kann.

Sub-Index	40 _h
Description	nominal_current
Data Type	UINT32
Access	ro
PDO Mapping	no
Units	mA
Value Range	--
Default Value	geräteabhängig

Gerätetyp	Wert
102	2500
105	5000
302	2500
305	5000
310	7127

Gerätetyp	Wert
320	17427
340	34672



Aufgrund eines Leistungsderating werden abhängig von der Reglerzykluszeit und der Endstufentaktfrequenz gegebenenfalls andere Werte angezeigt.

6.4.2.12 Objekt 6510_h_41_h: peak_current

Mit dem Objekt **peak_current** kann der Gerätespitzenstrom ausgelesen werden. Es handelt sich gleichzeitig um den oberen Grenzwert, der in das Objekt 6073_h (**max_current**) eingeschrieben werden kann.

Sub-Index	41 _h
Description	peak_current
Data Type	UINT32
Access	ro
PDO Mapping	no
Units	mA
Value Range	--
Default Value	geräteabhängig

Gerätetyp	Wert
102	5000
105	10000
302	7500
305	15000
310	14254

Gerätetyp	Wert
320	31461
340	53248



Aufgrund eines Leistungsderating werden abhängig von der Reglerzykluszeit und der Endstufentaktfrequenz gegebenenfalls andere Werte angezeigt.

6.5 Stromregler und Motoranpassung



Vorsicht !

Falsche Einstellungen der Stromreglerparameter und der Strombegrenzungen können den **Motor** und unter Umständen auch den **Servoregler** innerhalb kürzester Zeit **zerstören!**

6.5.1 Übersicht

Der Parametersatz des Servoreglers muss für den angeschlossenen Motor und den verwendeten Kabelsatz angepasst werden.

Betroffen sind folgende Parameter:

- Nennstrom Abhängig vom Motor
- Überlastbarkeit Abhängig vom Motor
- Polzahl Abhängig vom Motor
- Stromregler Abhängig vom Motor
- Drehsinn Abhängig vom Motor und der Phasenfolge im Motor- und Winkelgeberkabel
- Offsetwinkel Abhängig vom Motor und der Phasenfolge im Motor- und Winkelgeberkabel

Diese Daten müssen beim erstmaligen Einsatz eines Motortyps mit dem Programm S2Commander bestimmt werden. Für eine Reihe von Motoren können Sie auch fertige Parametersätze über Ihren Händler beziehen. Bitte beachten Sie, dass Drehsinn und Offsetwinkel auch vom verwendeten Kabelsatz abhängen. Die Parametersätze arbeiten daher nur bei identischer Verkabelung.



Bei verdrehter Phasenfolge im Motor- oder Winkelgeberkabel kann es zu einer Mitkopplung kommen, so dass die Drehzahl im Motor nicht geregelt werden kann. Der Motor kann unkontrolliert durchdrehen !

6.5.2 Beschreibung der Objekte

Index	Objekt	Name	Typ	Attr.
6075 _h	VAR	motorRatedCurrent	UINT32	rw
6073 _h	VAR	maxCurrent	UINT16	rw
604D _h	VAR	poleNumber	UINT8	rw
6410 _h	RECORD	motorData		rw
60F6 _h	RECORD	torqueControlParameters		rw

6.5.2.1 Objekt 6075_h: motorRatedCurrent

Dieser Wert ist dem Motortypenschild zu entnehmen und wird in der Einheit Milliampere eingegeben. Es wird immer der Effektivwert (RMS) angenommen. Es kann kein Strom vorgegeben werden, der oberhalb des Reglernennstromes (6510_h_40_h: nominalCurrent) liegt.

Index	6075 _h
Name	motorRatedCurrent
Object Code	VAR
Data Type	UINT32

Access	rw
PDO Mapping	yes
Units	mA
Value Range	0...nominalCurrent
Default Value	296



Wird das Objekt 6075_h (motorRatedCurrent) mit einem neuen Wert beschrieben, muss in jedem Fall auch das Objekt 6073_h (maxCurrent) neu parametrisiert werden.

6.5.2.2 Objekt 6073_h: max_current

Servomotoren dürfen in der Regel für einen bestimmten Zeitraum überlastet werden. Mit diesem Objekt wird der höchstzulässige Motorstrom eingestellt. Er bezieht sich auf den Motornennstrom (Objekt 6075_h: **motorRatedCurrent**) und wird in Tausendstel eingestellt. Der Wertebereich wird nach oben durch den maximalen Reglerstrom (Objekt 6510_h_41_h: **peakCurrent**) begrenzt. Viele Motoren dürfen kurzzeitig um den Faktor 2 überlastet werden. In diesem Fall ist in dieses Objekt der Wert einzuschreiben.



Das Objekt 6073_h (**max_current**) darf erst beschrieben werden, wenn zuvor das Objekt 6075_h (**motorRatedCurrent**) gültig beschrieben wurde.

Index	6073 _h
Name	max_current
Object Code	VAR
Data Type	UINT16

Access	rw
PDO Mapping	yes
Units	per thousands of rated current
Value Range	--
Default Value	2023

6.5.2.3 Objekt 604D_h: pole_number

Die Polzahl des Motors ist dem Motordatenblatt oder dem Parametrierprogramm S2Commander zu entnehmen. Die Polzahl ist immer geradzahlig. Oft wird statt der Polzahl die Polpaarzahl angegeben. Die Polzahl entspricht dann der doppelten Polpaarzahl.

Dieses Objekt wird durch **restore_default_parameters** nicht geändert.

Index	604D _h
Name	pole_number
Object Code	VAR
Data Type	UINT8

Access	rw
PDO Mapping	yes
Units	--
Value Range	2... 254
Default Value	4 (nach <i>INIT!</i>)

6.5.2.4 Objekt 6410_h_03_h: iit_time_motor

Servomotoren dürfen in der Regel für einen bestimmten Zeitraum überlastet werden. Über dieses Objekt wird angegeben, wie lange der angeschlossene Motor mit dem im Objekt 6073_h (**max_current**) angegebenen Strom bestromt werden darf. Nach Ablauf der IIT-Zeit wird der Strom zum Schutz des Motors automatisch auf den im Objekt 6075_h (**motorRatedCurrent**) angegebenen Wert begrenzt. Die Standardeinstellung liegt bei zwei Sekunden und trifft für die meisten Motoren zu.

Index	6410_h
Name	motor_data
Object Code	RECORD
No. of Elements	5

Sub-Index	03_h
Description	iit_time_motor
Data Type	UINT16
Access	rw
PDO Mapping	no
Units	ms
Value Range	0...10000
Default Value	2000

6.5.2.5 Objekt 6410_h_04_h: iit_ratio_motor

Über das Objekt kann **iit_ratio_motor** die aktuelle Auslastung der I²t-Begrenzung in Promille ausgelesen werden.

Sub-Index	04_h
Description	iit_ratio_motor
Data Type	UINT16
Access	ro
PDO Mapping	no
Units	promille
Value Range	--
Default Value	--

6.5.2.6 Objekt 6510_h_38_h: iit_error_enable

Über das Objekt **iit_error_enable** wird festgelegt, wie sich der Regler bei Auftreten der I²t-Begrenzung verhält. Entweder wird dieses nur im **statusword** angezeigt, oder es wird Fehler E 3 1 0 ausgelöst.

Index	6510_h
Name	drive_data
Object Code	RECORD
No. of Elements	51

Sub-Index	38_h
Description	iit_error_enable
Data Type	UINT16
Access	rw
PDO Mapping	no
Units	--
Value Range	0, 1
Default Value	0

Wert	Bedeutung
0	I ² t-Fehler AUS (Priorität WARNUNG)
1	I ² t-Fehler EIN (Priorität REGLERFREIGABE AUS)

Die Aktivierung des Fehlers 31-0 erfolgt durch Änderung der Fehlerreaktion. Reaktionen, die zum Stillsetzen des Antriebs führen, werden als **EIN**, alle anderen als **AUS** zurückgegeben. Beim Beschreiben mit 0 wird die Fehlerreaktion WARNUNG gesetzt, beim Beschreiben mit 1 die Fehlerreaktion REGLERFREIGABE AUS. Siehe Abschnitt 6.18: *Fehlermanagement*, Seite 151.

6.5.2.7 Objekt 6410_h_10_h: phase_order

In der Phasenfolge (**phase_order**) werden Verdrehungen zwischen Motorkabel und Winkelgeberkabel berücksichtigt. Sie kann dem Parametrierprogramm **S2Commander** entnommen werden. Eine Null entspricht „rechts“, eine Eins „links“.

Sub-Index	10_h
Description	phase_order
Data Type	INT16
Access	rw
PDO Mapping	yes
Units	--
Value Range	0, 1
Default Value	0

Wert	Bedeutung
0	Rechts
1	Links

6.5.2.8 Objekt 6410_h_11_h: **encoder_offset_angle**

Bei den verwendeten Servomotoren befinden sich Dauermagnete auf dem Rotor. Diese erzeugen ein magnetisches Feld, dessen Ausrichtung zum Stator von der Rotorlage abhängt. Für die elektronische Kommutierung muss der Regler das elektromagnetische Feld des Stators immer im richtigen Winkel zu diesem Permanentmagnetfeld einstellen. Er bestimmt hierzu laufend mit einem Winkelgeber (Resolver etc.) die Rotorlage.

Die Orientierung des Winkelgebers zum Dauermagnetfeld muss in das Objekt **encoder_offset_angle** eingetragen werden. Mit dem Parametrierprogramm S2Commander kann dieser Winkel bestimmt werden (Parameter / Geräteparameter / Winkelgeber-Einstellungen). Der mit dem S2Commander bestimmte Winkel liegt im Bereich von ±180°.

Er muss folgendermaßen umgerechnet werden:

$$\text{encoder_offset_angle} = \text{„Offsetwinkel des Winkelgebers“} \times \frac{32767}{180^\circ}$$

Dieses Objekt wird durch **restore_default_parameters** nicht geändert.

Index	6410_h
Name	motor_data
Object Code	RECORD
No. of Elements	5

Sub-Index	11_h
Description	encoder_offset_angle
Data Type	INT16
Access	rw
PDO Mapping	yes
Units	--
Value Range	-32767...32767
Default Value	E000 _h (-45°) (nach <i>INIT!</i>)

6.5.2.9 Objekt 6410_h_14_h: motor_temperature_sensor_polarity

Über dieses Objekt kann festgelegt werden, ob ein Öffner oder ein Schließer als digitaler Motortemperatur-Sensor verwendet wird.

Sub-Index	14_h
Description	motor_temperature_sensor_polarity
Data Type	INT16
Access	rw
PDO Mapping	yes
Units	--
Value Range	0, 1
Default Value	0

Ab Firmware 3.2.0.1.1

Wert	Bedeutung
0	Öffner
1	Schließer

6.5.2.10 Objekt 6510_h_2E_h: motor_temperature

Mit diesem Objekt kann die aktuelle Motortemperatur ausgelesen werden, falls ein analoger Temperatursensor angeschlossen ist. Anderenfalls ist das Objekt undefiniert.

Index	6510_h
Name	drive_data
Object Code	RECORD
No. of Elements	51

Sub-Index	2E_h
Description	motor_temperature
Data Type	INT16
Access	ro
PDO Mapping	yes
Units	°C
Value Range	--
Default Value	--

Ab Firmware 3.5.x.1.1

6.5.2.11 Objekt 6510_h_2F_h: max_motor_temperature

Wird die in diesem Objekt definierte Motortemperatur überschritten, erfolgt eine Reaktion gemäß Fehlermanagement (Fehler 3-0, Übertemperatur Motor analog). Ist eine Reaktion parametrierbar, die zum Stillsetzen des Antriebs führt, wird eine Emergency- Message gesendet. Zur Parametrierung des Fehlermanagements siehe Abschnitt 6.18: *Fehlermanagement*, ab Seite 151.

Sub-Index	2F_h
Description	max_motor_temperature
Data Type	INT16
Access	rw
PDO Mapping	no
Units	°C
Value Range	20...300
Default Value	100

Ab Firmware 3.5.x.1.1

6.5.2.12 Objekt 60F6_h: torque_control_parameters

Die Daten des Stromreglers müssen dem Parametrierprogramm S2Commander entnommen werden. Hierbei sind folgende Umrechnungen zu beachten:

Die Verstärkung des Stromreglers muss mit 256 multipliziert werden. Bei einer Verstärkung von 1.5 im Menü „Stromregler“ des Parametrierprogramms S2Commander ist in das Objekt **torque_control_gain** der Wert 384 = 180_h einzuschreiben.

Die Zeitkonstante des Stromreglers ist im Parametrierprogramm S2Commander in Millisekunden angegeben. Um diese Zeitkonstante in das Objekt **torque_control_time** übertragen zu können, muss sie zuvor in Mikrosekunden umgerechnet werden. Bei einer angegebenen Zeit von 0.6 Millisekunden ist entsprechend der Wert 600 in das Objekt **torque_control_time** einzutragen.

Index	60F6_h
Name	torque_control_parameters
Object Code	RECORD
No. of Elements	2

Sub-Index	01_h
Description	torque_control_gain
Data Type	UINT16
Access	rw
PDO Mapping	no
Units	256 = „1“
Value Range	0...32*256
Default Value	3*256 (768)

Sub-Index	02_h
Description	torque_control_time
Data Type	UINT16
Access	rw
PDO Mapping	no
Units	µs
Value Range	104... 64401
Default Value	1020

6.6 Drehzahlregler

6.6.1 Übersicht

Der Parametersatz des Servoreglers muss für die Applikation angepasst werden. Besonders die Verstärkung ist stark abhängig von eventuell an den Motor angekoppelten Massen. Die Daten müssen bei der Inbetriebnahme der Anlage mit Hilfe des Programms S2Commander optimal bestimmt werden.



Falsche Einstellungen der Drehzahlreglerparameter können zu starken Schwingungen führen und eventuell Teile der Anlage zerstören!

6.6.2 Beschreibung der Objekte

Index	Objekt	Name	Typ	Attr.
60F9 _h	RECORD	velocity_control_parameters		rw
2073 _h	VAR	velocity_display_filter_time	UINT32	rw

6.6.2.1 Objekt 60F9_h: velocity_control_parameters

Die Daten des Drehzahlreglers müssen dem Parametrierprogramm S2Commander entnommen werden. Hierbei sind folgende Umrechnungen zu beachten:

Die Verstärkung des Drehzahlreglers muss mit 256 multipliziert werden.

Bei einer Verstärkung von 1.5 im Menü „Drehzahlregler“ des Parametrierprogramms S2Commander ist in das Objekt **velocity_control_gain** der Wert 384 = 180_h einzuschreiben.

Die Zeitkonstante des Drehzahlreglers ist im Parametrierprogramm S2Commander in Millisekunden angegeben. Um diese Zeitkonstante in das Objekt **velocity_control_time** übertragen zu können, muss sie zuvor in Mikrosekunden umgerechnet werden. Bei einer angegebenen Zeit von 2.0 Millisekunden ist entsprechend der Wert 2000 in das Objekt **velocity_control_time** einzutragen.

Index	60F9_h
Name	velocity_control_parameter_set
Object Code	RECORD
No. of Elements	3

Sub-Index	01_h
Description	velocity_control_gain
Data Type	UINT16
Access	rw
PDO Mapping	no
Units	256 = Gain 1
Value Range	20...64*256 (16384)
Default Value	256

Sub-Index	02_h
Description	velocity_control_time
Data Type	UINT16
Access	rw
PDO Mapping	no
Units	μs
Value Range	1...32000
Default Value	2000

Sub-Index	04_h
Description	velocity_control_filter_time
Data Type	UINT16
Access	rw
PDO Mapping	no
Units	μs
Value Range	1...32000
Default Value	400

6.6.2.2 Objekt 2073_h: velocity_display_filter_time

Mit dem Objekt **velocity_display_filter_time** kann die Filterzeit des Anzeigedrehzahl-Istwertfilters eingestellt werden.

Index	2073_h
Name	velocity_display_filter_time
Object Code	VAR
Data Type	UINT32

Ab Firmware 3.5.x.1.1

Access	rw
PDO Mapping	no
Units	µs
Value Range	1000..50000
Default Value	20000



Bitte beachten Sie, dass das Objekt **velocity_actual_value_filtered** für den Durchdrehschutz verwendet wird. Bei sehr großer Filterzeit wird ein Durchdrehfehler erst mit entsprechender Verzögerung erkannt.

6.7 Lageregler (Position Control Function)

6.7.1 Übersicht

In diesem Kapitel sind alle Parameter beschrieben, die für den Lageregler erforderlich sind. Am Eingang des Lagereglers liegt der Lage-Sollwert (**position_demand_value**) vom Fahrkurven-Generator an. Außerdem wird der Lage-Istwert (**position_actual_value**) vom Winkelgeber (Resolver, Inkrementalgeber etc.) zugeführt. Das Verhalten des Lagereglers kann durch Parameter beeinflusst werden. Um den Lageregelkreis stabil zu halten, ist eine Begrenzung der Ausgangsgröße (**control_effort**) möglich. Die Ausgangsgröße wird als Drehzahl-Sollwert dem Drehzahlregler zugeführt. Alle Ein- und Ausgangsgrößen des Lagereglers werden in der **Factor Group** von den applikationsspezifischen Einheiten in die jeweiligen internen Einheiten des Reglers umgerechnet.

Folgende Unterfunktionen sind in diesem Kapitel definiert:

1. Schleppfehler (Following_Error)

Als Schleppfehler wird die Abweichung des Lage-Istwertes (**position_actual_value**) vom Lage-Sollwert (**position_demand_value**) bezeichnet. Wenn dieser Schleppfehler für einen bestimmten Zeitraum größer ist als im Schleppfehler-Fenster (**following_error_window**) angegeben, so wird das Bit 13 **following_error** im Objekt **statusword** gesetzt. Der zulässige Zeitraum kann über das Objekt **following_error_time_out** vorgegeben werden.

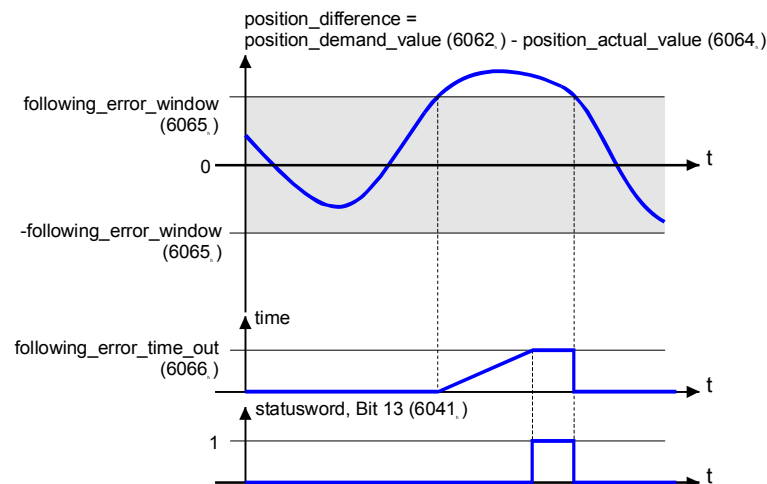


Abbildung 6.19: Schleppfehler – Funktionsübersicht

Die *Abbildung 6.20* zeigt, wie die Fensterfunktion für die Meldung „Schleppfehler“ definiert ist. Symmetrisch um die Sollposition (**position_demand_value**) x_i ist der Bereich zwischen x_i-x_0 und x_i+x_0 definiert. Die Positionen x_{i2} und x_{i3} liegen z.B. außerhalb dieses Fensters (**following_error_window**). Wenn der Antrieb dieses Fenster verlässt und nicht in der im Objekt **following_error_time_out** vorgegebenen Zeit in das Fenster zurückkehrt, dann wird das Bit 13 **following_error** im **statusword** gesetzt.

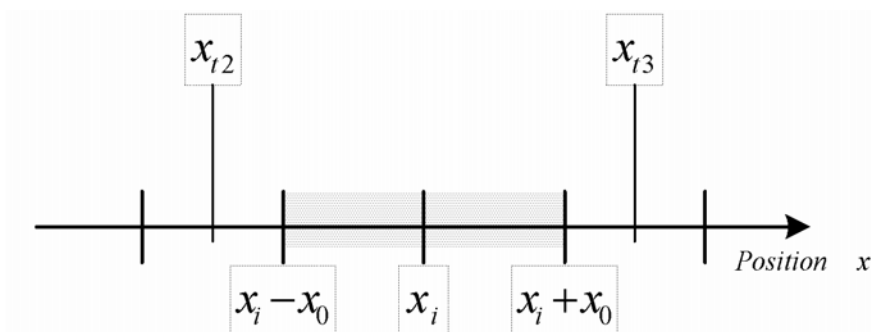


Abbildung 6.20: Schleppfehler

2. Position erreicht (Position Reached)

Diese Funktion bietet die Möglichkeit, ein Positionsfenster um die Zielposition (**target_position**) herum zu definieren. Wenn sich die Ist-Position des Antriebs für eine bestimmte Zeit – die **position_window_time** – in diesem Bereich befindet, dann wird das damit verbundene Bit 10 (**target_reached**) im **statusword** gesetzt.

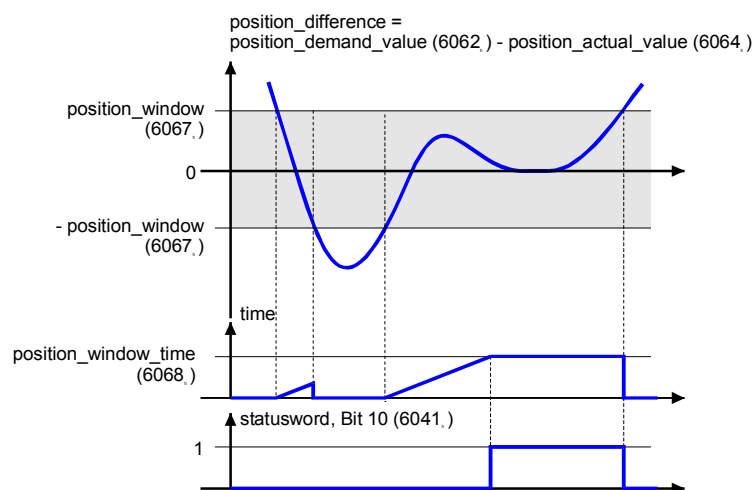


Abbildung 6.21: Position erreicht – Funktionsübersicht

Die Abbildung 6.22 zeigt, wie die Fensterfunktion für die Meldung „Position erreicht“ definiert ist. Symmetrisch um die Zielposition (**target_position**) x_i ist der Positionsbereich zwischen $x_i - x_0$ und $x_i + x_0$ definiert. Die Positionen x_{t0} und x_{t1} liegen z.B. innerhalb dieses Positionsfensters (**position_window**). Wenn sich der Antrieb in diesem Fenster befindet, dann wird im Regler ein Timer gestartet. Wenn dieser Timer die im Objekt **position_window_time** vorgegebene Zeit erreicht und sich der Antrieb während dieser Zeit ununterbrochen im gültigen Bereich zwischen $x_i - x_0$ und $x_i + x_0$ befindet, dann wird Bit 10 **target_reached** im **statusword** gesetzt. Sobald der Antrieb den zulässigen Bereich verlässt, wird sowohl das Bit 10 als auch der Timer auf Null gesetzt.

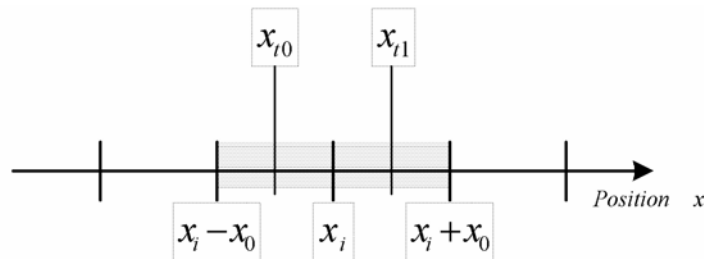


Abbildung 6.22: Position erreicht

6.7.2 Beschreibung der Objekte

6.7.2.1 In diesem Kapitel behandelte Objekte

Index	Objekt	Name	Typ	Attr.
202D _h	VAR	position_demand_sync_value	INT32	ro
2030 _h	VAR	set_position_absolute	INT32	wo
6062 _h	VAR	position_demand_value	INT32	ro
6063 _h	VAR	position_actual_value*	INT32	ro
6064 _h	VAR	position_actual_value	INT32	ro
6065 _h	VAR	following_error_window	UINT32	rw
6066 _h	VAR	following_error_time_out	UINT16	rw
6067 _h	VAR	position_window	UINT32	rw
6068 _h	VAR	position_window_time	UINT16	rw
607B _h	ARRAY	position_range_limit	INT32	rw
60FA _h	VAR	control_effort	INT32	ro
60FB _h	RECORD	position_control_parameter_set		rw
60FC _h	VAR	position_demand_value*	INT32	ro
6510 _{h_20h}	VAR	position_range_limit_enable	UINT16	rw
6510 _{h_22h}	VAR	position_error_switch_off_limit	UINT32	rw

6.7.2.2 Betroffene Objekte aus anderen Abschnitten

Index	Objekt	Name	Typ	Abschnitt
607A _h	VAR	target_position	INT32	8.3 Betriebsart Positionieren
607C _h	VAR	home_offset	INT32	8.2 Referenzfahrt
607D _h	VAR	software_position_limit	INT32	8.3 Betriebsart Positionieren
607E _h	VAR	polarity	UINT8	6.3 Umrechnungsfaktoren
6093 _h	VAR	position_factor	UINT32	6.3 Umrechnungsfaktoren
6094 _h	ARRAY	velocity_encoder_factor	UINT32	6.3 Umrechnungsfaktoren
6096 _h	ARRAY	acceleration_factor	UINT32	6.3 Umrechnungsfaktoren
6040 _h	VAR	controlword	INT16	7 Gerätesteuerung
6041 _h	VAR	statusword	UINT16	7 Gerätesteuerung

6.7.2.3 Objekt 60FB_h: position_control_parameter_set

Der Parametersatz des Servoreglers muss für die Applikation angepasst werden. Die Daten des Lagereglers müssen bei der Inbetriebnahme der Anlage mit Hilfe des Programms `S2Commander optimal` bestimmt werden.



Falsche Einstellungen der Lagereglerparameter können zu starken Schwingungen führen und eventuell Teile der Anlage zerstören !

Der Lageregler vergleicht die Soll-Lage mit der Ist-Lage und bildet aus der Differenz unter Berücksichtigung der Verstärkung und eventuell des Integrators eine Korrektorgeschwindigkeit (Objekt **60FA_h: control_effort**), die dem Drehzahlregler zugeführt wird. Der Lageregler ist, gemessen am Strom- und Drehzahlregler, relativ langsam. Der Regler arbeitet daher intern mit Aufschaltungen, so dass die Ausregelarbeit für den Lageregler minimiert wird und der Regler schnell einschwingen kann.

Als Lageregler genügt normalerweise ein Proportional-Glied. Die Verstärkung des Lagereglers muss mit 256 multipliziert werden. Bei einer Verstärkung von 1.5 im Menü „Lageregler“ des Parametrierprogramms `S2Commander` ist in das Objekt **position_control_gain** der Wert 384 einzuschreiben.

Normalerweise kommt der Lageregler ohne Integrator aus. Dann ist in das Objekt **position_control_time** der Wert Null einzuschreiben. Andernfalls muss die Zeitkonstante des Lagereglers in Mikrosekunden umgerechnet werden. Bei einer Zeit von 4.0 Millisekunden ist entsprechend der Wert 4000 in das Objekt **position_control_time** einzutragen.

Da der Lageregler schon kleinste Lageabweichungen in nennenswerte Korrekturgeschwindigkeiten umsetzt, würde es im Falle einer kurzen Störung (z.B. kurzzeitiges Klemmen der Anlage) zu sehr heftigen Ausregelvorgängen mit sehr großen Korrekturgeschwindigkeiten kommen. Dieses ist zu vermeiden, wenn der Ausgang des Lagereglers über das Objekt `position_control_v_max` sinnvoll (z.B. 500 min⁻¹) begrenzt wird.

Mit dem Objekt `position_error_tolerance_window` kann die Größe einer Lageabweichung definiert werden, bis zu der der Lageregler nicht eingreift (Totbereich). Dieses kann zur Stabilisierung eingesetzt werden, wenn z.B. Spiel in der Anlage vorhanden ist.

Index	60FB_h
Name	position_control_parameter_set
Object Code	RECORD
No. of Elements	4

Sub-Index	01_h
Description	position_control_gain
Data Type	UINT16
Access	rw
PDO Mapping	no
Units	256 = „1“
Value Range	0...64*256 (16384)
Default Value	102

Sub-Index	02_h
Description	position_control_time
Data Type	UINT16
Access	rw
PDO Mapping	no
Units	µs
Value Range	0
Default Value	0

Sub-Index	04_h
Description	position_control_v_max
Data Type	UINT32
Access	rw
PDO Mapping	no
Units	speed units
Value Range	0...131072 min ⁻¹
Default Value	500 min ⁻¹

Sub-Index	05_h
Description	position_error_tolerance_window
Data Type	UINT32
Access	rw
PDO Mapping	no
Units	position units
Value Range	1...65536 (1 U)
Default Value	2 (1 / 32768 U)

6.7.2.4 Objekt 6062_h: position_demand_value

Über dieses Objekt kann der aktuelle Lage-Sollwert ausgelesen werden. Diese wird vom Fahrkurven-Generator in den Lageregler eingespeist.

Index	6062_h
Name	position_demand_value
Object Code	VAR
Data Type	INT32

Access	ro
PDO Mapping	yes
Units	position units
Value Range	--
Default Value	--

6.7.2.5 Objekt 202D_h: position_demand_sync_value

Über dieses Objekt kann die Soll-Lage des Synchronisationsgeber ausgelesen werden. Diese wird durch das Objekt **2022_h synchronization_encoder_select** (siehe Abschnitt 6.11.2.4: *Objekt 2022_h: synchronisation_encoder_selection, Seite 125*) definiert. Dieses Objekt wird in benutzerdefinierten Einheiten angegeben.

Index	202D_h
Name	position_demand_sync_value
Object Code	VAR
Data Type	INT32

Ab Firmware 3.2.0.1.1

Access	ro
PDO Mapping	no
Units	position units
Value Range	--
Default Value	--

6.7.2.6 Objekt 6064_h: position_actual_value

Über dieses Objekt kann die Ist-Lage ausgelesen werden. Diese wird dem Lageregler vom Winkelgeber aus zugeführt. Dieses Objekt wird in benutzerdefinierten Einheiten angegeben.

Index	6064_h
Name	position_actual_value
Object Code	VAR
Data Type	INT32

Access	ro
PDO Mapping	yes
Units	position units
Value Range	--
Default Value	--

6.7.2.7 Objekt 6065_h: following_error_window

Das Objekt **following_error_window** (Schleppfehler-Fenster) definiert um den Lage-Sollwert (**position_demand_value**) einen symmetrischen Bereich. Wenn sich der Lage-Istwert (**position_actual_value**) außerhalb des Schleppfehler-Fensters (**following_error_window**) befindet, dann tritt ein Schleppfehler auf und das Bit 13 im Objekt **statusword** wird gesetzt. Folgende Ursachen können einen Schleppfehler verursachen:

- der Antrieb ist blockiert
- die Positioniergeschwindigkeit ist zu groß
- die Beschleunigungswerte sind zu groß
- das Objekt **following_error_window** ist mit einem zu kleinen Wert besetzt
- der Lageregler ist nicht richtig parametrier

Index	6065_h
Name	following_error_window
Object Code	VAR
Data Type	UINT32

Access	rw
PDO Mapping	yes
Units	position units
Value Range	--
Default Value	9101 (9101 / 65536 U = 50°)

6.7.2.8 Objekt 6066_h: following_error_time_out

Tritt ein Schleppfehler – länger als in diesem Objekt definiert – auf, dann wird das zugehörige Bit 13 **following_error** im **statusword** gesetzt.

Index	6066_h
Name	following_error_time_out
Object Code	VAR
Data Type	UINT16

Access	rw
PDO Mapping	yes
Units	ms
Value Range	0...27314
Default Value	0

6.7.2.9 Objekt 60FA_h: control_effort

Die Ausgangsgröße des Lagereglers kann über dieses Objekt ausgelesen werden. Dieser Wert wird intern dem Drehzahlregler als Sollwert zugeführt.

Index	60FA_h
Name	control_effort
Object Code	VAR
Data Type	INT32

Access	ro
PDO Mapping	yes
Units	speed units
Value Range	--
Default Value	--

6.7.2.10 Objekt 6067_h: position_window

Mit dem Objekt **position_window** wird um die Zielposition (**target_position**) herum ein symmetrischer Bereich definiert. Wenn der Lage-Istwert (**position_actual_value**) eine bestimmte Zeit innerhalb dieses Bereiches liegt, wird die Zielposition (**target_position**) als erreicht angesehen.

Index	6067_h
Name	position_window
Object Code	VAR
Data Type	UINT32

Access	rw
PDO Mapping	yes
Units	position units
Value Range	--
Default Value	1820 (1820 / 65536 U = 10°)

6.7.2.11 Objekt 6068_h: position_window_time

Wenn sich die Ist-Position des Antriebes innerhalb des Positionierfensters (**position_window**) befindet und zwar solange, wie in diesem Objekt definiert, dann wird das zugehörige Bit 10 **target_reached** im **statusword** gesetzt.

Index	6068_h
Name	position_window_time
Object Code	VAR
Data Type	UINT16

Access	rw
PDO Mapping	yes
Units	ms
Value Range	--
Default Value	0

6.7.2.12 Objekt 6510_h_22_h: position_error_switch_off_limit

Im Objekt **position_error_switch_off_limit** kann die maximal zulässige Abweichung zwischen der Soll- und der Istposition eingetragen werden. Im Gegensatz zur o.g. Schleppfehlermeldung wird bei einer Überschreitung die Endstufe sofort abgeschaltet und ein Fehler ausgelöst. Der Motor trudelt somit ungebremst aus (außer es ist eine Haltebremse vorhanden).

Index	6510_h
Name	drive_data
Object Code	RECORD
No. of Elements	51

Sub-Index	22_h
Description	position_error_switch_off_limit
Data Type	UINT32
Access	rw
PDO Mapping	no
Units	position units
Value Range	0...2 ³² -1
Default Value	0

Ab Firmware 3.2.0.1.1

Wert	Bedeutung
0	Grenzwert Schleppfehler AUS (Reaktion KEINE AKTION)
> 0	Grenzwert Schleppfehler EIN (Reaktion ENDSTUFE SOFORT ABSCHALTEN)

Die Aktivierung des Fehlers 17-0 erfolgt durch Änderung der Fehlerreaktion. Die Reaktion ENDSTUFE SOFORT ABSCHALTEN wird als **EIN**, alle anderen als **AUS** zurückgegeben. Beim Beschreiben mit 0 wird die Fehlerreaktion KEINE AKTION gesetzt, beim Beschreiben mit einem Wert größer 0 die Fehlerreaktion ENDSTUFE SOFORT ABSCHALTEN. Siehe hierzu auch Abschnitt 6.18: *Fehlermanagement*, ab Seite 151.

6.7.2.13 Objekt 607B_h: position_range_limit

Die Objektgruppe **position_range_limit** enthält zwei Unterparameter, die den numerischen Bereich der Positionswerte beschränken. Wenn eine dieser Grenzen überschritten wird, springt der Positionswert automatisch an die jeweils andere Grenze. Dieses ermöglicht die Parametrierung von sog. Rundachsen. Anzugeben sind die Grenzen, die physikalisch der gleichen Position entsprechen sollen, also beispielsweise 0° und 360°.

Damit diese Grenzen wirksam werden, muss über das Objekt **6510_h_20_h** (**position_range_limit_enable**) ein Rundachsmodus ausgewählt werden.

Index	607B_h
Name	position_range_limit
Object Code	ARRAY
No. of Elements	2
Data Type	INT32

Ab Firmware 3.3.x.1.1

Sub-Index	01_h
Description	min_position_range_limit
Access	rw
PDO Mapping	yes
Units	position units
Value Range	--
Default Value	--

Ab Firmware 3.3.x.1.1

Sub-Index	02_h
Description	max_position_range_limit
Access	rw
PDO Mapping	yes
Units	position units
Value Range	--
Default Value	--

Ab Firmware 3.3.x.1.1

6.7.2.14 Objekt 6510_h_20_h: position_range_limit_enable

Über das Objekt **position_range_limit_enable** können die durch das Objekt **607B_h** definierten Bereichsgrenzen aktiviert werden. Es sind verschiedene Modi möglich:

Wird der Modus "Kürzester Weg" gewählt, werden Positionierungen immer auf der physikalisch kürzeren Strecke zum Ziel ausgeführt. Der Antrieb passt dazu selber das Vorzeichen der Fahrgeschwindigkeit an. Bei den beiden Modi "Feste Drehrichtung" erfolgt die Positionierung grundsätzlich nur in die im Modus angegebene Richtung.

Index	6510_h
Name	drive_data
Object Code	RECORD
No. of Elements	51

Sub-Index	20_h
Description	position_range_limit_enable
Data Type	UINT16
Access	rw
PDO Mapping	no
Units	--
Value Range	0...5
Default Value	0

Ab Firmware 3.3.x.1.1

Wert	Bedeutung
0	Aus
1	Kürzester Weg (Aus Kompatibilitätsgründen)
2	Kürzester Weg
3	Reserviert
4	Feste Drehrichtung „Positiv“
5	Feste Drehrichtung „Negativ“

6.7.2.15 Objekt 2030_h: set_position_absolute

Über das Objekt **set_position_absolute** kann die auslesbare Istposition verschoben werden, ohne dass sich die physikalische Lage ändert. Der Antrieb führt dabei keine Bewegung aus. Wenn ein absolutes Gebersystem angeschlossen ist, wird die Lageverschiebung im Geber gespeichert, sofern das Gebersystem dies zulässt. Die Lageverschiebung bleibt in diesem Fall also nach einem Reset erhalten. Diese Speicheroperation läuft unabhängig von diesem Objekt im Hintergrund ab. Es werden dabei ebenfalls alle dem Geberspeicher zugehörigen Parameter mit ihren aktuellen Werten gespeichert.

Index	2030 _h
Name	set_position_absolute
Object Code	VAR
Data Type	INT32

Ab Firmware 3.5.x.1.1

Access	wo
PDO Mapping	no
Units	position units
Value Range	--
Default Value	--

6.8 Sollwert-Begrenzung

6.8.1 Beschreibung der Objekte

6.8.1.1 In diesem Kapitel behandelte Objekte

Index	Objekt	Name	Typ	Attr.
2415 _h	RECORD	current_limitation		rw
2416 _h	RECORD	speed_limitation		rw

6.8.1.2 Objekt 2415_h: current_limitation

Mit der Objektgruppe **current_limitation** kann in den Betriebsarten `profile_position_mode`, `interpolated_position_mode`, `homing_mode` und `velocity_mode` der Maximalstrom für den Motor begrenzt werden, wodurch z.B. ein drehmomentbegrenzter Drehzahlbetrieb ermöglicht wird. Über das Objekt **limit_current_input_channel** wird die Sollwert-Quelle des Begrenzungsmoment vorgegeben. Hier kann zwischen der Vorgabe eines direkten Sollwerts (Fester Wert) oder der Vorgabe über einen analogen Eingang gewählt werden. Über das Objekt **limit_current** wird je nach gewählter Quelle entweder das Begrenzungsmoment (Quelle = Fester Wert) oder der Skalierungsfaktor für die Analogeingänge (Quelle = Analogeingang) vorgegeben. Im ersten Fall wird direkt auf den momentproportionalen Strom in mA begrenzt, im zweiten Fall wird der Strom in mA angegeben, der einer anliegenden Spannung von 10V entsprechen soll.

Index	2415_h
Name	current_limitation
Object Code	RECORD
No. of Elements	2

Sub-Index	01_h
Description	limit_current_input_channel
Data Type	UINT8
Access	rw
PDO Mapping	no
Units	--
Value Range	0...4
Default Value	0

Sub-Index	02_h
Description	limit_current
Data Type	INT32
Access	rw
PDO Mapping	no
Units	mA
Value Range	--
Default Value	0

Wert	Bedeutung
0	Keine Begrenzung
1	AIN0
2	AIN1
3	AIN2
4	Feldbus (Feldbus-Selektor 2)

6.8.1.3 Objekt 2416_h: speed_limitation

Mit der Objektgruppe **speed_limitation** kann in der Betriebsart `profile_torque_mode` die Maximaldrehzahl des Motors begrenzt werden, wodurch ein drehzahlbegrenzter Drehmomentbetrieb ermöglicht wird. Über das Objekt **limit_speed_input_channel** wird die Sollwert-Quelle der Begrenzungsdrehzahl vorgegeben. Hier kann zwischen der Vorgabe eines direkten Sollwerts (Fester Wert) oder der Vorgabe über einen analogen Eingang gewählt werden. Über das Objekt **limit_speed** wird je nach gewählter Quelle entweder die Begrenzungsdrehzahl (Quelle = Fester Wert) oder der Skalierungsfaktor für die Analogeingänge (Quelle = Analogeingang) vorgegeben. Im ersten Fall wird direkt auf die angegebene Drehzahl begrenzt, im zweiten Fall wird die Drehzahl angegeben, die einer anliegenden Spannung von 10V entsprechen soll.

Index	2416_h
Name	speed_limitation
Object Code	RECORD
No. of Elements	2

Ab Firmware 3.3.0.1.1

Sub-Index	01_h
Description	limit_speed_input_channel
Data Type	UINT8
Access	rw
PDO Mapping	no
Units	--
Value Range	0...4
Default Value	0

Ab Firmware 3.3.0.1.1

Sub-Index	02_h
Description	limit_speed
Data Type	INT32
Access	rw
PDO Mapping	no
Units	speed units
Value Range	--
Default Value	--

Ab Firmware 3.3.0.1.1

Wert	Bedeutung
0	Keine Begrenzung
1	AIN0
2	AIN1
3	AIN2
4	Feldbus (Feldbus-Selektor 2)

6.9 Geberanpassungen

6.9.1 Übersicht

Dieses Kapitel beschreibt die Konfiguration des Winkelgebereingangs X2A, X2B und des Inkrementaleingangs X10.



Vorsicht!

Falsche Winkelgeber-Einstellungen können den Antrieb unkontrolliert drehen lassen und eventuell Teile der Anlage zerstören.

6.9.2 Beschreibung der Objekte

6.9.2.1 In diesem Kapitel behandelte Objekte

Index	Objekt	Name	Typ	Attr.
2024 _h	RECORD	encoder_x2a_data_field		ro
2024 _{h_01h}	VAR	encoder_x2a_resolution	UINT32	ro
2024 _{h_02h}	VAR	encoder_x2a_numerator	INT16	rw
2024 _{h_03h}	VAR	encoder_x2a_divisor	INT16	rw
2025 _h	RECORD	encoder_x10_data_field		ro
2025 _{h_01h}	VAR	encoder_x10_resolution	UINT32	rw
2025 _{h_02h}	VAR	encoder_x10_numerator	INT16	rw
2025 _{h_03h}	VAR	encoder_x10_divisor	INT16	rw
2025 _{h_04h}	VAR	encoder_x10_counter	UINT32	ro
2026 _h	RECORD	encoder_x2b_data_field		ro
2026 _{h_01h}	VAR	encoder_x2b_resolution	UINT32	rw
2026 _{h_02h}	VAR	encoder_x2b_numerator	INT16	rw
2026 _{h_03h}	VAR	encoder_x2b_divisor	INT16	rw
2026 _{h_04h}	VAR	encoder_x2b_counter	UINT32	ro

6.9.2.2 Objekt 2024_h: encoder_x2a_data_field

Im Record **encoder_x2a_data_field** sind Parameter zusammengefasst, die für den Betrieb des Winkelgebers am Stecker X2A notwendig sind.

Da zahlreiche Winkelgeber- Einstellungen nur nach einem Reset wirksam werden, sollte die Auswahl und die Einstellung der Geber über den S2Commander erfolgen. Unter CANopen lassen sich folgende Einstellungen auslesen bzw. ändern:

Das Objekt **encoder_x2a_resolution** gibt an, wie viele Inkremente vom Geber pro Umdrehung oder Längeneinheit erzeugt werden. Da am Eingang X2A nur Resolver angeschlossen werden können, die immer mit 16 Bit ausgewertet werden, wird hier immer 65536 zurückgegeben. Mit dem Objekt **encoder_x2a_numerator** und **encoder_x2a_divisor** kann ein eventuelles Getriebe (auch mit Vorzeichen) **zwischen Motorwelle und Geber** berücksichtigt werden.

Index	2024_h
Name	encoder_x2a_data_field
Object Code	RECORD
No. of Elements	3

Ab Firmware 3.2.0.1.1

Sub-Index	01_h
Description	encoder_x2a_resolution
Data Type	UINT32
Access	ro
PDO Mapping	No
Units	Inkremente (4 * Strichzahl)
Value Range	--
Default Value	65536

Ab Firmware 3.2.0.1.1

Sub-Index	02_h
Description	encoder_x2a_numerator
Data Type	INT16
Access	rw
PDO Mapping	no
Units	--
Value Range	-32768 ... 32767 (außer 0)
Default Value	1

Ab Firmware 3.2.0.1.1

Sub-Index	03_h
Description	encoder_x2a_divisor
Data Type	INT16
Access	rw
PDO Mapping	no
Units	--
Value Range	1 ... 32767
Default Value	1

Ab Firmware 3.2.0.1.1

6.9.2.3 Objekt 2026_h: encoder_x2b_data_field

Im Record **encoder_x2b_data_field** sind Parameter zusammengefasst, die für den Betrieb des Winkelgebers am Stecker X2B notwendig sind.

Das Objekt **encoder_x2b_resolution** gibt an, wie viele Inkremente vom Geber pro Umdrehung erzeugt werden (Bei Inkrementalgebern entspricht dies dem vierfachen der Strichzahl bzw der Perioden pro Umdrehung).

Das Objekt **encoder_x2b_counter** liefert die aktuell gezählte Inkrementzahl. Es liefert daher Werte zwischen 0 und der eingestellten Inkrementzahl-1. Mit den Objekten **encoder_x2b_numerator** und **encoder_x2b_divisor** kann ein Getriebe **zwischen Motorwelle und dem an X2b angeschlossenen Geber** berücksichtigt werden.

Index	2026_h
Name	encoder_x2b_data_field
Object Code	RECORD
No. of Elements	4

Ab Firmware 3.2.0.1.1

Sub-Index	01_h
Description	encoder_x2b_resolution
Data Type	UINT32
Access	rw
PDO Mapping	no
Units	Inkremente (4 * Strichzahl)
Value Range	hängt vom benutzten Geber ab
Default Value	hängt vom benutzten Geber ab

Ab Firmware 3.2.0.1.1

Sub-Index	02_h
Description	encoder_x2b_numerator
Data Type	INT16
Access	rw
PDO Mapping	no
Units	--
Value Range	-32768 ... 32767
Default Value	1

Ab Firmware 3.3.0.1.1

Sub-Index	03_h
Description	encoder_x2b_divisor
Data Type	INT16
Access	rw
PDO Mapping	no
Units	--
Value Range	1 ... 32767
Default Value	1

Ab Firmware 3.3.0.1.1

Sub-Index	04_h
Description	encoder_x2b_counter
Data Type	UINT32
Access	ro
PDO Mapping	yes
Units	Inkmente (4 * Strichzahl)
Value Range	0 ... (encoder_x2b_resolution – 1)
Default Value	--

Ab Firmware 3.2.0.1.1

6.9.2.4 Objekt 2025_h: encoder_x10_data_field

Im Record **encoder_X10_data_field** sind Parameter zusammengefasst, die für den Betrieb des Inkrementaleingangs X10 notwendig sind. Hier kann wahlweise ein digitaler Inkrementalgeber oder emulierte Inkrementalsignale beispielsweise eines anderen SERVOTEC S2 angeschlossen werden. Die Eingangssignale über X10 können wahlweise als Sollwert oder als Istwert verwendet werden. Näheres hierzu finden Sie in Abschnitt 6.11: *Soll-/Istwertaufschaltung, ab Seite 122.*

Im Objekt **encoder_X10_resolution** muss angegeben werden, wie viele Inkremente vom Geber pro Umdrehung des Gebers erzeugt werden. Dies entspricht dem vierfachen der Strichzahl. Das Objekt **encoder_X10_counter** liefert die aktuell gezählte Inkrementzahl (Zwischen 0 und der eingestellten Inkrementzahl-1).

Mit dem Objekt **encoder_X10_numerator** und **encoder_X10_divisor** kann ein eventuelles Getriebe (auch mit Vorzeichen) berücksichtigt werden.

Bei der Verwendung des X10- Signals als Istwert entspräche dies einem Getriebe zwischen dem Motor und dem an X10 angeschlossenen Istwertgeber, welches am Abtrieb montiert ist. Bei der Verwendung des X10- Signals als Sollwert, können hiermit Getriebeübersetzungen zwischen Master und Slave realisiert werden.

Index	2025_h
Name	encoder_x10_data_field
Object Code	RECORD
No. of Elements	4

Ab Firmware 3.2.0.1.1

Sub-Index	01_h
Description	encoder_x10_resolution
Data Type	UINT32
Access	rw
PDO Mapping	no
Units	Inkremente (4 * Strichzahl)
Value Range	hängt vom benutzten Geber ab
Default Value	hängt vom benutzten Geber ab

Ab Firmware 3.2.0.1.1

Sub-Index	02_h
Description	encoder_x10_numerator
Data Type	INT16
Access	rw
PDO Mapping	no
Units	--
Value Range	-32768 ... 32767 (außer 0)
Default Value	1

Ab Firmware 3.2.0.1.1

Sub-Index	03_h
Description	encoder_x10_divisor
Data Type	INT16
Access	rw
PDO Mapping	no
Units	--
Value Range	1 ... 32767
Default Value	1

Ab Firmware 3.2.0.1.1

Sub-Index	04_h
Description	encoder_x10_counter
Data Type	UINT32
Access	ro
PDO Mapping	yes
Units	Inkremente (4 * Strichzahl)
Value Range	0 ... (encoder_x10_resolution – 1)
Default Value	--

Ab Firmware 3.2.0.1.1

6.10 Inkrementalgeberemulation

6.10.1 Übersicht

Diese Objekt- Gruppe ermöglicht es, den Inkrementalgeberausgang X11 zu parametrieren. Somit können Master- Slave- Applikationen, bei denen der X11 Ausgang des Masters an den X10- Eingang des Slave angeschlossen ist, hiermit unter CANopen parametriert werden.

6.10.2 Beschreibung der Objekte

6.10.2.1 In diesem Kapitel behandelte Objekte

Index	Objekt	Name	Typ	Attr.
2028 _h	VAR	encoder_emulation_resolution	INT32	rw
201A _h	RECORD	encoder_emulation_data		ro
201A _h _01 _h	VAR	encoder_emulation_resolution	INT32	rw
201A _h _02 _h	VAR	encoder_emulation_offset	INT16	rw

6.10.2.2 Objekt 201A_h: encoder_emulation_data

Der Object- Record **encoder_emulation_data** kapselt alle Einstellmöglichkeiten für den Inkrementalgeberausgang X11:

Über das Objekt **encoder_emulation_resolution** kann die ausgegebene Inkrementzahl (= vierfache Strichzahl) als Vielfaches von 4 frei eingestellt werden. In einer Master- Slave- Applikation muss diese der **encoder_X10_resolution** des Slave entsprechen, um ein Verhältnis von 1:1 zu erreichen.

Mit dem Objekt **encoder_emulation_offset** kann die Position des ausgegebenen Nullimpulses gegenüber der Nulllage des Istwertgebers verschoben werden.

Index	201A_h
Name	encoder_emulation_data
Object Code	RECORD
No. of Elements	2

Ab Firmware 3.2.0.1.1

Sub-Index	01_h
Description	encoder_emulation_resolution
Data Type	INT32
Access	rw
PDO Mapping	no
Units	Inkremete (4 * Strichzahl)
Value Range	4 * (1...8192)
Default Value	4096

Ab Firmware 3.2.0.1.1

Sub-Index	02_h
Description	encoder_emulation_offset
Data Type	INT16
Access	rw
PDO Mapping	no
Units	32767 = 180°
Value Range	-32768...32767
Default Value	0

Ab Firmware 3.2.0.1.1

6.10.2.3 Objekt 2028_h: encoder_emulation_resolution

Das Objekt encoder_emulation_resolution ist nur aus Kompatibilitätsgründen vorhanden. Es entspricht dem Objekt 201A_h_01_h.

Index	2028_h
Name	encoder_emulation_resolution
Object Code	VAR
Data Type	INT32

Ab Firmware 3.2.0.1.1

Access	rw
PDO Mapping	no
Units	siehe 201A _h _01 _h
Value Range	siehe 201A _h _01 _h
Default Value	siehe 201A _h _01 _h

6.11 Soll-/Istwertaufschaltung

6.11.1 Übersicht

Mit Hilfe der nachfolgenden Objekte kann die Quelle für den Sollwert und die Quelle für den Istwert geändert werden. Als Standard verwendet der Regler den Eingang für den Motorgeber X2A bzw. X2B als Istwert für den Lageregler. Bei Verwendung eines externen Lagegebers, z.B. hinter einem Getriebe, kann der über X10 eingespeiste Lagewert als Istwert für den Lageregler aufgeschaltet werden. Darüber hinaus ist es möglich über X10 eingehende Signale (z.B. eines zweiten Reglers) als zusätzlichen Sollwert aufzuschalten, wodurch Synchronbetriebsarten ermöglicht werden.

6.11.2 Beschreibung der Objekte

6.11.2.1 In diesem Kapitel behandelte Objekte

Index	Objekt	Name	Typ	Attr.
201F _n	VAR	commutation_encoder_select	INT16	rw
2021 _n	VAR	position_encoder_selection	INT16	rw
2022 _n	VAR	synchronisation_encoder_selection	INT16	rw
2023 _n	VAR	synchronisation_filter_time	UINT32	rw
202F _n	RECORD	synchronisation_selector_data		ro
202F _{n_07} _n	VAR	synchronisation_main	UINT16	rw

6.11.2.2 Objekt 201F_h: commutation_encoder_select

Das Objekt **commutation_encoder_select** gibt den Gebereingang an, der als Kommutiergeber verwendet wird. Da dieser Wert erst nach einem Reset wirksam wird, sollte die Einstellung des Kommutiergebers grundsätzlich über den S2Commander erfolgen.

Index	201F_h
Name	commutation_encoder_select
Object Code	VAR
Data Type	INT16

Ab Firmware 3.2.0.1.1

Access	rw
PDO Mapping	no
Units	--
Value Range	0, 2 (siehe Tabelle)
Default Value	0

Wert	Bezeichnung
0	X2A
2	X2B

6.11.2.3 Objekt 2021_h: position_encoder_selection

Das Objekt **position_encoder_selection** gibt den Gebereingang an, der zur Bestimmung der Istlage (Istwertgeber) verwendet wird. Dieser Wert kann geändert werden, um auf Lage-
regelung über einen externen (am Abtrieb angeschlossenen) Geber umzuschalten. Dabei
kann zwischen X10 und dem als Kommutiergeber ausgewählten Gebereingang (X2A / X2B)
umgeschaltet werden. Wird einer der Gebereingänge X2A / X2B als Lageistwertgeber
ausgewählt, so muss derjenige verwendet werden, der als Kommutiergeber genutzt wird. Wird
der jeweils andere Geber angewählt, wird automatisch auf den Kommutiergeber
umgeschaltet.

Index	2021_h
Name	position_encoder_selection
Object Code	VAR
Data Type	INT16

Ab Firmware 3.2.0.1.1

Access	rw
PDO Mapping	no
Units	--
Value Range	0...2 (siehe Tabelle)
Default Value	0

Wert	Bezeichnung
0	X2A
1	X2B
2	X10



Es kann nur zwischen dem Gebereingang X10 und dem jeweiligen Kommutiergeber X2A oder X2B als Lageistwertgeber gewählt werden. Die Konfiguration X2A als Kommutiergeber und X2B als Lageistwertgeber zu nutzen, bzw. umgekehrt, ist nicht möglich.

6.11.2.4 Objekt 2022_h: synchronisation_encoder_selection

Das Objekt **synchronisation_encoder_selection** gibt den Gebereingang an, der als Synchronisationssollwert verwendet wird. Je nach Betriebsart entspricht dieses einem Lage-sollwert (Profile Position Mode) oder einem Drehzahlsollwert (Profile Velocity Mode).

Als Synchronisationseingang kann nur X10 verwendet werden. Somit kann zwischen X10 und keinem Eingang ausgewählt werden. Als Synchronisationssollwert sollte nicht der gleiche Eingang wie für den Istwertgeber gewählt werden.

Index	2022_h
Name	synchronisation_encoder_selection
Object Code	VAR
Data Type	INT16

Ab Firmware 3.2.0.1.1

Access	rw
PDO Mapping	no
Units	--
Value Range	-1, 2 (siehe Tabelle)
Default Value	2

Wert	Bezeichnung
-1	kein Geber / undefiniert
2	X10

6.11.2.5 Objekt 202F_h: synchronisation_selector_data

Über das Objekt **synchronisation_main** kann die Aufschaltung eines Synchronsollwerts erfolgen. Damit der Synchronsollwert überhaupt berechnet wird, muss Bit 0 gesetzt werden. Bit 1 ermöglicht es in zukünftigen Firmware- Versionen die Synchronlage erst durch das Starten eines Positionssatzes aufzuschalten. Zur Zeit ist nur 0 parametrierbar, so dass die Synchronlage immer zugeschaltet ist. Über das Bit 8 kann festgelegt werden, dass die Referenzfahrt ohne Aufschaltung der Synchronlage erfolgen soll, um Master und Slave getrennt referenzieren zu können.

Index	202F_h
Name	synchronisation_selector_data
Object Code	RECORD
No. of Elements	1

Ab Firmware 3.2.0.1.1

Sub-Index	07_h
Description	synchronisation_main
Data Type	UINT16
Access	rw
PDO Mapping	no
Units	--
Value Range	siehe Tabelle
Default Value	--

Ab Firmware 3.2.0.1.1

Bit	Wert	Bedeutung
0	0001 _h	0: Synchronisation inaktiv 1: Synchronisation aktiv
1	0002 _h	“Fliegende Säge” nicht möglich
8	0100 _h	0: Synchronisation während der Referenzfahrt 1: Keine Synchronisation während der Referenzfahrt

6.11.2.6 Objekt 2023_h: synchronisation_filter_time

Über das Objekt **synchronisation_filter_time** wird die Filterzeitkonstante eines PT1- Filters festgelegt, mit dem die Synchronisationsdrehzahl geglättet wird. Dies kann insbesondere bei geringen Strichzahlen nötig sein, da hier bereits kleine Änderungen des Eingangswertes hohen Drehzahlen entsprechend. Andererseits ist der Antrieb bei hohen Filterzeiten ggf. nicht mehr in der Lage schnell genug einem dynamischen Eingangssignal zu folgen.

Index	2023_h
Name	synchronisation_filter_time
Object Code	VAR
Data Type	UINT32

Ab Firmware 3.2.0.1.1

Access	rw
PDO Mapping	no
Units	µs
Value Range	10...50 000
Default Value	600

6.12 Analoge Eingänge

6.12.1 Übersicht

Die Antriebsregler der Reihe SERVOTEC S2 verfügen über drei analoge Eingänge, über die dem Regler beispielsweise Sollwerte vorgegeben werden können. Für alle diese analogen Eingänge bieten die nachfolgenden Objekte die Möglichkeit, die aktuelle Eingangsspannung auszulesen (**analog_input_voltage**) und einen Offset einzustellen (**analog_input_offset**).

6.12.2 Beschreibung der Objekte

Index	Objekt	Name	Typ	Attr.
2400 _h	ARRAY	analog_input_voltage	INT16	ro
2401 _h	ARRAY	analog_input_offset	INT32	rw

6.12.2.1 2400_h: analog_input_voltage (Eingangsspannung)

Die Objektgruppe **analog_input_voltage** liefert die aktuelle Eingangsspannung des jeweiligen Kanals unter Berücksichtigung des Offsets in Millivolt.

Index	2400_h
Name	analog_input_voltage
Object Code	ARRAY
No. of Elements	3
Data Type	INT16

Sub-Index	01_h
Description	analog_input_voltage_ch_0
Access	ro
PDO Mapping	no
Units	mV
Value Range	--
Default Value	--

Sub-Index	02_h
Description	analog_input_voltage_ch_1
Access	ro
PDO Mapping	no
Units	mV
Value Range	--
Default Value	--

Sub-Index	03_h
Description	analog_input_voltage_ch_2
Access	ro
PDO Mapping	no
Units	mV
Value Range	--
Default Value	--

6.12.2.2 Objekt 2401_h: analog_input_offset (Offset Analogeingänge)

Über die Objektgruppe **analog_input_offset** kann die Offsetspannung in Millivolt für die jeweiligen Eingänge gesetzt bzw. gelesen werden. Mit Hilfe des Offsets kann eine eventuelle anliegende Gleichspannung ausgeglichen werden. Ein positiver Offset kompensiert dabei eine positive Eingangsspannung.

Index	2401_h
Name	analog_input_offset
Object Code	ARRAY
No. of Elements	3
Data Type	INT32

Sub-Index	01_h
Description	analog_input_offset_ch_0
Access	rw
PDO Mapping	no
Units	mV
Value Range	-10000...10000
Default Value	0

Sub-Index	02_h
Description	analog_input_offset_ch_1
Access	rw
PDO Mapping	no
Units	mV
Value Range	-10000...10000
Default Value	0

Sub-Index	03_h
Description	analog_input_offset_ch_2
Access	rw
PDO Mapping	no
Units	mV
Value Range	-10000...10000
Default Value	0

6.13 Digitale Ein- und Ausgänge

6.13.1 Übersicht

Alle digitalen Eingänge des Reglers können über den CAN-Bus gelesen und fast alle digitalen Ausgänge können beliebig gesetzt werden. Zudem können den digitalen Ausgängen des Reglers Statusmeldungen zugeordnet werden. Dies ist ebenfalls mit den Ausgängen eines ggf. vorhandenen Technologiemoduls EA88 im Schacht 1 möglich.

6.13.2 Beschreibung der Objekte

6.13.2.1 In diesem Kapitel behandelte Objekte

Index	Objekt	Name	Typ	Attr.
60FD _h	VAR	digital_inputs	UINT32	ro
60FE _h	ARRAY	digital_outputs	UINT32	rw
2420 _h	RECORD	digital_output_state_mapping		ro
2420 _h _01 _h	VAR	dig_out_state_mapp_dout_1	UINT8	rw
2420 _h _02 _h	VAR	dig_out_state_mapp_dout_2	UINT8	rw
2420 _h _03 _h	VAR	dig_out_state_mapp_dout_3	UINT8	rw
2420 _h _11 _h	VAR	dig_out_state_mapp_ea88_0_low	UINT32	rw
2420 _h _12 _h	VAR	dig_out_state_mapp_ea88_0_high	UINT32	rw

6.13.2.2 Objekt 60FD_h: digital_inputs

Über das Objekt 60FD_h können die digitalen Eingänge ausgelesen werden:

Index	60FD _h
Name	digital_inputs
Object Code	VAR
Data Type	UINT32

Access	ro
PDO Mapping	yes
Units	--
Value Range	gemäß u. Tabelle
Default Value	0

Bit	Wert	digitaler Eingang
0	00000001 _h	Negativer Endschalter
1	00000002 _h	Positiver Endschalter
2	00000004 _h	Referenzschalter
3	00000008 _h	Interlock (Regler- oder Endstufenfreigabe fehlt)
16...23	00FF0000 _h	Gegebenenfalls zusätzliche digitale Eingänge eines EA88-Moduls (EA88-0)
24...27	0F000000 _h	DIN0...DIN3
28	10000000 _h	DIN8
29	20000000 _h	DIN9

6.13.2.3 Objekt 60FE_h: digital_outputs

Über das Objekt 60FE_h können die digitalen Ausgänge angesteuert werden. Hierzu ist im Objekt **digital_outputs_mask** anzugeben, welche der digitalen Ausgänge angesteuert werden sollen. Über das Objekt **digital_outputs_data** können die ausgewählten Ausgänge dann beliebig gesetzt werden. Es ist zu beachten, dass bei der Ansteuerung der digitalen Ausgänge eine Verzögerung von bis zu 10 ms auftreten kann. Wann die Ausgänge wirklich gesetzt werden, kann durch Zurücklesen des Objekts 60FE_h festgestellt werden.

Index	60FE _h
Name	digital_outputs
Object Code	ARRAY
No. of Elements	2
Data Type	UINT32

Sub-Index	01 _h
Description	digital_outputs_data
Access	rw
PDO Mapping	yes
Units	--
Value Range	--
Default Value	(abhängig vom Zustand der Bremse)

Sub-Index	02 _h
Description	digital_outputs_mask
Access	rw
PDO Mapping	yes
Units	--
Value Range	--
Default Value	00000000 _h

Bit	Wert	Digitaler Ausgang
0	00000001 _h	1 = Bremse anziehen
16...23	00FF0000 _h	ggf. zusätzliche digitale Ausgänge eines EA88-Moduls (EA88-0)
25...27	0E000000 _h	DOUT1...DOUT3



Wenn die Bremsansteuerung über digital_output_mask freigegeben ist, wird durch Löschen von Bit 0 in digital_output_data die Haltebremse manuell gelüftet!
Dies kann bei hängenden Achsen zu einem Absacken der Achse führen.

6.13.2.4 Objekt 2420_h: digital_output_state_mapping

Über die Objektgruppe **digital_outputs_state_mapping** können verschiedene Statusmeldungen des Reglers über die digitalen Ausgänge ausgegeben werden.

Für die integrierten digitalen Ausgänge des Reglers ist hierzu für jeden Ausgang ein eigener Subindex vorhanden. Für die optional verfügbaren Ausgänge eines EA88- Moduls im Technologieschacht 1 sind innerhalb eines Subindex immer 4 Ausgänge zusammengefasst. Somit ist für jeden Ausgang ein Byte vorhanden, in das die Funktionsnummer einzutragen ist.

Wenn einem digitalen Ausgang eine derartige Funktion zugeordnet wurde und der Ausgang dann direkt über **digital_outputs (60FE_h)** ein- oder ausgeschaltet wird, wird auch das Objekt **digital_outputs_state_mapping** auf AUS (0) bzw. EIN (12) gesetzt.

Index	2420_h
Name	digital_outputs_state_mapping
Object Code	RECORD
No. of Elements	5

Ab Firmware 3.2.0.1.1

Sub-Index	01_h
Description	dig_out_state_mapp_dout_1
Data Type	UINT8
Access	rw
PDO Mapping	no
Units	--
Value Range	0 ... 16, siehe Tabelle
Default Value	0

Ab Firmware 3.2.0.1.1

Sub-Index	02_h
Description	dig_out_state_mapp_dout_2
Data Type	UINT8
Access	rw
PDO Mapping	no
Units	--
Value Range	0 ... 16, siehe Tabelle
Default Value	0

Ab Firmware 3.2.0.1.1

Sub-Index	03_h
Description	dig_out_state_mapp_dout_3
Data Type	UINT8
Access	rw
PDO Mapping	no
Units	--
Value Range	0... 16, siehe Tabelle
Default Value	0

Ab Firmware 3.2.0.1.1

Wert	Bezeichnung	Wert	Bezeichnung
0	Aus (Ausgang ist Low)	9	Unterspannung Zwischenkreis
1	Position $X_{soll} = X_{ziel}$	10	Feststellbremse gelüftet
2	Position $X_{ist} = X_{ziel}$	11	Endstufe aktiv
3	Reserviert	12	Ein (Ausgang ist High)
4	Restweg	13	Reserviert
5	Referenzfahrt aktiv	14	Reserviert
6	Vergleichsdrehzahl erreicht	15	Linearmotor identifiziert
7	I ² t-Überwachung aktiv	16	Referenzposition gültig
8	Schleppfehler		

Sub-Index	11_h
Description	dig_out_state_mapp_ea88_0_low
Data Type	UINT32
Access	rw
PDO Mapping	no
Units	--
Value Range	0 ... FFFFFFFF _h , siehe Tabelle
Default Value	0

Ab Firmware 3.2.0.1.1

Bit	Maske	Name	Bezeichnung
0 ... 7	000000FF _h	EA88_0_dout_0_mapping	Funktion für EA88 0 DOUT1
8 ... 15	0000FF00 _h	EA88_0_dout_1_mapping	Funktion für EA88 0 DOUT2
16 ... 23	00FF0000 _h	EA88_0_dout_2_mapping	Funktion für EA88 0 DOUT3
23 ... 31	FF000000 _h	EA88_0_dout_3_mapping	Funktion für EA88 0 DOUT4

Sub-Index	12_h
Description	dig_out_state_mapp_ea88_0_high
Data Type	UINT32
Access	rw
PDO Mapping	no
Units	--
Value Range	0 ... FFFFFFFF _h , siehe Tabelle
Default Value	0

Ab Firmware 3.2.0.1.1

Bit	Maske	Name	Bezeichnung
0 ... 7	000000FF _h	EA88_0_dout_4_mapping	Funktion für EA88 0 DOUT5
8 ... 15	0000FF00 _h	EA88_0_dout_5_mapping	Funktion für EA88 0 DOUT6
16 ... 23	00FF0000 _h	EA88_0_dout_6_mapping	Funktion für EA88 0 DOUT7
23 ... 31	FF000000 _h	EA88_0_dout_7_mapping	Funktion für EA88 0 DOUT8

6.14 Endschalter / Referenzschalter

6.14.1 Übersicht

Für die Definition der Referenzposition des Antriebreglers können wahlweise Endschalter (limit switch) oder Referenzschalter (homing switch) verwendet werden. Nähere Informationen zu den möglichen Referenzfahrt-Methoden finden Sie im Abschnitt 8.2: *Betriebsart Referenzfahrt (Homing Mode)*, Seite 179.

6.14.2 Beschreibung der Objekte

Index	Objekt	Name	Typ	Attr.
6510 _h	RECORD	drive_data		rw

6.14.2.1 Objekt 6510_h_11_h: limit_switch_polarity

Die Polarität der Endschalter kann durch das Objekt **6510_h_11_h (limit_switch_polarity)** programmiert werden. Für öffnende Endschalter ist in dieses Objekt eine Null, bei der Verwendung von schließenden Kontakten ist eine Eins einzutragen.

Index	6510_h
Name	drive_data
Object Code	RECORD
No. of Elements	51

Sub-Index	11_h
Description	limit_switch_polarity
Data Type	INT16
Access	rw
PDO Mapping	no
Units	--
Value Range	0, 1
Default Value	1

Wert	Bedeutung
0	Öffner
1	Schließer

6.14.2.2 Objekt 6510_h_12_h: limit_switch_selector

Über das Objekt 6510_h_12_h (**limit_switch_selector**) kann die Zuordnung der Endschalter (negativ, positiv) vertauscht werden, ohne Änderungen an der Verkabelung vornehmen zu müssen. Um die Zuordnung der Endschalter zu tauschen, ist eine Eins einzutragen.

Sub-Index	12_h
Description	limit_switch_selector
Data Type	INT16
Access	rw
PDO Mapping	no
Units	--
Value Range	0, 1
Default Value	0

Ab Firmware 3.5.x.1.1

Wert	Bedeutung
0	DIN6 = E0 (Endschalter negativ) DIN7 = E1 (Endschalter positiv)
1	DIN6 = E1 (Endschalter positiv) DIN7 = E0 (Endschalter negativ)

6.14.2.3 Objekt 6510_h_14_h: homing_switch_polarity

Die Polarität des Referenzschalters kann durch das Objekt 6510_h_14_h (**homing_switch_polarity**) programmiert werden. Für einen öffnenden Referenzschalter ist in dieses Objekt eine Null, bei der Verwendung von schließenden Kontakten ist eine Eins einzutragen.

Sub-Index	14_h
Description	homing_switch_polarity
Data Type	INT16
Access	rw
PDO Mapping	no
Units	--
Value Range	0, 1
Default Value	1

Wert	Bedeutung
0	Öffner
1	Schließer

6.14.2.4 Objekt 6510_h_13_h: homing_switch_selector

Das Objekt 6510_h_13_h (**homing_switch_selector**) legt fest, ob DIN8 oder DIN9 als Referenzschalter verwendet werden soll.

Sub-Index	13_h
Description	homing_switch_selector
Data Type	INT16
Access	rw
PDO Mapping	no
Units	--
Value Range	0, 1
Default Value	0

Wert	Bedeutung
0	DIN9
1	DIN8

6.14.2.5 Objekt 6510_h_15_h: limit_switch_deceleration

Das Objekt **limit_switch_deceleration** legt die Beschleunigung fest, mit der gebremst wird, wenn während des normalen Betriebs der Endschalter erreicht wird (Endschalter-Nothalt-Rampe).

Sub-Index	15_h
Description	limit_switch_deceleration
Data Type	INT32
Access	rw
PDO Mapping	no
Units	acceleration units
Value Range	0...3000000 min ⁻¹ /s
Default Value	2000000 min ⁻¹ /s

6.15 Sampling von Positionen

6.15.1 Übersicht

Die SERVOTEC S2 Familie bietet die Möglichkeit den Lageistwert auf der steigenden oder fallenden Flanke eines digitalen Eingangs hin abzuspeichern. Dieser Lagewert kann dann z.B. zur Berechnung innerhalb einer Steuerung ausgelesen werden.

Alle notwendigen Objekte sind in dem Record **sample_data** zusammengefasst: Das Objekt **sample_mode** legt die Art des Samplings fest: Soll nur ein einmaliges Sample- Ereignis aufgezeichnet werden oder soll kontinuierlich gesampelt werden ? Über das Objekt **sample_status** kann die Steuerung abfragen, ob ein Sample- Ereignis aufgetreten ist. Dies wird durch ein gesetztes Bit signalisiert, welches ebenfalls im **statusword** angezeigt werden kann, wenn das Objekt **sample_status_mask** entsprechend gesetzt ist.

Das Objekt **sample_control** dient dazu, die Freigabe des Sample- Ereignisses zu steuern und letztlich können über die Objekte **sample_position_rising_edge** und **sample_position_falling_edge** die gesampelten Positionen ausgelesen werden.

Welcher digitale Eingang verwendet wird, lässt sich mit dem S2Commander unter Parameter / IOs / Digitale Eingänge / Sample-Eingang festlegen.

6.15.2 Beschreibung der Objekte

6.15.2.1 In diesem Kapitel behandelte Objekte

Index	Objekt	Name	Typ	Attr.
204A _h	RECORD	sample_data		ro
204A _{h_01h}	VAR	sample_mode	UINT16	rw
204A _{h_02}	VAR	sample_status	UINT8	ro
204A _{h_03h}	VAR	sample_status_mask	UINT8	rw
204A _{h_04h}	VAR	sample_control	UINT8	wo
204A _{h_05h}	VAR	sample_position_rising_edge	INT32	ro
204A _{h_06h}	VAR	sample_position_falling_edge	INT32	ro

6.15.2.2 Objekt 204A_h: sample_data

Index	204A_h
Name	sample_data
Object Code	RECORD
No. of Elements	6

Ab Firmware 3.2.0.1.1

Mit dem folgenden Objekt kann gewählt werden, ob auf jedes Auftreten eines Sample-Events die Position bestimmt werden soll (Kontinuierliches Sampling) oder ob das Sampling nach einem Sample-Ereignis gesperrt werden soll, bis das Sampling erneut freigegeben wird. Beachten Sie hierbei, dass auch bereits ein Prellen beide Flanken auslösen kann !

Sub-Index	01_h
Description	sample_mode
Data Type	UINT16
Access	rw
PDO Mapping	no
Units	--
Value Range	0 ... 1, siehe Tabelle
Default Value	0

Ab Firmware 3.2.0.1.1

Wert	Bezeichnung
0	Kontinuierliches Sampling
1	Autolock sampling

Das folgenden Objekt zeigt ein neues Sample-Ereignis an.

Sub-Index	02_h
Description	sample_status
Data Type	UINT8
Access	ro
PDO Mapping	yes
Units	--
Value Range	0 ... 3, siehe Tabelle
Default Value	0

Ab Firmware 3.2.0.1.1

Bit	Wert	Name	Beschreibung
0	01 _h	falling_edge_occurred	= 1: Neue Sample-Position (fallende Flanke)
1	02 _h	rising_edge_occurred	= 1: Neue Sample-Position (steigende Flanke)

Mit dem folgenden Objekt können die Bits des Objekts **sample_status** festgelegt werden, die auch zum Setzen von Bit 15 des **statusword** führen sollen. Dadurch ist im üblicherweise ohnehin zu übertragenden **statusword** die Information "Sample- Ereignis aufgetreten" vorhanden, so dass die Steuerung nur in diesem Fall das Objekt **sample_status** lesen muss, um ggf. festzustellen welche Flanke aufgetreten ist.

Sub-Index	03_h
Description	sample_status_mask
Data Type	UINT8
Access	rw
PDO Mapping	yes
Units	--
Value Range	0 ... 3, siehe Tabelle
Default Value	0

Ab Firmware 3.2.0.1.1

Bit	Wert	Name	Beschreibung
0	01 _h	falling_edge_visible	Wenn falling_edge_occured = 1 => Statuswort Bit 15 = 1
1	02 _h	rising_edge_visible	Wenn rising_edge_occured = 1 => Statuswort Bit 15 = 1

Das Setzen des jeweiligen Bits in **sample_control** setzt zum einen das entsprechende Statusbit in **sample_status** zurück und schaltet im Falle des "Autolock"-Samplings das Sampling wieder frei.

Sub-Index	04_h
Description	sample_control
Data Type	UINT8
Access	wo
PDO Mapping	yes
Units	--
Value Range	0 ... 3, siehe Tabelle
Default Value	0

Ab Firmware 3.2.0.1.1

Bit	Wert	Name	Beschreibung
0	01 _h	falling_edge_enable	Sampling bei fallender Flanke
1	02 _h	rising_edge_enable	Sampling bei steigender Flanke

Die folgenden Objekte enthalten die gesampelten Positionen.

Sub-Index	05_h
Description	sample_position_rising_edge
Data Type	INT32
Access	ro
PDO Mapping	yes
Units	position units
Value Range	--
Default Value	--

Ab Firmware 3.2.0.1.1

Sub-Index	06_h
Description	sample_position_falling_edge
Data Type	INT32
Access	ro
PDO Mapping	yes
Units	position units
Value Range	--
Default Value	--

Ab Firmware 3.2.0.1.1

6.16 Bremsen-Ansteuerung

6.16.1 Übersicht

Mittels der nachfolgenden Objekte kann parametrierbar werden, wie der Regler eine eventuell im Motor integrierte Haltebremse ansteuert. Die Haltebremse wird immer freigeschaltet, sobald die Reglerfreigabe eingeschaltet wird. Für Haltebremsen mit hoher mechanischer Trägheit kann eine Verzögerungszeit parametrierbar werden, damit die Haltebremse in Eingriff ist, bevor die Endstufe ausgeschaltet wird (Durchsacken vertikaler Achsen). Diese Verzögerung wird durch das Objekt **brake_delay_time** parametrierbar. Wie aus der Skizze zu entnehmen ist, wird bei Einschalten der Reglerfreigabe der Drehzahl-Sollwert erst nach der **brake_delay_time** freigegeben und bei Ausschalten der Reglerfreigabe das Abschalten der Regelung um diese Zeit verzögert.

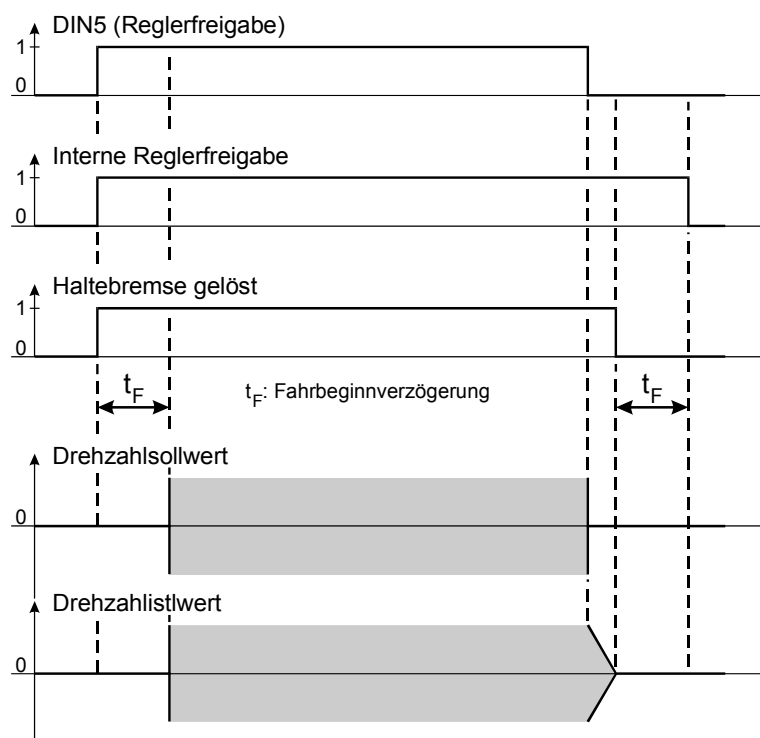


Abbildung 6.23: Funktion der Bremsverzögerung (bei Drehzahlregelung / Positionieren)

6.16.2 Beschreibung der Objekte

Index	Objekt	Name	Typ	Attr.
6510 _h	RECORD	drive_data		rw

6.16.2.1 Objekt 6510_h_18_h: brake_delay_time

Über das Objekt **brake_delay_time** kann die Bremsverzögerungszeit parametrierbar werden.

Index	6510_h
Name	drive_data
Object Code	RECORD
No. of Elements	51

Sub-Index	18_h
Description	brake_delay_time
Data Type	UINT16
Access	rw
PDO Mapping	no
Units	ms
Value Range	0...32000
Default Value	0

6.17 Geräteinformationen

Index	Objekt	Name	Typ	Attr.
1018 _h	RECORD	identity_object		rw
6510 _h	RECORD	drive_data		rw

Über zahlreiche CAN-Objekte können die verschiedensten Informationen wie Reglertyp, verwendete Firmware, etc. aus dem Gerät ausgelesen werden.

6.17.1 Beschreibung der Objekte

6.17.1.1 Objekt 1018_h: identity_object

Über das in der DS301 festgelegte **identity_object** kann der Regler in einem CANopen-Netzwerk eindeutig identifiziert werden. Zu diesem Zweck kann der Herstellercode (**vendor_id**), ein eindeutiger Produktcode (**product_code**), die Revisionsnummer der CANopen-Implementation (**revision_number**) und die Seriennummer des Geräts (**serial_number**) ausgelesen werden.

Index	1018_h
Name	identity_object
Object Code	RECORD
No. of Elements	4

Sub-Index	01_h
Description	vendor_id
Data Type	UINT32
Access	ro
PDO Mapping	no
Units	--
Value Range	000000E4
Default Value	000000E4

Sub-Index	02_h
Description	product_code
Data Type	UINT32
Access	ro
PDO Mapping	no
Units	--
Value Range	S.U.
Default Value	S.U.

Wert	Bedeutung
2005 _h	SERVOTEC S2 2102
2006 _h	SERVOTEC S2 2105
2009 _h	SERVOTEC S2 2302
200A _h	SERVOTEC S2 2305
200B _h	SERVOTEC S2 2310
200C _h	SERVOTEC S2 2320
200D _h	SERVOTEC S2 2340

Sub-Index	03_h
Description	revision_number
Data Type	UINT32
Access	ro
PDO Mapping	no
Units	MMMMSSSS _h (M: main version, S: sub version)
Value Range	--
Default Value	--

Sub-Index	04_h
Description	serial_number
Data Type	UINT32
Access	ro
PDO Mapping	no
Units	--
Value Range	--
Default Value	--

6.17.1.2 Objekt 6510_h_A0_h: drive_serial_number

Über das Objekt **drive_serial_number** kann die Seriennummer des Reglers ausgelesen werden. Dieses Objekt dient der Kompatibilität zu früheren Versionen.

Index	6510_h
Name	drive_data
Object Code	RECORD
No. of Elements	51

Sub-Index	A0_h
Description	drive_serial_number
Data Type	UINT32
Access	ro
PDO Mapping	no
Units	--
Value Range	--
Default Value	--

6.17.1.3 Objekt 6510_h_A1_h: drive_type

Über das Objekt **drive_type** kann der Gerätetyp des Reglers ausgelesen werden. Dieses Objekt dient der Kompatibilität zu früheren Versionen.

Sub-Index	A1_h
Description	drive_type
Data Type	UINT32
Access	ro
PDO Mapping	no
Units	--
Value Range	siehe 1018 _h _02 _h , product_code
Default Value	siehe 1018 _h _02 _h , product_code

6.17.1.4 Objekt 6510_n_A9_n: firmware_main_version

Über das Objekt **firmware_main_version** kann die Hauptversionsnummer der Firmware (Produktstufe) ausgelesen werden.

Sub-Index	A9_n
Description	firmware_main_version
Data Type	UINT32
Access	ro
PDO Mapping	no
Units	MMMMSSSS _n (M: main version, S: sub version)
Value Range	--
Default Value	--

6.17.1.5 Objekt 6510_n_AA_n: firmware_custom_version

Über das Objekt **firmware_custom_version** kann die Versionsnummer der kunden-spezifischen Variante der Firmware ausgelesen werden.

Sub-Index	AA_n
Description	firmware_custom_version
Data Type	UINT32
Access	ro
PDO Mapping	no
Units	MMMMSSSS _n (M: main version, S: sub version)
Value Range	--
Default Value	--

6.17.1.6 Objekt 6510_n_AD_n: km_release

Über die Versionsnummer des **km_release** können Firmwarestände der gleichen Produktstufe unterschieden werden.

Sub-Index	AD_n
Description	km_release
Data Type	UINT32
Access	ro
PDO Mapping	no
Units	--
Value Range	MMMMSSSS _n (M: main version, S: sub version)
Default Value	--

Ab Firmware 3.5.x.1.1

6.17.1.7 Objekt 6510_h_AC_h: firmware_type

Über das Objekt **firmware_type** kann ausgelesen werden, für welche Gerätefamilie und für welchen Winkelgeber typ die geladene Firmware geeignet ist. Da bei der SERVOTEC S2 - Familie das Winkelgeber-Interface nicht mehr steckbar ist, sind im Parameter G grundsätzlich alle Bits gesetzt (F_h).

Sub-Index	AC_h
Description	firmware_type
Data Type	UINT32
Access	ro
PDO Mapping	no
Units	00000GX _h
Value Range	00000F2 _h
Default Value	00000F2 _h

Wert (X)	Bedeutung
0 _h	IMD-F
1 _h	ARS
2 _h	SERVOTEC S2

6.17.1.8 Objekt 6510_h_B0_h: cycletime_current_controller

Über das Objekt **cycletime_current_controller** kann die Zykluszeit des Stromreglers in Mikrosekunden ausgelesen werden.

Sub-Index	B0_h
Description	cycletime_current_controller
Data Type	UINT32
Access	ro
PDO Mapping	no
Units	µS
Value Range	--
Default Value	00000068 _h

6.17.1.9 Objekt 6510_h_B1_h: cycletime_velocity_controller

Über das Objekt **cycletime_velocity_controller** kann die Zykluszeit des Drehzahlreglers in Mikrosekunden ausgelesen werden.

Sub-Index	B1_h
Description	cycletime_velocity_controller
Data Type	UINT32
Access	ro
PDO Mapping	no
Units	µs
Value Range	--
Default Value	00000D0 _h

6.17.1.10 Objekt 6510_h_B2_h: cycletime_position_controller

Über das Objekt **cycletime_position_controller** kann die Zykluszeit des Lagereglers in Mikrosekunden ausgelesen werden.

Sub-Index	B2_h
Description	cycletime_position_controller
Data Type	UINT32
Access	ro
PDO Mapping	no
Units	µs
Value Range	--
Default Value	000001A0 _h

6.17.1.11 Objekt 6510_h_B3_h: cycletime_trajectory_generator

Über das Objekt **cycletime_trajectory_generator** kann die Zykluszeit der Positionier-Steuerung in Mikrosekunden ausgelesen werden.

Sub-Index	B3_h
Description	cycletime_tracectory_generator
Data Type	UINT32
Access	ro
PDO Mapping	no
Units	µs
Value Range	--
Default Value	00000341 _h

6.17.1.12 Objekt 6510_h_C0_h: commissioning_state

Das Objekt **commissioning_state** wird von der Parametrier-Software S2Commander beschrieben, wenn bestimmte Parametrierungen durchgeführt worden sind (z.B. des Nennstroms). Nach der Auslieferung und nach **restore_default_parameter** enthält dieses Objekt eine Null. In diesem Fall wird auf dem 7-Segment-Display des Antriebsreglers ein „A“ angezeigt, um darauf hinzuweisen, dass dieses Gerät noch nicht parametriert wurde. Wenn der Regler komplett unter CANopen parametriert wird, muss mindestens ein Bit in diesem Objekt gesetzt werden, um die Anzeige „A“ zu unterdrücken. Natürlich ist es bei Bedarf auch möglich, dieses Objekt zu nutzen, um sich den Zustand der Reglerparametrierung zu merken. Beachten Sie in diesem Fall, dass der S2Commander ebenfalls auf dieses Objekt zugreift.

Sub-Index	C0 _h
Description	commissioning_state
Data Type	UINT32
Access	rw
PDO Mapping	no
Units	--
Value Range	--
Default Value	0

Bit	Bedeutung	Bit	Bedeutung
0	Nennstrom gültig	8	Stromregler-Parameter gültig
1	Maximalstrom gültig	9	Reserviert
2	Polzahl des Motors gültig	10	Physik. Einheiten gültig
3	Offsetwinkel / Drehsinn gültig	11	Drehzahlregler gültig
4	Reserviert	12	Lageregler gültig
5	Offsetwinkel / Drehsinn Hallgeber gültig	13	Sicherheitsparameter gültig
6	Reserviert	14	Reserviert
7	Absolutlage Gebersystem gültig	15	Endschalter-Polarität gültig
		16...31	Reserviert



Vorsicht !

Dieses Objekt enthält keinerlei Informationen darüber, ob der Regler dem Motor und der Applikation entsprechend **richtig** parametriert wurde, sondern nur, ob die genannten Punkte nach der Auslieferung mindestens einmal überhaupt parametriert wurden.



„A“ im 7-Segment-Display

Beachten Sie, dass mindestens ein Bit im Objekt commissioning_state gesetzt werden muss, um das „A“ auf dem Displays Ihres Reglers zu unterdrücken.

6.18 Fehlermanagement

6.18.1 Übersicht

Die Servoregler der SERVOTEC S2-Familie bieten die Möglichkeit, die Fehlerreaktion einzelner Ereignisse, wie z.B. das Auftreten eines Schleppfehlers, zu ändern. Dadurch reagiert der Regler unterschiedlich, wenn ein bestimmtes Ereignis eintritt: So kann je nach Einstellung heruntergebremst werden, die Einstufe sofort ausgeschaltet werden aber auch lediglich eine Warnung auf dem Display angezeigt werden.

Für jedes Ereignis ist herstellereitig eine Mindestreaktion vorgesehen, die nicht unterschritten werden kann. So lassen sich „kritische“ Fehler wie beispielsweise 06-0 Kurzschluss Endstufe nicht umparametrieren, da hier eine sofortige Abschaltung notwendig ist, um den Servoregler vor einer eventuellen Zerstörung zu schützen.

Wird eine niedrigere Fehlerreaktion als für den jeweiligen Fehler zulässig eingetragen, wird der Wert auf die niedrigste zulässige Fehlerreaktion begrenzt. Eine Liste aller Fehlernummern befindet sich im Softwarehandbuch „Servopositionierregler SERVOTEC S2“.

6.18.2 Beschreibung der Objekte

6.18.2.1 In diesem Kapitel behandelte Objekte

Index	Objekt	Name	Typ	Attr.
2100 _h	RECORD	error_management		ro
2100 _{h_01_h}	VAR	error_number	UINT8	rw
2100 _{h_02_h}	VAR	error_reaction_code	UINT8	rw
200F _h	VAR	last_warning_code	UINT16	ro

6.18.2.2 Objekt 2100_h: error_management

Index	2100_h
Name	error_management
Object Code	RECORD
No. of Elements	2

Ab Firmware 3.2.0.1.1

Im Objekt **error_number** muss die Hauptfehlnummer angegeben werden, deren Reaktion geändert werden soll. Die Hauptfehlnummer ist in der Regel vor dem Bindestrich angegeben (z.B. Fehler 08-2, Hauptfehlnummer 8).

Sub-Index	01_h
Description	error_number
Data Type	UINT8
Access	rw
PDO Mapping	no
Units	--
Value Range	1 ... 96
Default Value	1

Ab Firmware 3.2.0.1.1

Im Objekt **error_reaction_code** kann die Reaktion des Fehlers verändert werden. Wird die herstellerseitige Mindestreaktion unterschritten, wird auf diese begrenzt. Die wirklich eingestellte Reaktion kann durch Rücklesen bestimmt werden.

Sub-Index	02_h
Description	error_reaction_code
Data Type	UINT8
Access	rw
PDO Mapping	no
Units	--
Value Range	0, 1, 3, 5, 7, 8
Default Value	hängt von error_number ab

Ab Firmware 3.2.0.1.1

Wert	Bedeutung
0	Keine Aktion
1	Eintrag im Puffer
3	Warnung auf dem 7-Segment-Display
5	Reglerfreigabe aus
7	Bremsen mit Maximalstrom
8	Endstufe aus

6.18.2.3 Objekt 200F_h: last_warning_code

Warnungen sind bemerkenswerte Ereignisse des Antriebs (z.B. ein Schleppfehler), die im Gegensatz zu einem Fehler nicht zum Stillsetzen des Antriebs führen sollen. Warnungen werden auf der 7-Segmentanzeige des Reglers angezeigt und danach automatisch vom Regler zurückgesetzt.

Die letzte aufgetretene Warnung kann über das folgende Objekt ausgelesen werden: Dabei zeigt Bit 15 an, ob die Warnung aktuell noch aktiv ist.

Index	200F _h
Name	last_warning_code
Object Code	VAR
Data Type	UINT16

Ab Firmware 3.5.x.1.1

Access	ro
PDO Mapping	yes
Units	--
Value Range	--
Default Value	--

Bit	Wert	Beschreibung
0... 3	000F _h	Unternummer der Warnung
4... 11	0FF0 _h	Hauptnummer der Warnung
15	8000 _h	Warnung ist aktiv

7 Gerätesteuerung (Device Control)

7.1 Zustandsdiagramm (State Machine)

7.1.1 Übersicht

Das nachfolgende Kapitel beschreibt, wie der Regler unter CANopen gesteuert wird, also wie beispielsweise die Endstufe eingeschaltet oder ein Fehler quittiert wird.

Unter CANopen wird die gesamte Steuerung des Reglers über zwei Objekte realisiert: Über das **controlword** kann der Host den Regler steuern, während der Status des Reglers im Objekt **statusword** zurückgelesen werden kann. Zur Erklärung der Reglersteuerung werden die folgenden Begriffe verwendet:

Zustand: (State)	Je nachdem ob beispielsweise die Endstufe eingeschaltet oder ein Fehler aufgetreten ist befindet sich der Regler in verschiedenen Zuständen. Die unter CANopen definierten Zustände werden im Laufe des Kapitels vorgestellt. Beispiel: SWITCH_ON_DISABLED
Zustandsübergang (State Transition)	Ebenso wie die Zustände ist es unter CANopen ebenfalls definiert, wie man von einem Zustand zu einem anderen gelangt (z.B. um einen Fehler zu quittieren). Zustandsübergänge werden vom Host durch Setzen von Bits im controlword ausgelöst oder intern durch den Regler, wenn dieser beispielsweise einen Fehler erkennt.
Kommando (Command)	Zum Auslösen von Zustandsübergängen müssen bestimmte Kombinationen von Bits im controlword gesetzt werden. Eine solche Kombination wird als Kommando bezeichnet. Beispiel: Enable Operation
Zustandsdiagramm (State Machine)	Die Zustände und Zustandsübergänge bilden zusammen das Zustandsdiagramm, also die Übersicht über alle Zustände und die von dort möglichen Übergänge.

7.1.2 Das Zustandsdiagramm des Reglers (State Machine)

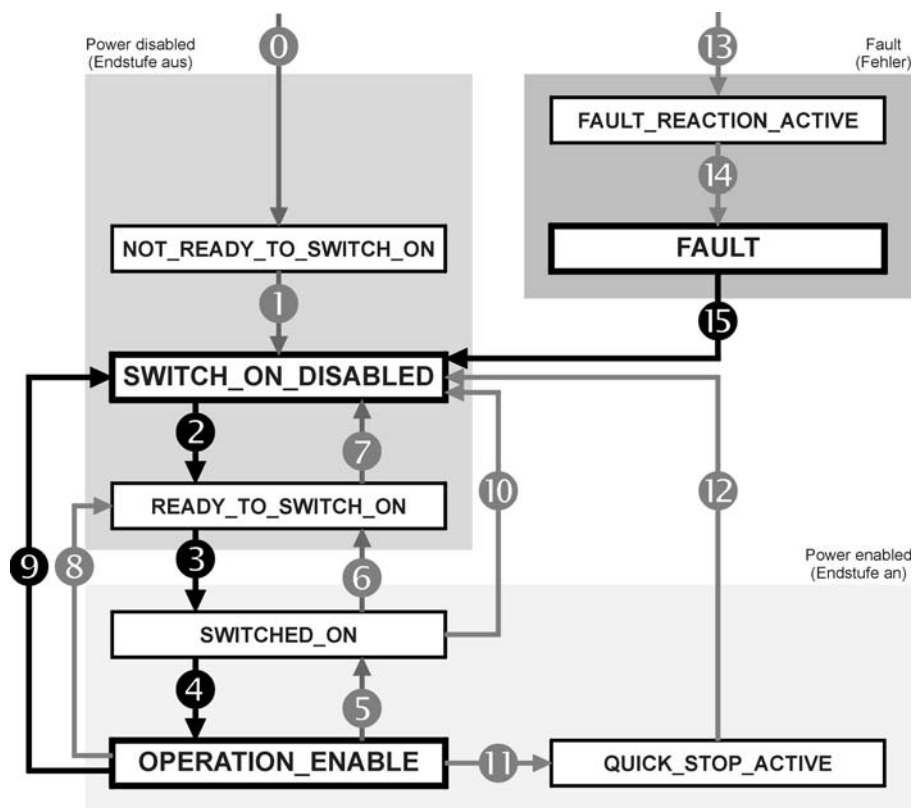


Abbildung 7.24: Zustandsdiagramm des Reglers

Das Zustandsdiagramm kann grob in drei Bereiche aufgeteilt werden: „Power Disabled“ bedeutet, dass die Endstufe ausgeschaltet ist und „Power Enabled“ dass die Endstufe eingeschaltet ist. Im Bereich „Fault“ sind die zur Fehlerbehandlung notwendigen Zustände zusammengefasst.

Die wichtigsten Zustände des Reglers sind im Diagramm hervorgehoben dargestellt. Nach dem Einschalten initialisiert sich der Regler und erreicht schließlich den Zustand **SWITCH_ON_DISABLED**. In diesem Zustand ist die CAN-Kommunikation voll funktionsfähig und der Regler kann parametrieren (z.B. die Betriebsart „Drehzahlregelung“ eingestellt werden). Die Endstufe ist ausgeschaltet und die Welle ist somit frei drehbar. Durch die Zustandsübergänge 2, 3, 4 – was im Prinzip der CAN-Reglerfreigabe entspricht – gelangt man in den Zustand **OPERATION_ENABLE**. In diesem Zustand ist die Endstufe eingeschaltet und der Motor wird gemäß der eingestellten Betriebsart geregelt. Stellen Sie daher vorher unbedingt sicher, dass der Antrieb richtig parametrieren ist und ein entsprechender Sollwert gleich Null ist. Der Zustandsübergang 9 entspricht der Wegnahme der Freigabe, d.h. ein noch laufender Motor würde ungeregelt austrudeln.

Tritt ein Fehler auf so wird (egal aus welchem Zustand) letztlich in den Zustand **FAULT** verzweigt. Je nach Schwere des Fehlers können vorher noch bestimmte Aktionen, wie z.B. eine Notbremsung ausgeführt werden (**FAULT_REACTION_ACTIVE**).

Um die genannten Zustandsübergänge auszuführen müssen bestimmte Bitkombinationen im **controlword** (siehe unten) gesetzt werden. Die unteren 4 Bits des **controlwords** werden gemeinsam ausgewertet, um einen Zustandsübergang auszulösen. Im Folgenden werden zunächst nur die wichtigsten Zustandsübergänge 2, 3, 4, 9 und 15 erläutert. Eine Tabelle aller möglichen Zustände und Zustandsübergänge findet sich am Ende dieses Kapitels.

Die folgende Tabelle enthält in der 1. Spalte den gewünschten Zustandsübergang und in der 2. Spalte die dazu notwendigen Voraussetzungen (Meistens ein Kommando durch den Host, hier mit Rahmen dargestellt). Wie dieses Kommando erzeugt wird, d.h. welche Bits im **controlword** zu setzen sind, ist in der 3. Spalte ersichtlich (x = nicht relevant).


Nr.	Wird durchgeführt wenn	Bitkombination (controlword)				Aktion	
		Bit 3	2	1	0		
2	Endstufen- u. Reglerfreig. vorh. + Kommando Shutdown	Shutdown	= x	1	1	0	Keine
3	Kommando Switch On	Switch On	= x	1	1	1	Einschalten der Endstufe
4	Kommando Enable Operation	Enable Operation	= 1	1	1	1	Regelung gemäß eingestellter Betriebsart
9	Kommando Disable Voltage	Disable Voltage	= x	x	0	x	Endstufe wird gesperrt. Motor ist frei drehbar.
15	Fehler behoben+ Kommando Fault Reset	Fault Reset	=	Bit 7 = 			Fehler quittieren

Abbildung 7.25: Wichtigste Zustandsübergänge des Reglers

BEISPIEL



Nachdem der Regler parametrierung wurde, soll der Regler „freigegeben“, d.h. die Endstufe eingeschaltet werden:

- 1.) Der Regler ist im Zustand **SWITCH_ON_DISABLED**
- 2.) Der Regler soll in den Zustand **OPERATION_ENABLE**
- 3.) Laut Zustandsdiagramm (Abbildung 7.24) sind die Übergänge 2, 3 und 4 auszuführen.
- 4.) Aus Abbildung 7.25 folgt:

Übergang 2: controlword = 0006_h Neuer Zustand: **READY_TO_SWITCH_ON** *¹⁾

Übergang 3: controlword = 0007_h Neuer Zustand: **SWITCHED_ON** *¹⁾

Übergang 4: controlword = 000F_h Neuer Zustand: **OPERATION_ENABLE** *¹⁾

Hinweise:

- 1.) Das Beispiel geht davon aus, dass keine weiteren Bits im **controlword** gesetzt sind (Für die Übergänge sind ja nur die Bits 0..3 wichtig).
- 2.) Die Übergänge 3 und 4 können zusammengefasst werden, indem das **controlword** gleich auf 000F_h gesetzt wird. Für den Zustandsübergang 2 ist das gesetzte Bit 3 nicht relevant.

*¹⁾ Der Host muss warten, bis der Zustand im **statusword** zurückgelesen werden kann. Dieses wird weiter unten noch ausführlich erläutert.

7.1.2.1 Zustandsdiagramm: Zustände

In der folgenden Tabelle sind alle Zustände und deren Bedeutung aufgeführt:


Name	Bedeutung
NOT_READY_TO_SWITCH_ON	Der Regler führt einen Selbsttest durch. Die CAN-Kommunikation arbeitet noch nicht.
SWITCH_ON_DISABLED	Der Regler hat seinen Selbsttest abgeschlossen. CAN-Kommunikation ist möglich.
READY_TO_SWITCH_ON	Der Regler wartet bis die digitalen Eingänge „Endstufen-“ und „Reglerfreigabe“ an 24 V liegen. (Reglerfreigabelogik „Digitaler Eingang und CAN“).
SWITCHED_ON ^{*1)}	Die Endstufe ist eingeschaltet.
OPERATION_ENABLE ^{*1)}	Der Motor liegt an Spannung und wird entsprechend der Betriebsart geregelt.
QUICKSTOP_ACTIVE ^{*1)}	Die Quick Stop Function wird ausgeführt (siehe: quick_stop_option_code). Der Motor liegt an Spannung und wird entsprechend der Quick Stop Function geregelt.
FAULT_REACTION_ACTIVE ^{*1)}	Es ist ein Fehler aufgetreten. Bei kritischen Fehlern wird sofort in den Status Fault gewechselt. Ansonsten wird die im fault_reaction_option_code vorgegebene Aktion ausgeführt. Der Motor liegt an Spannung und wird entsprechend der Fault Reaction Function geregelt.
FAULT	Es ist ein Fehler aufgetreten. Der Motor ist spannungsfrei.

^{*1)}Die Endstufe ist eingeschaltet.

7.1.2.2 Zustandsdiagramm: Zustandsübergänge

In der folgenden Tabelle sind alle Zustände und deren Bedeutung aufgeführt:

Nr.	Wird durchgeführt wenn	Bitkombination (controlword)				Aktion	
		Bit 3	2	1	0		
0	Eingeschaltet o. Reset erfolgt	interner Übergang				Selbsttest ausführen	
1	Selbsttest erfolgreich	interner Übergang				Aktivierung der CAN-Kommunikation	
2	Endstufen- u. Reglerfreig. vorh. + Kommando Shutdown	Shutdown	= x	1	1	0	-
3	Kommando Switch On	Switch On	= x	1	1	1	Einschalten der Endstufe
4	Kommando Enable Operation	Enable Operation	= 1	1	1	1	Regelung gemäß eingestellter Betriebsart
5	Kommando Disable Operation	Disable Operation	= 0	1	1	1	Endstufe wird gesperrt. Motor ist frei drehbar
6	Kommando Shutdown	Shutdown	= x	1	1	0	Endstufe wird gesperrt. Motor ist frei drehbar

Nr.	Wird durchgeführt wenn	Bitkombination (controlword)					Aktion
		Bit	3	2	1	0	
7	Kommando Quick Stop	Quick Stop	= x	0	1	x	-
8	Kommando Shutdown	Shutdown	= x	1	1	0	Endstufe wird gesperrt. Motor ist frei drehbar
9	Kommando Disable Voltage	Disable Voltage	= x	x	0	x	Endstufe wird gesperrt. Motor ist frei drehbar.
10	Kommando Disable Voltage	Disable Voltage	= x	x	0	x	Endstufe wird gesperrt. Motor ist frei drehbar
11	Kommando Quick Stop	Quick Stop	= x	0	1	x	Es wird eine Bremsung gemäß quick_stop_option_code eingeleitet.
12	Bremsung beendet o. Kommando Disable Voltage	Disable Voltage	= x	x	0	x	Endstufe wird gesperrt. Motor ist frei drehbar
13	Fehler aufgetreten	interner Übergang					Bei unkritischen Fehlern Reaktion gemäß fault_reaction_option_code . Bei kritischen Fehlern folgt Übergang 14
14	Fehlerbehandlung ist beendet	interner Übergang					Endstufe wird gesperrt. Motor ist frei drehbar
15	Fehler behoben+ Kommando Fault Reset	Fault Reset	=	Bit 7 = 			Fehler quittieren (bei steigender Flanke)



Endstufe gesperrt...

...bedeutet, dass die Leistungshalbleiter (Transistoren) nicht mehr angesteuert werden. **Wenn dieser Zustand bei einem drehenden Motor eingenommen wird, so trudelt dieser ungebremst aus.** Eine eventuell vorhandene mechanische Motorbremse wird hierbei automatisch angezogen.



Vorsicht: Das Signal garantiert nicht, dass der Motor wirklich spannungsfrei ist.



Endstufe freigegeben...

...bedeutet, dass der Motor entsprechend der gewählten Betriebsart angesteuert und geregelt wird. Eine eventuell vorhandene mechanische Motorbremse wird automatisch gelöst. Bei einem Defekt oder einer Fehlparametrierung (Motorstrom, Polzahl, Resolveroffsetwinkel etc.) kann es zu einem unkontrollierten Verhalten des Antriebes kommen.

7.1.3 controlword (Steuerwort)

7.1.3.1 Objekt 6040_h: controlword

Mit dem **controlword** kann der aktuelle Zustand des Reglers geändert bzw. direkt eine bestimmte Aktion (z.B. Start der Referenzfahrt) ausgelöst werden. Die Funktion der Bits 4, 5, 6 und 8 hängt von der aktuellen Betriebsart (**modes_of_operation**) des Reglers ab, die nach diesem Kapitel erläutert wird.

Index	6040 _h
Name	controlword
Object Code	VAR
Data Type	UINT16

Access	rw
PDO Mapping	yes
Units	--
Value Range	--
Default Value	0

Bit	Wert	Funktion
0	0001 _h	Steuerung der Zustandsübergänge. (Diese Bits werden gemeinsam ausgewertet)
1	0002 _h	
2	0004 _h	
3	0008 _h	
4	0010 _h	new_set_point / start_homing_operation / enable_ip_mode
5	0020 _h	change_set_immediatly
6	0040 _h	absolute / relative
7	0080 _h	reset_fault
8	0100 _h	halt
9	0200 _h	reserved set to 0
10	0400 _h	reserved set to 0
11	0800 _h	reserved set to 0
12	1000 _h	reserved set to 0
13	2000 _h	reserved set to 0
14	4000 _h	reserved set to 0
15	8000 _h	reserved set to 0

Tabelle 7.1: Bitbelegung des controlword

Wie bereits umfassend beschrieben können mit den Bits 0..3 Zustandsübergänge ausgeführt werden. Die dazu notwendigen Kommandos sind hier noch einmal in einer Übersicht dargestellt. Das Kommando **Fault Reset** wird durch einen positiven Flankenwechsel (von 0 nach 1) von Bit 7 erzeugt.


Kommando:	Bit 7	Bit 3	Bit 2	Bit 1	Bit 0
	0080 _h	0008 _h	0004 _h	0002 _h	0001 _h
Shutdown	×	×	1	1	0
Switch On	×	×	1	1	1
Disable Voltage	×	×	×	0	×
Quick Stop	×	×	0	1	×
Disable Operation	×	0	1	1	1
Enable Operation	×	1	1	1	1
Fault Reset		×	×	×	×

Tabelle 7.2: Übersicht aller Kommandos (× = nicht relevant)



Da einige Statusänderungen einen gewissen Zeitraum beanspruchen, müssen alle über das **controlword** ausgelösten Statusänderungen über das **statusword** zurückgelesen werden. Erst wenn der angeforderte Status auch im **statusword** gelesen werden kann, darf über das **controlword** ein weiteres Kommando eingeschrieben werden.

Nachfolgend sind die restlichen Bits des **controlwords** erläutert. Einige Bits haben dabei je nach Betriebsart (**modes_of_operation**), d.h. ob der Regler z.B. drehzahl- oder momenten-geregelt wird, unterschiedliche Bedeutung:

Bit 4		Abhängig von modes_of_operation :
	new_set_point	Im Profile Position Mode : Eine steigende Flanke signalisiert dem Regler, dass ein neuer Fahrauftrag übernommen werden soll. Siehe dazu auch Abschnitt 8.3: <i>Betriebsart Positionieren (Profile Position Mode)</i> , ab Seite 191.
	start_homing_operation	Im Homing Mode : Eine steigende Flanke bewirkt, dass die parametrisierte Referenzfahrt gestartet wird. Eine fallende Flanke bricht eine laufende Referenzfahrt vorzeitig ab.

	enable_ip_mode	Im Interpolated Position Mode: Dieses Bit muss gesetzt werden, wenn die Interpolations-Datensätze ausgewertet werden sollen. Es wird durch das Bit ip_mode_active im statusword quittiert. Siehe hierzu auch Abschnitt 8.4: <i>Interpolated Position Mode, ab Seite 198.</i>
--	-----------------------	--

Bit 5	change_set_immediatly	Nur im Profile Position Mode : Wenn dieses Bit nicht gesetzt ist, so wird bei einem neuen Fahrauftrag zuerst ein eventuell laufender abgearbeitet und erst dann mit dem neuen begonnen. Bei gesetztem Bit wird eine laufende Positionierung sofort abgebrochen und durch den neuen Fahrauftrag ersetzt. Siehe dazu unbedingt auch Abschnitt 8.3: <i>Betriebsart Positionieren (Profile Position Mode)</i> , ab Seite 191.
Bit 6	relative	Nur im Profile Position Mode : Bei gesetztem Bit bezieht der Regler die Zielposition (target_position) des aktuellen Fahrauftrages auf die Sollposition (position_demand_value) des Lagereglers.
Bit 7	reset_fault	Beim Übergang von Null auf Eins versucht der Regler die vorhandenen Fehler zu quittieren. Dies gelingt nur, wenn die Ursache für den Fehler behoben wurde.
Bit 8		Abhängig von modes_of_operation :
	halt	Im Profile Position Mode : Bei gesetztem Bit wird die laufende Positionierung abgebrochen. Gebremst wird hierbei mit der profile_deceleration . Nach Beendigung des Vorgangs wird im statusword das Bit target_reached gesetzt. Das Löschen des Bits hat keine Auswirkung.
	halt	Im Profile Velocity Mode : Bei gesetztem Bit wird die Drehzahl auf Null abgesenkt. Gebremst wird hierbei mit der profile_deceleration . Das Löschen des Bits bewirkt, dass der Regler wieder beschleunigt.
	halt	Im Profile Torque Mode : Bei gesetztem Bit wird das Drehmoment auf Null abgesenkt. Dies geschieht mit der torque_slope . Das Löschen des Bits bewirkt, dass der Regler wieder beschleunigt.
	halt	Im Homing Mode : Bei gesetztem Bit wird die laufende Referenzfahrt abgebrochen. Das Löschen des Bits hat keine Auswirkung.

7.1.4 Auslesen des Reglerzustands

Ähnlich wie über die Kombination mehrerer Bits des **controlwords** verschiedene Zustandsübergänge ausgelöst werden können, kann über die Kombination verschiedener Bits des **statusword** ausgelesen werden, in welchem Zustand sich der Regler befindet.

Die folgende Tabelle listet die möglichen Zustände des Zustandsdiagramms sowie die zugehörige Bitkombination auf, mit der sie im **statusword** angezeigt werden.

Zustand	Bit 6	Bit 5	Bit 3	Bit 2	Bit 1	Bit 0	Maske	Wert
	0040 _h	0020 _h	0008 _h	0004 _h	0002 _h	0001 _h		
NOT_READY_TO_SWITCH_ON	0	×	0	0	0	0	004F _h	0000 _h
SWITCH_ON_DISABLED	1	×	0	0	0	0	004F _h	0040 _h
READY_TO_SWITCH_ON	0	1	0	0	0	1	006F _h	0021 _h
SWITCHED_ON	0	1	0	0	1	1	006F _h	0023 _h
OPERATION_ENABLE	0	1	0	1	1	1	006F _h	0027 _h
QUICK_STOP_ACTIVE	0	0	0	1	1	1	006F _h	0007 _h
FAULT_REACTION_ACTIVE	0	×	1	1	1	1	004F _h	000F _h
FAULT	0	×	1	1	1	1	004F _h	000F _h
FAULT (gemäß DS402) ¹⁾	0	×	1	0	0	0	004F _h	0008 _h

Tabelle 7.3: Gerätestatus (× = nicht relevant)

1)



In bisherigen CANopen-Implementierungen wird der Zustand FAULT nicht gemäß DS 402 zurückgemeldet. Daher besteht die Möglichkeit über das Objekt **compatibility_control** (siehe Abschnitt 6.2.2.2: *Objekt 6510h_F0h: compatibility_control*, Seite 65) die Rückmeldung gemäß DS402 auszuwählen.

Für Kompatibilität zu früheren Firmwareversionen brauchen keine Änderungen durchgeführt werden !

BEISPIEL



Das obige Beispiel zeigt, welche Bits im **controlword** gesetzt werden müssen, um den Regler freizugeben. Jetzt soll dabei der neu eingeschriebene Zustand aus dem **statusword** ausgelesen werden:

Übergang von **SWITCH_ON_DISABLED** zu **OPERATION_ENABLE**:

- 1.) Zustandsübergang 2 ins **controlword** schreiben.
- 2.) Warten, bis der Zustand **READY_TO_SWITCH_ON** im **statusword** angezeigt wird.

Übergang 2: **controlword** = 0006_h Warten bis $(\text{statusword} \& 006F_h) = 0021_h$ ^{*1)}

- 3.) Zustandsübergang 3 und 4 können zusammengefasst ins **controlword** geschrieben werden.

- 4.) Warten, bis der Zustand **OPERATION_ENABLE** im **statusword** angezeigt wird.

Übergang 3+4: **controlword** = 000F_h Warten bis $(\text{statusword} \& 006F_h) = 0027_h$ ^{*1)}

Hinweis:

Das Beispiel geht davon aus, dass keine weiteren Bits im **controlword** gesetzt sind (Für die Übergänge sind ja nur die Bits 0..3 wichtig).

^{*1)}Für die Identifizierung der Zustände müssen auch nicht gesetzte Bits ausgewertet werden (siehe Tabelle). Daher muss das **statusword** entsprechend maskiert werden.

7.1.5 statuswords (Statusworte)


7.1.5.1 Objekt 6041_h: statusword

Index	6041 _h
Name	statusword
Object Code	VAR
Data Type	UINT16


Access	ro
PDO Mapping	yes
Units	--
Value Range	--
Default Value	--

Bit	Wertigkeit	Name
0	0001 _h	Zustand des Reglers (siehe <i>Tabelle 7.3, Seite 163</i>). (Diese Bits müssen gemeinsam ausgewertet werden)
1	0002 _h	
2	0004 _h	
3	0008 _h	
4	0010 _h	voltage_enabled
5	0020 _h	Zustand des Reglers (siehe <i>Tabelle 7.3, Seite 163</i>).
6	0040 _h	
7	0080 _h	Warning
8	0100 _h	drive_is_moving
9	0200 _h	remote
10	0400 _h	target_reached
11	0800 _h	internal_limit_active
12	1000 _h	set_point_acknowledge / speed_0 / homing_attained / ip_mode_active
13	2000 _h	following_error / homing_error
14	4000 _h	manufacturer_statusbit
15	8000 _h	trigger_result

Tabelle 7.4: Bitbelegung im statusword

	<p>Alle Bits des statusword sind nicht gepuffert. Sie repräsentieren den aktuellen Gerätestatus.</p>
---	---

Neben dem Reglerstatus werden im **statusword** diverse Ereignisse angezeigt, d.h. jedem Bit ist ein bestimmtes Ereignis wie z.B. Schleppfehler zugeordnet. Die einzelnen Bits haben dabei folgende Bedeutung:

Bit 4	voltage_enabled	
		<p>Dieses Bit ist gesetzt, wenn die Endstufentransistoren ausgeschaltet sind.</p> <p>Wenn im Objekt 6510_h_F0_h (compatibility_control) Bit 7 gesetzt ist, gilt (siehe Abschnitt 6.2: <i>Kompatibilitäts-Einstellungen, ab Seite 65</i>) ¹⁾:</p> <p>Dieses Bit ist gesetzt, wenn die Endstufentransistoren eingeschaltet sind.</p>
		<div style="display: flex; align-items: center;">  <div> <p>ACHTUNG: Bei einem Defekt kann der Motor trotzdem unter Spannung stehen.</p> </div> </div>
Bit 5	quick_stop	
		Bei gelöschtem Bit führt der Antrieb einen Quick Stop gemäß quick_stop_option_code aus.
Bit 7	warning	
		Dieses Bit zeigt an, dass eine Drehrichtung gesperrt ist, weil einer Endschalter ausgelöst wurde. Die Sollwertsperrung wird wieder gelöscht, wenn eine Fehlerquittierung durchgeführt wird (Siehe controlword, fault_reset)
Bit 8	drive_is_moving	herstellerspezifisch
		Dieses Bit wird – unabhängig von modes_of_operation – gesetzt, wenn sich die aktuelle Ist-Drehzahl (velocity_actual_value) des Antriebes außerhalb des zugehörigen Toleranzfenster befindet (velocity_threshold).
Bit 9	remote	
		Dieses Bit zeigt an, dass die Endstufe des Reglers über das CAN-Netzwerk freigegeben werden kann. Es ist gesetzt, wenn die Reglerfreigabelogik über das Objekt enable_logic entsprechend eingestellt ist.

1):



In bisherigen CANopen- Implementierungen wird Bit 4 (**voltage_enabled**) im Gegensatz zur Spezifikation in der DS 402 invertiert zurückgemeldet. Daher besteht die Möglichkeit über das Objekt **compatibility_control** (siehe Abschnitt 6.2: *Kompatibilitäts-Einstellungen, ab Seite 65*) die Rückmeldung gemäß DS402 auszuwählen.

Für Kompatibilität zu früheren Firmwareversionen brauchen keine Änderungen durchgeführt werden !

Bit 10		Abhängig von modes_of_operation :
	target_reached	<p>Im Profile Position Mode:</p> <p>Das Bit wird gesetzt, wenn die aktuelle Zielposition erreicht ist und sich die aktuelle Position (position_actual_value) im parametrisierten Positionsfenster (position_window) befindet.</p> <p>Außerdem wird es gesetzt, wenn der Antrieb bei gesetztem Halt-Bit zum Stillstand kommt.</p> <p>Es wird gelöscht, sobald ein neues Ziel vorgegeben wird.</p>
	target_reached	<p>Im Profile Velocity Mode:</p> <p>Das Bit wird gesetzt, wenn sich die Drehzahl (velocity_actual_value) des Antriebs im Toleranzfenster befindet (velocity_window, velocity_window_time).</p>
Bit 11	internal_limit_active	
		Dieses Bit zeigt an, dass die I ² t-Begrenzung aktiv ist.
Bit 12		Abhängig von modes_of_operation :
	set_point_acknowledge	<p>Im Profile Position Mode:</p> <p>Dieses Bit wird gesetzt, wenn der Regler das gesetzte Bit new_set_point im controlword erkannt hat. Es wird wieder gelöscht, nachdem das Bit new_set_point im controlword auf Null gesetzt wurde. Siehe dazu auch Abschnitt 8.3: <i>Betriebsart Positionieren (Profile Position Mode)</i>, ab Seite 191.</p>
	speed_0	<p>Im Profile Velocity Mode:</p> <p>Dieses Bit wird gesetzt, wenn sich die aktuelle Ist-Drehzahl (velocity_actual_value) des Antriebes im zugehörigen Toleranzfenster befindet (velocity_threshold).</p>
	homing_attained	<p>Im Homing Mode:</p> <p>Dieses Bit wird gesetzt, wenn die Referenzfahrt ohne Fehler beendet wurde.</p>
	ip_mode_active	<p>Im Interpolated Position Mode:</p> <p>Dieses Bit zeigt an, dass die Interpolation aktiv ist und die Interpolations-Datensätze ausgewertet werden. Es wird gesetzt, wenn dies durch das Bit enable_ip_mode im controlword angefordert wurde. Siehe hierzu unbedingt auch Abschnitt 8.4: <i>Interpolated Position Mode</i>, ab Seite 198.</p>

Bit 13		Abhängig von modes_of_operation :
	following_error	<p>Im Profile Position Mode:</p> <p>Dieses Bit wird gesetzt, wenn die aktuelle Ist-Position (position_actual_value) von der Soll-Position (position_demand_value) soweit abweicht, dass die Differenz außerhalb des parametrisierten Toleranzfensters liegt (following_error_window, following_error_time_out).</p>
	homing_error	<p>Im Homing Mode:</p> <p>Dieses Bit wird gesetzt, wenn die Referenzfahrt unterbrochen wird (Halt-Bit), beide Endschalter gleichzeitig ansprechen oder die bereits zurückgelegte Endschaltersuchfahrt größer als der vorgegebene Positionierraum ist (min_position_limit, max_position_limit).</p>
Bit 14	manufacturer_statusbit	<i>herstellerspezifisch</i>
		<p>Die Bedeutung dieses Bits ist konfigurierbar: Es kann gesetzt werden, wenn ein beliebiges Bit des manufacturer_statusword_1 gesetzt bzw. zurückgesetzt wird. Siehe hierzu auch Abschnitt 7.1.5.2: <i>Objekt 2000h: manufacturer_statuswords</i>, Seite 168.</p>
Bit 15	trigger_result	<i>herstellerspezifisch</i>
		<p>Die Bedeutung dieses Bits ist konfigurierbar: Es wird gesetzt, wenn ein Sample- Ereignis eingetreten ist und die Samplemaske entsprechend gesetzt ist. Siehe hierzu auch Abschnitt 6.15: <i>Sampling von Positionen</i>, ab Seite 138.</p>

7.1.5.2 Objekt 2000_h: manufacturer_statuswords

Um weitere Reglerzustände abbilden zu können, die nicht im – häufig zyklisch abgefragten – **statusword** vorhanden sein müssen, wurde die Objektgruppe **manufacturer_statuswords** eingeführt.

Index	2000 _h
Name	manufacturer_statuswords
Object Code	RECORD
No. of Elements	1

Ab Firmware 3.3.x.1.1

Sub-Index	01 _h
Description	manufacturer_statusword_1
Data Type	UINT32
Access	ro
PDO Mapping	yes
Units	--
Value Range	--
Default Value	--

Ab Firmware 3.3.x.1.1

Bit	Wertigkeit	Name
0	00000001 _h	is_referenced
1	00000002 _h	commutation_valid
2	00000004 _h	ready_for_enable
...		
31	80000000 _h	---

Ab Firmware 3.3.x.1.1

Ab Firmware 3.5.x.1.1

Ab Firmware 3.5.x.1.1

Tabelle 7.5: Bitbelegung im manufacturer_statusword_1

Bit 0	is_referenced	
		Das Bit wird gesetzt, wenn der Regler referenziert ist. Dies ist der Fall, wenn entweder eine Referenzfahrt erfolgreich durchgeführt wurde oder aufgrund des angeschlossenen Gebersystems (z.B. bei einem Absolutwertgeber) keine Referenzfahrt nötig ist.
Bit 1	commutation_valid	
		Das Bit wird gesetzt, wenn die Kommutierinformation gültig ist. Es ist insbesondere bei Gebersystemen ohne Kommutierinformation (z.B. Linearmotoren) hilfreich, weil dort die automatische Kommutierungsfindung einige Zeit in Anspruch nehmen kann. Wird dieses Bit überwacht, kann z.B. ein Timeout der Steuerung bei Freigabe des Reglers verhindert werden.
Bit 2	ready_for_enab	
		Das Bit wird gesetzt, wenn alle Bedingungen vorliegen, um den Regler freizugeben und nur noch die Reglerfreigabe selber fehlt. Folgende Bedingungen müssen vorliegen: <ul style="list-style-type: none"> - Der Antrieb ist fehlerfrei - Der Zwischenkreis ist geladen - Die Winkelgeberauswertung ist bereit. Es sind keine Prozesse (z.B. serielle Übertragung) aktiv, die eine Freigabe verhindern - Es ist kein blockierender Prozess aktiv (z.B. die automatische Motorparameter- Identifikation)



BEISPIEL

- A) Bit 14 des **statusword** soll gesetzt werden, wenn der Antrieb referenziert ist. *Antrieb referenziert* ist Bit 0 des **manufacturer_statusword_1**
- manufacturer_status_invert** = 0x00000000
manufacturer_status_mask = 0x00000001 (Bit 0)
- B) Bit 14 des **statusword** soll gesetzt werden, wenn der Antrieb keine gültige Kommutierlage hat.
Gültige Kommutierlage ist Bit 1 des **manufacturer_statusword_1**. Dieses Bit muss invertiert werden, damit es gesetzt wird, wenn die Kommutierinformation ungültig ist:
- manufacturer_status_invert** = 0x00000002 (Bit 1)
manufacturer_status_mask = 0x00000002 (Bit 1)
- C) Bit 14 des **statusword** soll gesetzt werden, wenn der Antrieb nicht bereit zur Freigabe ist ODER der Antrieb referenziert ist.
Gültige Kommutierlage ist Bit 2 des **manufacturer_statusword_1**. *Antrieb referenziert* ist Bit 0. Bit 2 muss invertiert werden, damit es gesetzt wird, wenn der Antrieb nicht bereit zur Freigabe ist:
- manufacturer_status_invert** = 0x00000004 (Bit 2)
manufacturer_status_mask = 0x00000005 (Bit 2, Bit 0)

7.1.5.3 Objekt 2005_h: manufacturer_status_masks

Mit dieser Objektgruppe wird festgelegt, welche gesetzten Bits der **manufacturer_statuswords** in das **statusword** eingeblendet werden. Siehe hierzu auch Abschnitt 7.1.5.2: *Objekt 2000_h: manufacturer_statuswords*, Seite 168.

Index	2005_h
Name	manufacturer_status_masks
Object Code	RECORD
No. of Elements	1

Ab Firmware 3.5.x.1.1

Sub-Index	01_h
Description	manufacturer_status_mask_1
Data Type	UINT32
Access	rw
PDO Mapping	yes
Units	--
Value Range	--
Default Value	0x00000000

Ab Firmware 3.5.x.1.1

7.1.5.4 Objekt 200A_h: manufacturer_status_invert

Mit dieser Objektgruppe wird festgelegt, welche Bits der **manufacturer_statuswords** invertiert in das **statusword** eingeblendet werden. Siehe hierzu auch Abschnitt 7.1.5.2: *Objekt 2000h: manufacturer_statuswords*, Seite 168.

Index	200A_h
Name	manufacturer_status_invert
Object Code	RECORD
No. of Elements	1

Ab Firmware 3.5.x.1.1

Sub-Index	01_h
Description	manufacturer_status_invert_1
Data Type	UINT32
Access	rw
PDO Mapping	yes
Units	--
Value Range	--
Default Value	0x00000000

Ab Firmware 3.5.x.1.1

7.1.6 Beschreibung der weiteren Objekte

7.1.6.1 In diesem Kapitel behandelte Objekte

Index	Objekt	Name	Typ	Attr.
605B _h	VAR	shutdown_option_code	INT16	rw
605C _h	VAR	disable_operation_option_code	INT16	rw
605A _h	VAR	quick_stop_option_code	INT16	rw
605E _h	VAR	fault_reaction_option_code	INT16	rw

7.1.6.2 Objekt 605B_h: shutdown_option_code

Mit dem Objekt **shutdown_option_code** wird vorgegeben, wie sich der Regler beim Zustandsübergang 8 (von **OPERATION ENABLE** nach **READY TO SWITCH ON**) verhält. Das Objekt zeigt das implementierte Verhalten des Reglers an. Es kann nicht verändert werden.

Index	605B_h
Name	shutdown_option_code
Object Code	VAR
Data Type	INT16

Access	rw
PDO Mapping	no
Units	--
Value Range	0
Default Value	0

Wert	Name
0	Endstufe wird ausgeschaltet, Motor ist frei drehbar

7.1.6.3 Objekt 605C_h: disable_operation_option_code

Mit dem Objekt **disable_operation_option_code** wird vorgegeben, wie sich der Regler beim Zustandsübergang 5 (von **OPERATION ENABLE** nach **SWITCHED ON**) verhält. Das Objekt zeigt das implementierte Verhalten des Reglers an. Es kann nicht verändert werden.

Index	605C_h
Name	disable_operation_option_code
Object Code	VAR
Data Type	INT16

Access	rw
PDO Mapping	no
Units	--
Value Range	-1
Default Value	-1

Wert	Name
-1	Bremsen mit quickstop_deceleration

7.1.6.4 Objekt 605A_h: quick_stop_option_code

Mit dem Parameter **quick_stop_option_code** wird vorgegeben, wie sich der Regler bei einem **Quick Stop** verhält. Das Objekt zeigt das implementierte Verhalten des Reglers an. Es kann nicht verändert werden.

Index	605A_h
Name	quick_stop_option_code
Object Code	VAR
Data Type	INT16

Access	rw
PDO Mapping	no
Units	--
Value Range	2
Default Value	2

Wert	Name
2	Bremsen mit quickstop_deceleration

7.1.6.5 Objekt 605E_h: fault_reaction_option_code

Mit dem Objekt **fault_reaction_option_code** wird vorgegeben, wie sich der Regler bei einem Fehler (**fault**) verhält. Da bei der SERVOTEC S2 2000-Reihe die Fehlerreaktion vom jeweiligen Fehler abhängt, kann dieses Objekt nicht parametrierbar werden und gibt immer 0 zurück. Um die Fehlerreaktion der einzelnen Fehler zu verändern siehe Abschnitt 6.18: *Fehlermanagement, ab Seite 151*.

Index	605E_h
Name	fault_reaction_option_code
Object Code	VAR
Data Type	INT16

Access	rw
PDO Mapping	no
Units	--
Value Range	0
Default Value	0

8 Betriebsarten

8.1 Einstellen der Betriebsart

8.1.1 Übersicht

Der Antriebsregler kann in eine Vielzahl von Betriebsarten versetzt werden. Nur einige sind unter CANopen detailliert spezifiziert:

- momentengeregelter Betrieb profile torque mode
- drehzahl geregelter Betrieb profile velocity mode
- Referenzfahrt homing mode
- Positionierbetrieb profile position mode
- Synchrone Positionsvorgabe interpolated position mode

8.1.2 Beschreibung der Objekte

8.1.2.1 In diesem Kapitel behandelte Objekte

Index	Objekt	Name	Typ	Attr.
6060 _h	VAR	modes_of_operation	INT8	wo
6061 _h	VAR	modes_of_operation_display	INT8	ro

8.1.2.2 Objekt 6060_h: modes_of_operation

Mit dem Objekt **modes_of_operation** wird die Betriebsart des Reglers eingestellt.

Index	6060 _h
Name	modes_of_operation
Object Code	VAR
Data Type	INT8

Access	rw
PDO Mapping	yes
Units	--
Value Range	1, 3, 4, 6, 7
Default Value	--

Wert	Aktion
1	Profile Position Mode (Lageregler mit Positionierbetrieb)
3	Profile Velocity Mode (Drehzahlregler mit Sollwertrampe)
4	Torque Profile Mode (Momentenregler mit Sollwertrampe)
6	Homing Mode (Referenzfahrt)
7	Interpolated Position Mode



Die aktuelle Betriebsart kann nur im Objekt **modes_of_operation_display** gelesen werden !
 Da ein Wechsel der Betriebsart etwas Zeit in Anspruch nehmen kann, **muss** solange gewartet werden, bis der neu ausgewählte Modus im Objekt **modes_of_operation_display** erscheint.

8.1.2.3 Objekt 6061_h: modes_of_operation_display

Im Objekt **modes_of_operation_display** kann die aktuelle Betriebsart des Reglers gelesen werden. Wird eine Betriebsart über das Objekt **6060_h** eingestellt, werden neben der eigentlichen Betriebsart auch die Sollwert- Aufschaltungen (Sollwert- Selektor) vorgenommen, die für einen Betrieb des Reglers unter CANopen nötig sind.

Dies sind:

	Profile Velocity Mode	Profile Torque Mode
Selektor A	Drehzahl- Sollwert (Feldbus 1)	Drehmoment- Sollwert (Feldbus 1)
Selektor B	Ggf. Momentenbegrenzung	inaktiv
Selektor C	Drehzahl- Sollwert (Synchrondrehz.)	inaktiv

Außerdem wird die Sollwert- Rampe grundsätzlich eingeschaltet. Nur wenn diese Aufschaltungen in der genannten Weise eingestellt sind, wird auch eine der CANopen- Betriebsarten zurückgegeben. Werden diese Einstellungen z.B. mit dem ServoCommander™ geändert, wird eine jeweilige „User“-Betriebsart zurückgegeben, um anzuzeigen, dass die Selektoren verändert wurden.

Index	6061_h
Name	modes_of_operation_display
Object Code	VAR
Data Type	INT8

Access	ro
PDO Mapping	yes
Units	--
Value Range	-1, 1, 3, 4, 6, 7
Default Value	3

Wert	Aktion
-1	Unbekannte Betriebsart / Betriebsartenwechsel
-11	User Position Mode
-13	User Velocity Mode
-14	User Torque Mode
1	Profile Position Mode (Lageregler mit Positionierbetrieb)
3	Profile Velocity Mode (Drehzahlregler mit Sollwertrampe)
4	Torque Profile Mode (Momentenregler mit Sollwertrampe)
6	Homing Mode (Referenzfahrt)
7	Interpolated Position Mode



Die Betriebsart kann nur über das Objekt **modes_of_operation** gesetzt werden. Da ein Wechsel der Betriebsart etwas Zeit in Anspruch nehmen kann, **muss** solange gewartet werden, bis der neu ausgewählte Modus im Objekt **modes_of_operation_display** erscheint. Während dieses Zeitraumes kann kurzzeitig „ungültige Betriebsart“ (-1) angezeigt werden.

8.2 Betriebsart Referenzfahrt (Homing Mode)

8.2.1 Übersicht

In diesem Kapitel wird beschrieben, wie der Antriebsregler die Anfangsposition sucht (auch Bezugspunkt, Referenzpunkt oder Nullpunkt genannt). Es gibt verschiedene Methoden diese Position zu bestimmen, wobei entweder die Endschalter am Ende des Positionierbereiches benutzt werden können oder aber ein Referenzschalter (Nullpunkt-Schalter) innerhalb des möglichen Verfahrweges. Um eine möglichst große Reproduzierbarkeit zu erreichen, kann bei einigen Methoden der Nullimpuls des verwendeten Winkelgebers (Resolver, Inkrementalgeber etc.) mit einbezogen werden.

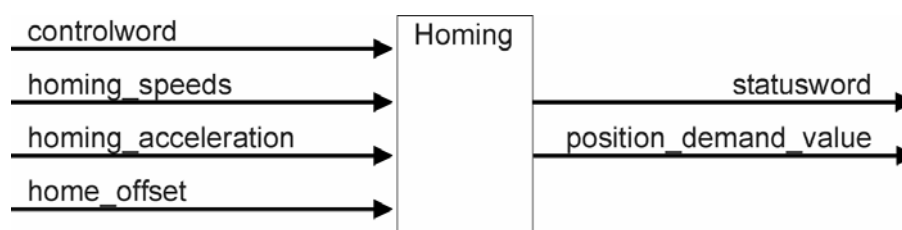


Abbildung 8.1: Die Referenzfahrt

Der Benutzer kann die Geschwindigkeit, Beschleunigung und die Art der Referenzfahrt bestimmen. Mit dem Objekt `home_offset` kann die Nullposition des Antriebs an eine beliebige Stelle verschoben werden.

Es gibt zwei Referenzfahrgeschwindigkeiten. Die höhere Suchgeschwindigkeit (`speed_during_search_for_switch`) wird benutzt, um den Endschalter bzw. den Referenzschalter zu finden. Um dann die Position der betreffenden Schaltflanke exakt bestimmen zu können, wird auf die Kriechgeschwindigkeit (`speed_during_search_for_zero`) umgeschaltet.

Soll der Antrieb nicht neu referenziert werden, sondern lediglich die Position auf einen vorgegebenen Wert gesetzt werden, kann das Objekt **2030_h (set_position_absolute)** benutzt werden. Siehe hierzu Abschnitt 6.7.2.15: *Objekt 2030_h: set_position_absolute*, Seite 110.



Die Fahrt auf die Nullposition ist unter CANopen in der Regel nicht Bestandteil der Referenzfahrt. Sind dem Regler alle erforderlichen Größen bekannt (z.B. weil er die Lage des Nullimpulses bereits kennt), wird keine physikalische Bewegung ausgeführt. Dieses Verhalten kann durch das Objekt **6510_h_F0_h (compatibility_control)**, siehe Abschnitt 6.2: *Kompatibilitäts-Einstellungen*, ab Seite 65) geändert werden, so dass immer eine Fahrt auf Null ausgeführt wird.

8.2.2 Beschreibung der Objekte

8.2.2.1 In diesem Kapitel behandelte Objekte

Index	Objekt	Name	Typ	Attribute
607C _h	VAR	home_offset	INT32	rw
6098 _h	VAR	homing_method	INT8	rw
6099 _h	ARRAY	homing_speeds	UINT32	rw
609A _h	VAR	homing_acceleration	UINT32	rw
2045 _h	VAR	homing_timeout	UINT16	rw

8.2.2.2 Betroffene Objekte aus anderen Abschnitten

Index	Objekt	Name	Typ	Abschnitt
6040 _h	VAR	controlword	UINT16	7 Gerätesteuerung
6041 _h	VAR	statusword	UINT16	7 Gerätesteuerung

8.2.2.3 Objekt 607C_h: home_offset

Das Objekt **home_offset** legt die Verschiebung der Nullposition gegenüber der ermittelten Referenzposition fest.

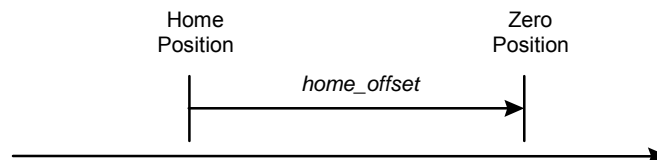


Abbildung 8.2: Home Offset

Index	607C_h
Name	home_offset
Object Code	VAR
Data Type	INT32

Access	rw
PDO Mapping	yes
Units	position units
Value Range	--
Default Value	0

8.2.2.4 Objekt 6098_h: homing_method

Für eine Referenzfahrt werden eine Reihe unterschiedlicher Methoden bereitgestellt. Über das Objekt **homing_method** kann die für die Applikation benötigte Variante ausgewählt werden. Es gibt vier mögliche Referenzfahrt-Signale: den negativen und positiven Endschalter, den Referenzschalter und den (periodischen) Nullimpuls des Winkelgebers. Außerdem kann der Regler sich ganz ohne zusätzliches Signal auf den negativen oder positiven Anschlag referenzieren. Wenn über das Objekt **homing_method** eine Methode zum Referenzieren bestimmt wird, so werden hiermit folgende Einstellungen gemacht:

- Die Referenzquelle (neg./pos. Endschalter, der Referenzschalter, neg. / pos. Anschlag)
- Die Richtung und der Ablauf der Referenzfahrt
- Die Art der Auswertung des Nullimpulses vom verwendeten Winkelgeber

Index	6098_h
Name	homing_method
Object Code	VAR
Data Type	INT8

Access	rw
PDO Mapping	yes
Units	--
Value Range	-18, -17, -2, -1, 1, 2, 7, 11, 17, 18, 23, 27, 32, 33, 34, 35
Default Value	17

Wert	Richtung	Ziel	Bezugspunkt für Null	DS402
-18	positiv	Anschlag	Anschlag	-18
-17	negativ	Anschlag	Anschlag	-17
-2	positiv	Anschlag	Nullimpuls	-2
-1	negativ	Anschlag	Nullimpuls	-1
1	negativ	Endschalter	Nullimpuls	1
2	positiv	Endschalter	Nullimpuls	2
7	positiv	Referenzschalter	Nullimpuls	7
11	negativ	Referenzschalter	Nullimpuls	11
17	negativ	Endschalter	Endschalter	17
18	positiv	Endschalter	Endschalter	18
23	positiv	Referenzschalter	Referenzschalter	23
27	negativ	Referenzschalter	Referenzschalter	27
32	negativ	Nullimpuls	Nullimpuls	33
33	positiv	Nullimpuls	Nullimpuls	34
34		Keine Fahrt	Aktuelle Ist-Position	35



In bisherigen CANopen- Implementierungen sind die Referenzfahrt-Methoden 32, 33, 34 und 35 nicht gemäß DS402 zugeordnet. Daher besteht die Möglichkeit über das Objekt **compatibility_control** (siehe Abschnitt 6.2: *Kompatibilitäts-Einstellungen, ab Seite 65*) die Zuordnung gemäß DS402 auszuwählen. In diesem Fall sind die kursiv gedruckten Methoden- Nummern zu verwenden. **Für Kompatibilität zu früheren Versionen brauchen keine Änderungen durchgeführt werden und es können die bisherigen Nummern verwendet werden !**

Die **homing_method** kann nur verstellt werden, wenn die Referenzfahrt nicht aktiv ist. Ansonsten wird eine Fehlermeldung zurückgegeben.

Der Ablauf der einzelnen Methoden ist in Abschnitt 8.2.3: *Referenzfahrt-Abläufe, ab Seite 184*.

8.2.2.5 Objekt 6099_h: homing_speeds

Dieses Objekt bestimmt die Geschwindigkeiten, die während der Referenzfahrt benutzt werden.

Index	6099_h
Name	homing_speeds
Object Code	ARRAY
No. of Elements	2
Data Type	UINT32

Sub-Index	01_h
Description	speed_during_search_for_switch
Access	rw
PDO Mapping	yes
Units	speed units
Value Range	--
Default Value	100 min ⁻¹

Sub-Index	02_h
Description	speed_during_search_for_zero
Access	rw
PDO Mapping	yes
Units	speed units
Value Range	--
Default Value	10 min ⁻¹



Wird Bit 6 im Objekt **compatibility_control** (siehe Abschnitt 6.2: *Kompatibilitäts-Einstellungen, ab Seite 65*) gesetzt, wird nach der Referenzfahrt eine Fahrt auf Null durchgeführt. Ist dieses Bit gesetzt und das Objekt **speed_during_search_for_switch** wird beschrieben, wird sowohl die Geschwindigkeit für die Schaltersuche, als auch die Geschwindigkeit für die Fahrt auf Null beschrieben.

8.2.2.6 Objekt 609A_h: homing_acceleration

Das Objekt **homing_acceleration** legt die Beschleunigung fest, die während der Referenzfahrt für alle Beschleunigungs- und Bremsvorgänge verwendet wird.

Index	609A_h
Name	homing_acceleration
Object Code	VAR
Data Type	UINT32

Access	rw
PDO Mapping	yes
Units	acceleration units
Value Range	--
Default Value	1000 min ⁻¹ / s

8.2.2.7 Objekt 2045_h: homing_timeout

Die Referenzfahrt kann auf ihre maximale Ausführungszeit überwacht werden. Dazu kann mit dem Objekt **homing_timeout** die maximale Ausführungszeit angegeben werden. Wird diese Zeit überschritten, ohne dass die Referenzfahrt beendet wurde, wird der Fehler 11-3 ausgelöst.

Index	2045_h
Name	homing_timeout
Object Code	VAR
Data Type	UINT16

Ab Firmware 3.2.0.1.1

Access	rw
PDO Mapping	no
Units	ms
Value Range	0 (aus), 1 ... 65535
Default Value	60000

8.2.3 Referenzfahrt-Abläufe

Die verschiedenen Referenzfahrt-Methoden sind in den folgenden Abbildungen dargestellt. Die eingekreisten Nummern entsprechen dem im Objekt **homing_method** einzutragenden Code.

8.2.3.1 Methode 1: Negativer Endschalter mit Nullimpulsauswertung

Bei dieser Methode bewegt sich der Antrieb zunächst relativ schnell in negativer Richtung, bis er den negativen Endschalter erreicht. Dieses wird im Diagramm durch die steigende Flanke dargestellt. Danach fährt der Antrieb langsam zurück und sucht die genaue Position des Endschalters. Die Nullposition bezieht sich auf den ersten Nullimpuls des Winkelgebers in positiver Richtung vom Endschalter.

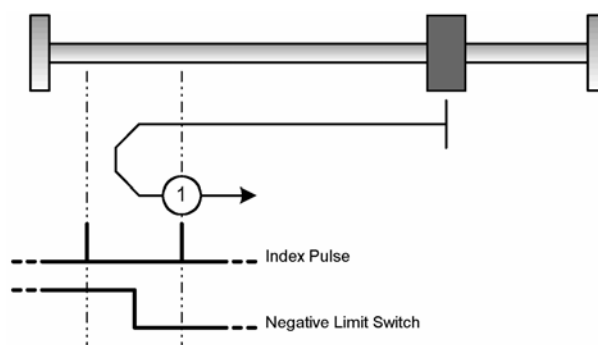


Abbildung 8.3: Referenzfahrt auf den negativen Endschalter mit Auswertung des Nullimpulses

8.2.3.2 Methode 2: Positiver Endschalter mit Nullimpulsauswertung

Bei dieser Methode bewegt sich der Antrieb zunächst relativ schnell in positiver Richtung, bis er den positiven Endschalter erreicht. Dieses wird im Diagramm durch die steigende Flanke dargestellt. Danach fährt der Antrieb langsam zurück und sucht die genaue Position des Endschalters. Die Nullposition bezieht sich auf den ersten Nullimpuls des Winkelgebers in negativer Richtung vom Endschalter.

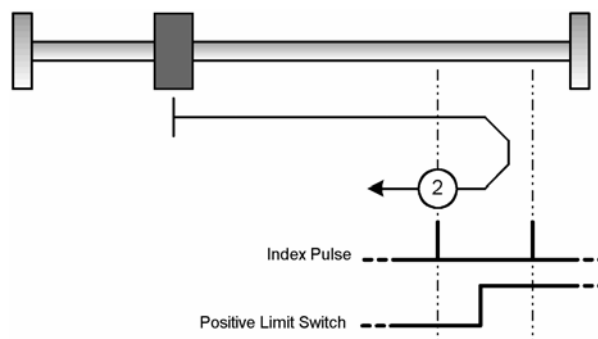


Abbildung 8.4: Referenzfahrt auf den positiven Endschalter mit Auswertung des Nullimpulses

8.2.3.3 Methoden 7 u. 11: Referenzschalter und Nullimpulsauswertung

Diese beiden Methoden nutzen den Referenzschalter, der nur über einen Teil der Strecke aktiv ist. Diese Referenzmethoden bieten sich besonders für Rundachsen-Applikationen an, wo der Referenzschalter einmal pro Umdrehung aktiviert wird.

Bei der Methode 7 bewegt sich der Antrieb zunächst in positiver und bei Methode 11 in negativer Richtung. Abhängig von der Fahrtrichtung bezieht sich die Nullposition auf den ersten Nullimpuls in negativer oder positiver Richtung vom Referenzschalter. Dieses ist in den beiden folgenden Abbildungen ersichtlich.

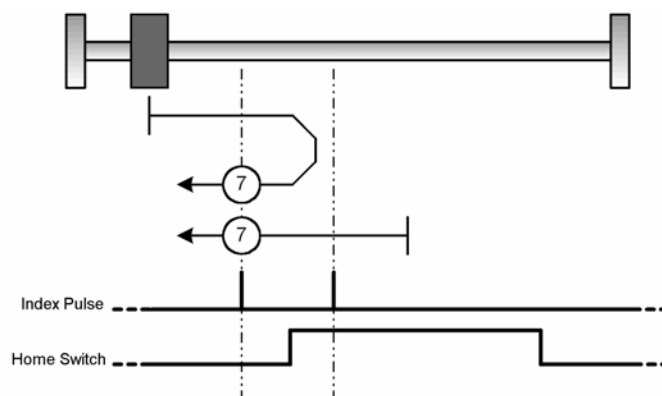


Abbildung 8.5: Referenzfahrt auf den Referenzschalter mit Auswertung des Nullimpulses bei positiver Anfangsbewegung



Bei Referenzfahrten auf den Referenzschalter dienen die Endschalter zunächst zur Suchrichtungsumkehr. Wird im Anschluss der gegenüberliegende Endschalter erreicht, wird ein Fehler ausgelöst.

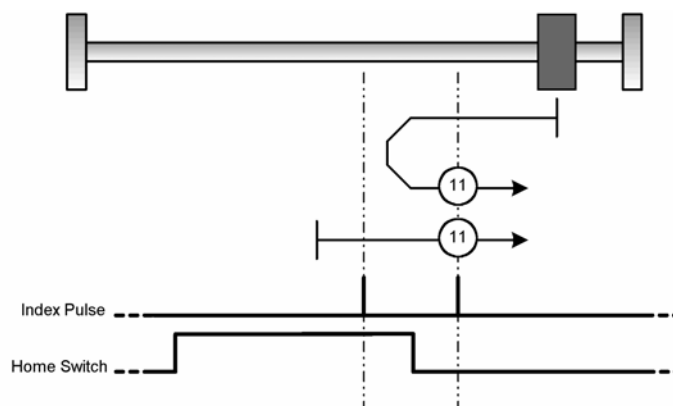


Abbildung 8.6: Referenzfahrt auf den Referenzschalter mit Auswertung des Nullimpulses bei negativer Anfangsbewegung

8.2.3.4 Methode 17: Referenzfahrt auf den negativen Endschalter

Bei dieser Methode bewegt sich der Antrieb zunächst relativ schnell in negativer Richtung, bis er den negativen Endschalter erreicht. Dieses wird im Diagramm durch die steigende Flanke dargestellt. Danach fährt der Antrieb langsam zurück und sucht die genaue Position des Endschalters. Die Nullposition bezieht sich auf die fallende Flanke vom negativen Endschalter.

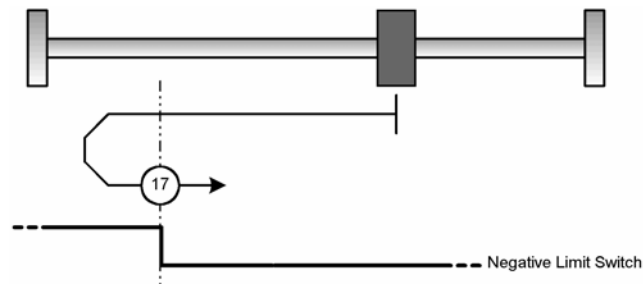


Abbildung 8.7: Referenzfahrt auf den negativen Endschalter

8.2.3.5 Methode 18: Referenzfahrt auf den positiven Endschalter

Bei dieser Methode bewegt sich der Antrieb zunächst relativ schnell in positiver Richtung, bis er den positiven Endschalter erreicht. Dieses wird im Diagramm durch die steigende Flanke dargestellt. Danach fährt der Antrieb langsam zurück und sucht die genaue Position des Endschalters. Die Nullposition bezieht sich auf die fallende Flanke vom positiven Endschalter.

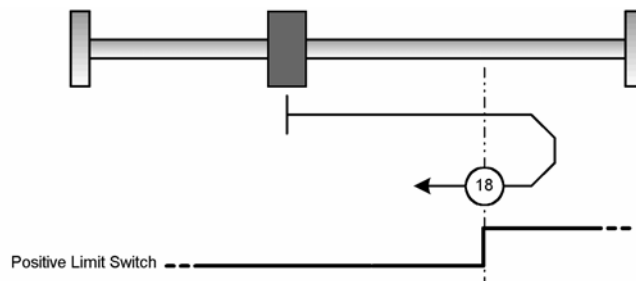


Abbildung 8.8: Referenzfahrt auf den positiven Endschalter

8.2.3.6 Methoden 23 und 27: Referenzfahrt auf den Referenzschalter

Diese beiden Methoden nutzen den Referenzschalter, der nur über einen Teil der Strecke aktiv ist. Diese Referenzmethode bietet sich besonders für Rundachsen-Applikationen an, wo der Referenzschalter einmal pro Umdrehung aktiviert wird.

Bei der Methode 23 bewegt sich der Antrieb zunächst in positiver und bei Methode 27 in negativer Richtung. Die Nullposition bezieht sich auf die Flanke vom Referenzschalter. Dieses ist in den beiden folgenden Abbildungen ersichtlich.

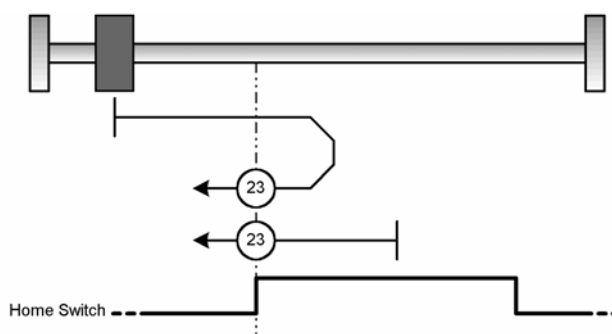


Abbildung 8.9: Referenzfahrt auf den Referenzschalter bei positiver Anfangsbewegung



Bei Referenzfahrten auf den Referenzschalter dienen die Endschalter zunächst zur Suchrichtungsumkehr. Wird im Anschluss der gegenüberliegende Endschalter erreicht, wird ein Fehler ausgelöst.

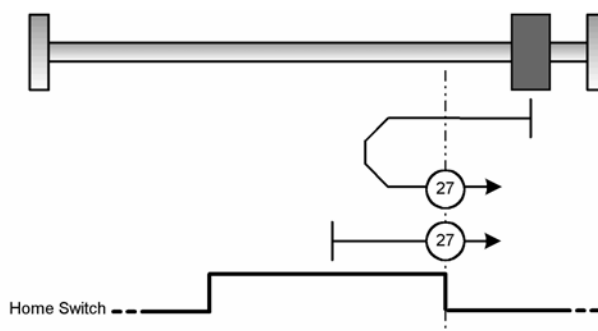


Abbildung 8.10: Referenzfahrt auf den Referenzschalter bei negativer Anfangsbewegung

8.2.3.7 Methode –1: negativer Anschlag mit Nullimpulsauswertung

Bei dieser Methode bewegt sich der Antrieb in negativer Richtung, bis er den Anschlag erreicht. Hierbei steigt das I^2t -Integral des Motors auf maximal 90%. Der Anschlag muss mechanisch so dimensioniert sein, dass er bei dem parametrierten Maximalstrom keinen Schaden nimmt. Die Nullposition bezieht sich auf den ersten Nullimpuls des Winkelgebers in positiver Richtung vom Anschlag.

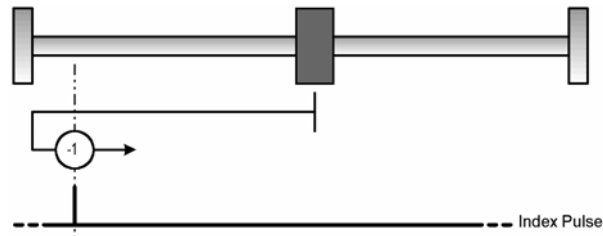


Abbildung 8.11: Referenzfahrt auf den negativen Anschlag mit Auswertung des Nullimpulses

8.2.3.8 Methode –2: positiver Anschlag mit Nullimpulsauswertung

Bei dieser Methode bewegt sich der Antrieb in positiver Richtung, bis er den Anschlag erreicht. Hierbei steigt das I^2t -Integral des Motors auf maximal 90%. Der Anschlag muss mechanisch so dimensioniert sein, dass er bei dem parametrierten Maximalstrom keinen Schaden nimmt. Die Nullposition bezieht sich auf den ersten Nullimpuls des Winkelgebers in negativer Richtung vom Anschlag.

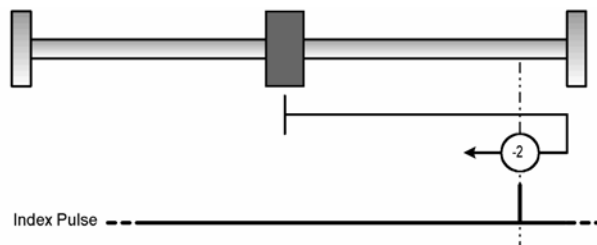


Abbildung 8.12: Referenzfahrt auf den positiven Anschlag mit Auswertung des Nullimpulses

8.2.3.9 Methode –17: Referenzfahrt auf den negativen Anschlag

Bei dieser Methode bewegt sich der Antrieb in negativer Richtung, bis er den Anschlag erreicht. Hierbei steigt das I^2t -Integral des Motors auf maximal 90%. Der Anschlag muss mechanisch so dimensioniert sein, dass er bei dem parametrierten Maximalstrom keinen Schaden nimmt. Die Nullposition bezieht sich direkt auf den Anschlag.

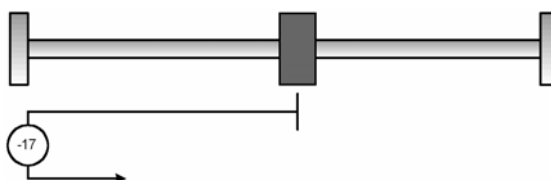


Abbildung 8.13: Referenzfahrt auf den negativen Anschlag

8.2.3.10 Methode –18: Referenzfahrt auf den positiven Anschlag

Bei dieser Methode bewegt sich der Antrieb in positiver Richtung, bis er den Anschlag erreicht. Hierbei steigt das I^2t -Integral des Motors auf maximal 90%. Der Anschlag muss mechanisch so dimensioniert sein, dass er bei dem parametrierten Maximalstrom keinen Schaden nimmt. Die Nullposition bezieht sich direkt auf den Anschlag.

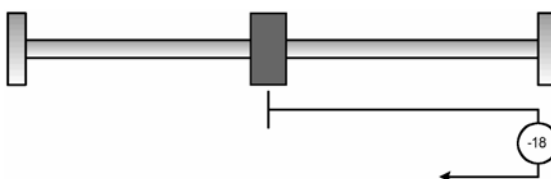


Abbildung 8.14: Referenzfahrt auf den positiven Anschlag

8.2.3.11 Methoden 32 und 33: Referenzfahrt auf den Nullimpuls

Bei den Methoden 32 (33 gemäß DS402) und 33 (34 gemäß DS402) ist die Richtung der Referenzfahrt negativ bzw. positiv. Die Nullposition bezieht sich auf den ersten Nullimpuls vom Winkelgeber in Suchrichtung.

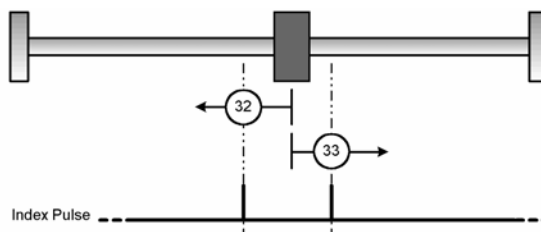


Abbildung 8.15: Referenzfahrt nur auf den Nullimpuls bezogen

8.2.3.12 Methode 34: Referenzfahrt auf die aktuelle Position

Bei der Methode 34 (35 gemäß DS402) wird die Nullposition auf die aktuelle Position bezogen.

Soll der Antrieb nicht neu referenziert werden, sondern lediglich die Position auf einen vorgegebenen Wert gesetzt werden, kann das Objekt 2030_h (**set_position_absolute**) benutzt werden. Siehe hierzu Abschnitt 6.7.2.15: *Objekt 2030h: set_position_absolute, Seite 110.*

8.2.4 Steuerung der Referenzfahrt

Die Referenzfahrt wird durch das **controlword** / **statusword** gesteuert und überwacht. Das Starten erfolgt durch Setzen des Bit 4 im **controlword**. Der erfolgreiche Abschluss der Fahrt wird durch ein gesetztes Bit 12 im Objekt **statusword** angezeigt. Ein gesetztes Bit 13 im Objekt **statusword** zeigt an, dass während der Referenzfahrt ein Fehler aufgetreten ist. Die Fehlerursache kann über die Objekte **error_register** und **pre_defined_error_field** bestimmt werden.

Bit 4	Bedeutung
0	Referenzfahrt ist nicht aktiv
0 → 1	Referenzfahrt starten
1	Referenzfahrt ist aktiv
1 → 0	Referenzfahrt unterbrechen

Tabelle 8.1: Beschreibung der Bits im controlword

Bit 13	Bit 12	Bedeutung
0	0	Referenzfahrt ist noch nicht fertig
0	1	Referenzfahrt erfolgreich durchgeführt
1	0	Referenzfahrt nicht erfolgreich durchgeführt
1	1	verbotener Zustand

Tabelle 8.2: Beschreibung der Bits im statusword

8.3 Betriebsart Positionieren (Profile Position Mode)

8.3.1 Übersicht

Die Struktur dieser Betriebsart wird in *Abbildung 8.16* ersichtlich:

Die Zielposition (**target_position**) wird dem Fahrkurven-Generator übergeben. Dieser erzeugt einen Lage-Sollwert (**position_demand_value**) für den Lageregler, der in dem Kapitel **Lageregler** beschrieben wird (Position Control Function, Abschnitt 6.6.2.2: *Objekt 2073h: velocity_display_filter_time, Seite 97*). Diese zwei Funktionsblöcke können unabhängig voneinander eingestellt werden.

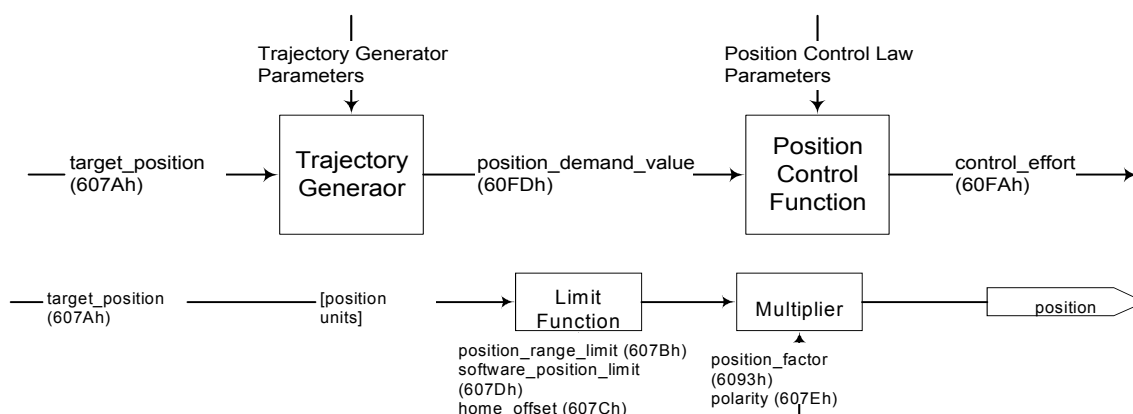


Abbildung 8.16: Fahrkurven-Generator und Lageregler

Alle Eingangsgrößen des Fahrkurven-Generators werden mit den Größen der Factor-Group (siehe Abschnitt 6.2: *Kompatibilitäts-Einstellungen, ab Seite 65*) in die internen Einheiten des Reglers umgerechnet. Die internen Größen werden hier mit einem Sternchen gekennzeichnet und werden vom Anwender in der Regel nicht benötigt.

8.3.2 Beschreibung der Objekte

8.3.2.1 In diesem Kapitel behandelte Objekte

Index	Objekt	Name	Typ	Attr.
607A _h	VAR	target_position	INT32	rw
6081 _h	VAR	profile_velocity	UINT32	rw
6082 _h	VAR	end_velocity	UINT32	rw
6083 _h	VAR	profile_acceleration	UINT32	rw
6084 _h	VAR	profile_deceleration	UINT32	rw
6085 _h	VAR	quick_stop_deceleration	UINT32	rw
6086 _h	VAR	motion_profile_type	INT16	rw

8.3.2.2 Betroffene Objekte aus anderen Abschnitten

Index	Objekt	Name	Typ	Abschnitt
6040 _h	VAR	controlword	INT16	7 Gerätesteuerung
6041 _h	VAR	statusword	UINT16	7 Gerätesteuerung
605A _h	VAR	quick_stop_option_code	INT16	7 Gerätesteuerung
607E _h	VAR	polarity	UINT8	6.3 Umrechnungsfaktoren
6093 _h	ARRAY	position_factor	UINT32	6.3 Umrechnungsfaktoren
6094 _h	ARRAY	velocity_encoder_factor	UINT32	6.3 Umrechnungsfaktoren
6097 _h	ARRAY	acceleration_factor	UINT32	6.3 Umrechnungsfaktoren

8.3.2.3 Objekt 607A_h: target_position

Das Objekt **target_position** (Zielposition) bestimmt, an welche Position der Antriebsregler fahren soll. Dabei muss die aktuelle Einstellung der Geschwindigkeit, der Beschleunigung, der Bremsverzögerung und die Art des Fahrprofils (**motion_profile_type**) etc. berücksichtigt werden. Die Zielposition (**target_position**) wird entweder als absolute oder relative Angabe interpretiert (**controlword**, Bit 6).

Index	607A_h
Name	target_position
Object Code	VAR
Data Type	INT32

Access	rw
PDO Mapping	yes
Units	position units
Value Range	--
Default Value	0

8.3.2.4 Objekt 6081_h: profile_velocity

Das Objekt **profile_velocity** gibt die Geschwindigkeit an, die normalerweise während einer Positionierung am Ende der Beschleunigungsrampe erreicht wird. Das Objekt **profile_velocity** wird in speed units angegeben.

Index	6081 _h
Name	profile_velocity
Object Code	VAR
Data Type	UINT32

Access	rw
PDO Mapping	yes
Units	speed_units
Value Range	--
Default Value	1000

8.3.2.5 Objekt 6082_h: end_velocity

Das Objekt **end_velocity** (Endgeschwindigkeit) definiert die Geschwindigkeit, die der Antrieb haben muss, wenn er die Zielposition (**target_position**) erreicht. Normalerweise ist dieses Objekt auf Null zu setzen, damit der Regler beim Erreichen der Zielposition (**target_position**) stoppt. Für lückenlose Positionierungen kann eine von Null abweichende Geschwindigkeit vorgegeben werden. Das Objekt **end_velocity** wird in denselben Einheiten wie das Objekt **profile_velocity** angegeben.

Index	6082 _h
Name	end_velocity
Object Code	VAR
Data Type	UINT32

Access	rw
PDO Mapping	yes
Units	speed units
Value Range	--
Default Value	0

8.3.2.6 Objekt 6083_h: profile_acceleration

Das Objekt **profile_acceleration** gibt die Beschleunigung an, mit der auf den Sollwert beschleunigt. Es wird in benutzerdefinierten Beschleunigungseinheiten (acceleration units) angegeben (siehe Abschnitt 6.3: *Umrechnungsfaktoren (Factor Group)*, ab Seite 68).

Index	6083_h
Name	profile_acceleration
Object Code	VAR
Data Type	UINT32

Access	rw
PDO Mapping	yes
Units	acceleration units
Value Range	--
Default Value	10000 min ⁻¹ /s

8.3.2.7 Objekt 6084_h: profile_deceleration

Das Objekt **profile_deceleration** gibt die Beschleunigung an, mit der gebremst wird. Es wird in benutzerdefinierten Beschleunigungseinheiten (acceleration units) angegeben (siehe Abschnitt 6.3: *Umrechnungsfaktoren (Factor Group)*, ab Seite 68).

Index	6084_h
Name	profile_deceleration
Object Code	VAR
Data Type	UINT32

Access	rw
PDO Mapping	yes
Units	acceleration units
Value Range	--
Default Value	10000 min ⁻¹ /s

8.3.2.8 Objekt 6085_h: quick_stop_deceleration

Das Objekt **quick_stop_deceleration** gibt an, mit welcher Bremsverzögerung der Motor stoppt, wenn ein **Quick Stop** ausgeführt wird (siehe Abschnitt 7.1.2.2: Zustandsdiagramm: Zustandsübergänge, ab Seite 157). Das Objekt **quick_stop_deceleration** wird in derselben Einheit wie das Objekt **profile_deceleration** angegeben.

Index	6085_h
Name	quick_stop_deceleration
Object Code	VAR
Data Type	UINT32

Access	rw
PDO Mapping	yes
Units	acceleration units
Value Range	--
Default Value	14100 min ⁻¹ /s

8.3.2.9 Objekt 6086_h: motion_profile_type

Das Objekt **motion_profile_type** wird verwendet, um die Art des Positionierprofils auszuwählen.

Index	6086_h
Name	motion_profile_type
Object Code	VAR
Data Type	INT16

Access	rw
PDO Mapping	yes
Units	--
Value Range	0, 2
Default Value	0

Wert	Kurvenform
0	Lineare Rampe
2	Ruckfreie Rampe Ab Firmware 3.1.0.1.1

8.3.3 Funktionsbeschreibung

Es gibt zwei Möglichkeiten eine Zielposition an den Regler zu übergeben:

Einfacher Fahrauftrag

Wenn der Regler eine Zielposition erreicht hat, signalisiert er dies dem Host mit dem Bit **target_reached** (Bit 10 im Objekt **statusword**). In dieser Betriebsart stoppt der Regler, wenn er das Ziel erreicht hat.

Folge von Fahraufträgen

Nachdem der Regler ein Ziel erreicht hat, beginnt er sofort das nächste Ziel anzufahren. Dieser Übergang kann fließend erfolgen, ohne dass der Regler zwischendurch zum Stillstand kommt.

Diese beiden Methoden werden durch die Bits **new_set_point** und **change_set_immediatly** in dem Objekt **controlword** und **set_point_acknowledge** in dem Objekt **statusword** kontrolliert. Diese Bits stehen in einem Frage-Antwort-Verhältnis zueinander. Hierdurch wird es möglich, einen Fahrauftrag vorzubereiten, während ein anderer noch läuft.

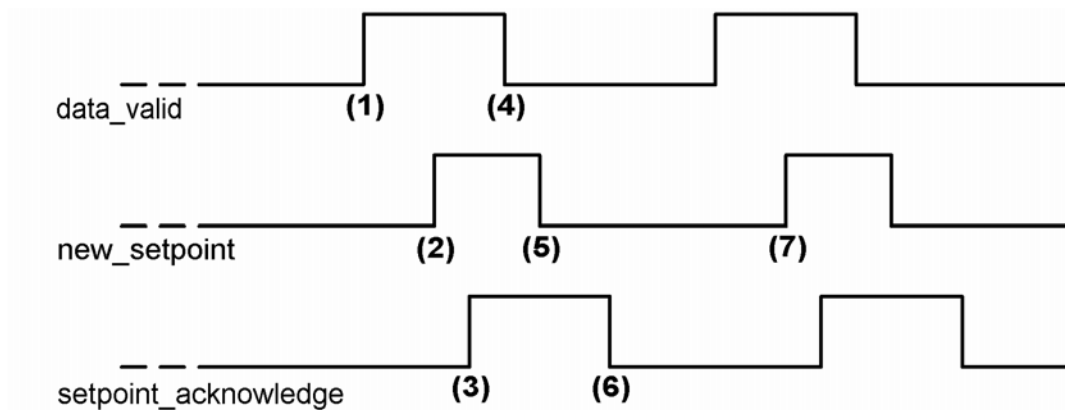


Abbildung 8.17: Fahrauftrag-Übertragung von einem Host

In *Abbildung 8.17* können Sie sehen, wie der Host und der Regler über den CAN-Bus miteinander kommunizieren:

Zuerst werden die Positionierdaten (Zielposition, Fahrgeschwindigkeit, Endgeschwindigkeit und die Beschleunigung) an den Regler übertragen. Wenn der Positionierdatensatz vollständig eingeschrieben ist (1), kann der Host die Positionierung starten, indem er das Bit **new_set_point** im **controlword** auf „1“ setzt (2). Nachdem der Regler die neuen Daten erkannt und in seinen Puffer übernommen hat, meldet er dies dem Host durch das Setzen des Bits **set_point_acknowledge** im **statusword** (3).

Daraufhin kann der Host beginnen, einen neuen Positionierdatensatz in den Regler einzuschreiben (4) und das Bit **new_set_point** wieder zu löschen (5). Erst wenn der Regler einen neuen Fahrauftrag akzeptieren kann (6), signalisiert er dies durch eine „0“ im **set_point_acknowledge**-Bit. Vorher darf vom Host keine neue Positionierung gestartet werden (7).

In Abbildung 8.18 wird eine neue Positionierung erst gestartet, nachdem die vorherige vollständig abgeschlossen wurde. Der Host wertet hierzu das Bit **target_reached** im Objekt **statusword** aus.

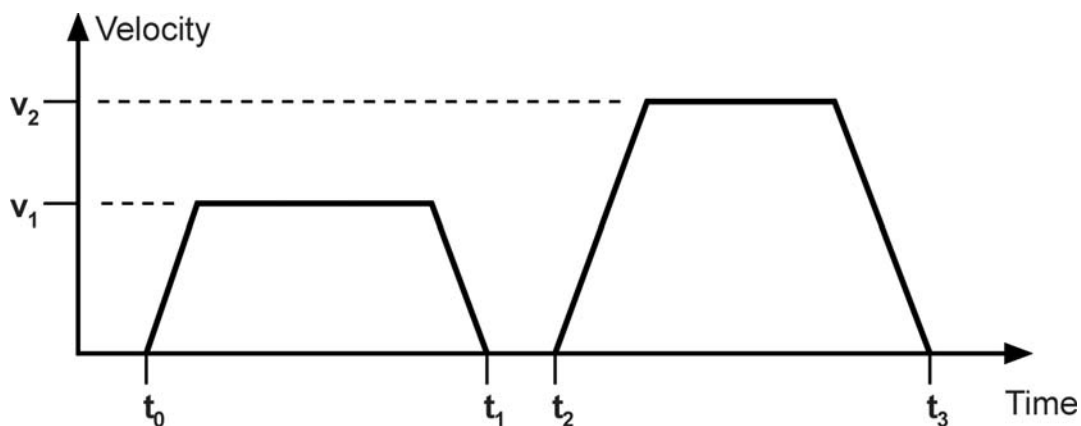


Abbildung 8.18: Einfacher Fahrauftrag

In *Abbildung 8.19* wird eine neue Positionierung bereits gestartet, während sich die Vorherige noch in Bearbeitung befindet. Der Host übergibt hierzu dem Regler das nachfolgende Ziel schon dann, wenn dieser mit dem Löschen des Bits **set_point_acknowledge** signalisiert, dass er den Puffer gelesen und die zugehörige Positionierung gestartet hat. Die Positionierungen werden auf diese Weise nahtlos aneinander gereiht. Damit der Regler zwischen den einzelnen Positionierungen nicht jedes Mal kurzzeitig auf Null abbremst, sollte für diese Betriebsart das Objekt **end_velocity** mit dem gleichen Wert wie das Objekt **profile_velocity** beschrieben werden.

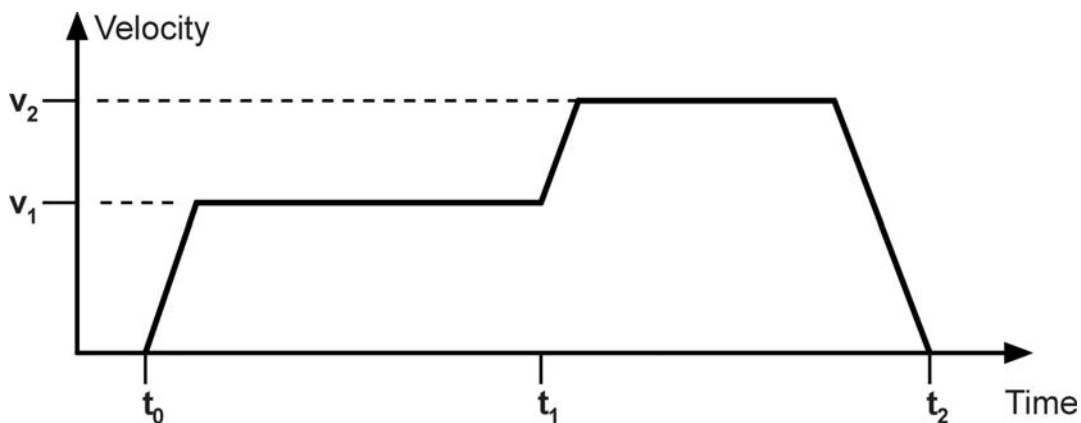


Abbildung 8.19: Lückenlose Folge von Fahraufträgen

Wenn im **controlword** neben dem Bit **new_set_point** auch das Bit **change_set_immediately** auf „1“ gesetzt wird, weist der Host den Regler damit an, *sofort* den neuen Fahrauftrag zu beginnen. Ein bereits in Bearbeitung befindlicher Fahrauftrag wird in diesem Fall abgebrochen.

8.4 Interpolated Position Mode

8.4.1 Übersicht

Der Interpolated Position Mode (**IP**) ermöglicht die Vorgabe von Lagesollwerten in einer mehrachsigen Anwendung des Reglers. Dazu werden in einem festen Zeitraster (Synchronisations-Intervall) Synchronisations-Telegramme (SYNC) und Lagesollwerte von einer übergeordneten Steuerung vorgegeben. Da in der Regel das Intervall größer als ein Lagereglerzyklus ist, interpoliert der Regler selbständig die Datenwerte zwischen zwei vorgegebenen Positionswerten, wie in der folgenden Grafik skizziert.

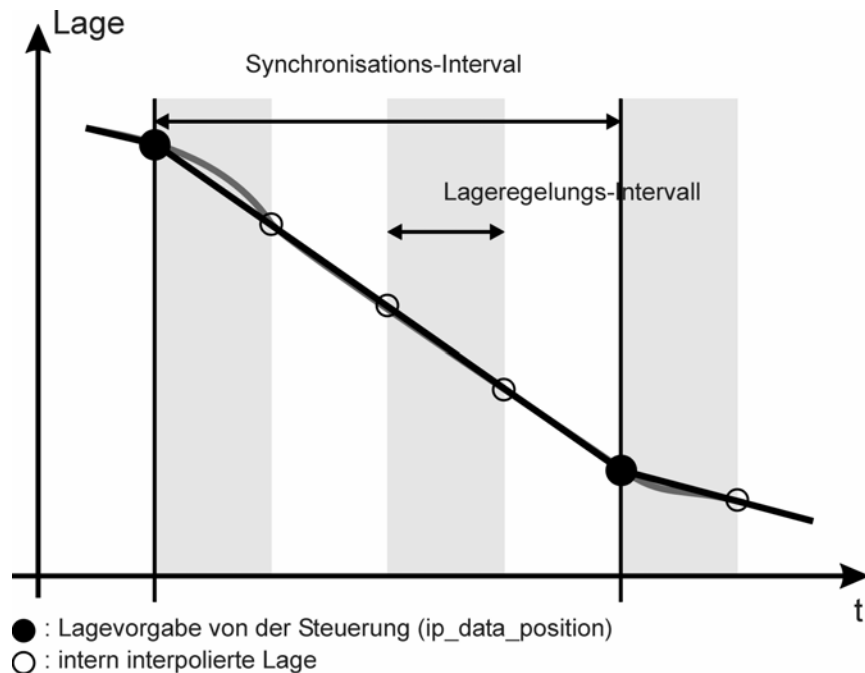


Abbildung 8.20: Fahrauftrag Lineare Interpolation zwischen zwei Datenwerten

Im Folgenden sind zunächst die für den **interpolated position mode** benötigten Objekte beschrieben. In einer anschließenden Funktionsbeschreibung wird umfassend auf die Aktivierung und die Reihenfolge der Parametrierung eingegangen.

8.4.2 Beschreibung der Objekte

8.4.2.1 In diesem Kapitel behandelte Objekte

Index	Objekt	Name	Typ	Attr.
60C0 _h	VAR	interpolation_submode_select	INT16	rw
60C1 _h	REC	interpolation_data_record		rw
60C2 _h	REC	interpolation_time_period		rw
60C3 _h	ARRAY	interpolation_sync_definition	UINT8	rw
60C4 _h	REC	interpolation_data_configuration		rw

8.4.2.2 Betroffene Objekte aus anderen Kapiteln

Index	Objekt	Name	Typ	Kapitel
6040 _h	VAR	controlword	INT16	7 Gerätesteuerung
6041 _h	VAR	statusword	UINT16	7 Gerätesteuerung
6093 _h	ARRAY	position_factor	UINT32	6.3 Umrechnungsfaktoren
6094 _h	ARRAY	velocity_encoder_factor	UINT32	6.3 Umrechnungsfaktoren
6097 _h	ARRAY	acceleration_factor	UINT32	6.3 Umrechnungsfaktoren

8.4.2.3 Objekt 60C0_h: interpolation_submode_select

Über das Objekt **interpolation_submode_select** wird der Typ der Interpolation festgelegt. Zur Zeit ist nur die herstellerepezifische Variante „Lineare Interpolation ohne Puffer“ verfügbar.

Index	60C0_h
Name	interpolation_submode_select
Object Code	VAR
Data Type	INT16

Access	rw
PDO Mapping	yes
Units	--
Value Range	-2
Default Value	-2

Wert	Interpolationstyp
-2	Lineare Interpolation ohne Puffer

8.4.2.4 Objekt 60C1_h: interpolation_data_record


Der Objekt-Record **interpolation_data_record** repräsentiert den eigentlichen Datensatz. Er besteht aus einem Eintrag für den Lagewert (**ip_data_position**) und einem Steuerwort (**ip_data_controlword**), welches angibt, ob der Lagewert absolut oder relativ zu interpretieren ist. Die Angabe des Steuerworts ist optional. Wird er nicht angegeben, wird der Lagewert als absolut interpretiert. Soll das Steuerwort mit angegeben werden, muss aus Gründen der Datenkonsistenz zuerst Subindex 2 (**ip_data_controlword**) und anschließend Subindex 1 (**ip_data_position**) geschrieben werden, da intern die Datenübernahme mit Schreibzugriff auf **ip_data_position** ausgelöst wird.

Index	60C1_h
Name	interpolation_data_record
Object Code	RECORD
No. of Elements	2

Sub-Index	01_h
Description	ip_data_position
Data Type	INT32
Access	rw
PDO Mapping	yes
Units	position units
Value Range	--
Default Value	--

Sub-Index	02_h
Description	ip_data_controlword
Data Type	UINT8
Access	rw
PDO Mapping	yes
Units	--
Value Range	0, 1
Default Value	0

Wert	ip_data_position ist
0	Absolute Position
1	Relative Entfernung

 Die interne Datenübernahme erfolgt bei Schreibzugriff auf Subindex 1. Soll außerdem Subindex 2 verwendet werden, muss dieser vor Subindex 1 beschrieben werden.

8.4.2.5 Objekt 60C2_h: interpolation_time_period


Über den Objekt-Record **interpolation_time_period** kann das Synchronisations-Intervall eingestellt werden. Über **ip_time_index** wird die Einheit (ms oder 1/10 ms) des Intervalls festgelegt, welches über **ip_time_units** parametrisiert wird. Zur Synchronisation wird die komplette Reglerkaskade (Strom-, Drehzahl- und Lageregler) auf den externen Takt auf-synchronisiert. Die Änderung des Synchronisationsintervalls wird daher nur nach einem Reset wirksam. Soll das Interpolationsintervall über den CAN-Bus geändert werden, muss daher der Parametersatz gesichert (siehe Abschnitt 6: *Parameter einstellen, ab Seite 61*) und ein Reset ausgeführt werden (siehe Abschnitt 5.6: *Netzwerkmanagement (NMT-Service), ab Seite 52*), damit das neue Synchronisations-Intervall wirksam wird. Das Synchronisations-Intervall muss exakt eingehalten werden.

Index	60C2_h
Name	interpolation_time_period
Object Code	RECORD
No. of Elements	2

Sub-Index	01_h
Description	ip_time_units
Data Type	UINT8
Access	rw
PDO Mapping	yes
Units	gemäß ip_time_index
Value Range	ip_time_index = -3: 1, 2, ..., 9, 10 ip_time_index = -4: 10, 20, ..., 90, 100
Default Value	--

Sub-Index	02_h
Description	ip_time_index
Data Type	INT8
Access	rw
PDO Mapping	yes
Units	--
Value Range	-3, -4
Default Value	-3

Wert	ip_time_units wird angegeben in
-3	10 ⁻³ Sekunden (ms)
-4	10 ⁻⁴ Sekunden (0.1 ms)

 Die Änderung des Synchronisationsintervalls wird nur nach einem Reset wirksam. Soll das Interpolationsintervall über den CAN-Bus geändert werden, muss der Parametersatz gesichert und ein Reset ausgeführt werden.

8.4.2.6 Objekt 60C3_h: interpolation_sync_definition

Über das Objekt **interpolation_sync_definition** wird die Art (**synchronize_on_group**) und die Anzahl (**ip_sync_every_n_event**) von Synchronisations-Telegrammen pro Synchronisations-Intervall vorgegeben. Für die SERVOTEC S2 -Reihe kann nur das Standard-SYNC-Telegramm und 1 SYNC pro Intervall eingestellt werden.

Index	60C3_h
Name	interpolation_sync_definition
Object Code	ARRAY
No. of Elements	2
Data Type	UINT8

Sub-Index	01_h
Description	synchronize_on_group
Access	rw
PDO Mapping	yes
Units	--
Value Range	0
Default Value	0

Wert	Bedeutung
0	Standard SYNC-Telegramm verwenden

Sub-Index	02_h
Description	ip_sync_every_n_event
Access	rw
PDO Mapping	yes
Units	--
Value Range	1
Default Value	1

8.4.2.7 Objekt 60C4_h: interpolation_data_configuration

Über den Objekt-Record **interpolation_data_configuration** kann die Art (**buffer_organisation**) und Größe (**max_buffer_size**, **actual_buffer_size**) eines eventuell vorhandenen Puffers sowie der Zugriff auf diesen (**buffer_position**, **buffer_clear**) konfiguriert werden. Über das Objekt **size_of_data_record** kann die Größe eines Puffer-Elements ausgelesen werden. Obwohl bei der Interpolationsart „Lineare Interpolation ohne Puffer“ kein Puffer zur Verfügung steht, muss der Zugriff über das Objekt **buffer_clear** allerdings auch in diesem Fall freigegeben werden.

Index	60C4_h
Name	interpolation_data_configuration
Object Code	RECORD
No. of Elements	6

Sub-Index	01_h
Description	max_buffer_size
Data Type	UINT32
Access	ro
PDO Mapping	no
Units	--
Value Range	0
Default Value	0

Sub-Index	02_h
Description	actual_size
Data Type	UINT32
Access	rw
PDO Mapping	yes
Units	--
Value Range	0...max_buffer_size
Default Value	0

Sub-Index	03_h
Description	buffer_organisation
Data Type	UINT8
Access	rw
PDO Mapping	yes
Units	--
Value Range	0
Default Value	0

Wert	Bedeutung
0	FIFO

Sub-Index	04_h
Description	buffer_position
Data Type	UINT16
Access	rw
PDO Mapping	yes
Units	--
Value Range	0
Default Value	0

Sub-Index	05_h
Description	size_of_data_record
Data Type	UINT8
Access	wo
PDO Mapping	yes
Units	--
Value Range	2
Default Value	2

Sub-Index	06_h
Description	buffer_clear
Data Type	UINT8
Access	wo
PDO Mapping	yes
Units	--
Value Range	0, 1
Default Value	0

Wert	Bedeutung
0	Puffer löschen / Zugriff auf 60C1 _h nicht erlaubt
1	Zugriff auf 60C1 _h freigegeben

8.4.3 Funktionsbeschreibung

8.4.3.1 Vorbereitende Parametrierung

Bevor der Regler in die Betriebsart **interpolated position mode** geschaltet werden kann, müssen diverse Einstellungen vorgenommen werden: Dazu zählen die Einstellung des Interpolations-Intervalls (**interpolation_time_period**), also der Zeit zwischen zwei SYNC-Telegrammen, der Interpolationstyp (**interpolation_submode_select**) und die Art der Synchronisation (**interpolation_sync_definition**). Zusätzlich muss der Zugriff auf den Positionspuffer über das Objekt **buffer_clear** freigegeben werden.

BEISPIEL



Aufgabe		CAN-Objekt / COB
Interpolationsart	-2	60C0h, interpolation_submode_select = -2
Zeiteinheit	0.1 ms	60C2h_02h, interpolation_time_index = -04
Zeitintervall	4 ms	60C2h_01h, interpolation_time_units = 40
Parameter sichern		1010h_01h, save_all_parameters
Reset ausführen		NMT reset node
Warten auf Bootup		Bootup-Nachricht
Puffer-Freigabe	1	60C4h_06h, buffer_clear = 1
SYNC erzeugen		SYNC (Raster 4 ms)

8.4.3.2 Aktivierung des Interpolated Position Mode und Aufsynchronisation

Der IP wird über das Objekt **modes_of_operation (6060_h)** aktiviert. Ab diesem Zeitpunkt versucht der Regler sich auf das externe Zeitraster, welches durch die SYNC-Telegramme vorgegeben wird, aufzusynchronisieren. Konnte sich der Regler erfolgreich aufsynchronisieren, meldet er die Betriebsart **interpolated position mode** im Objekt **modes_of_operation_display (6061_h)**. Während der Aufsynchronisation meldet der Regler **ungültige Betriebsart (-1)** zurück. Werden nach der erfolgten Aufsynchronisation die SYNC-Telegramme nicht im richtigen Zeitraster gesendet, wechselt der Regler zurück in die **ungültige Betriebsart**.

Ist die Betriebsart eingenommen, kann die Übertragung von Positionsdaten an den Antrieb beginnen. Sinnvollerweise liest dazu die übergeordnete Steuerung zunächst die aktuelle Istposition aus dem Regler aus und schreibt diese zyklisch als neuen Sollwert (**interpolation_data_record**) in den Regler. Über Handshake-Bits des **controlword** und des **statusword** wird die Übernahme der Daten durch den Regler aktiviert. Durch Setzen des Bits **enable_ip_mode** im **controlword** zeigt der Host an, dass mit der Auswertung der Lagedaten begonnen werden soll. Erst wenn der Regler über das Statusbit **ip_mode_selected** im **statusword** dieses quittiert, werden die Datensätze ausgewertet.

Im Einzelnen ergibt sich daher folgende Zuordnung und der folgende Ablauf:

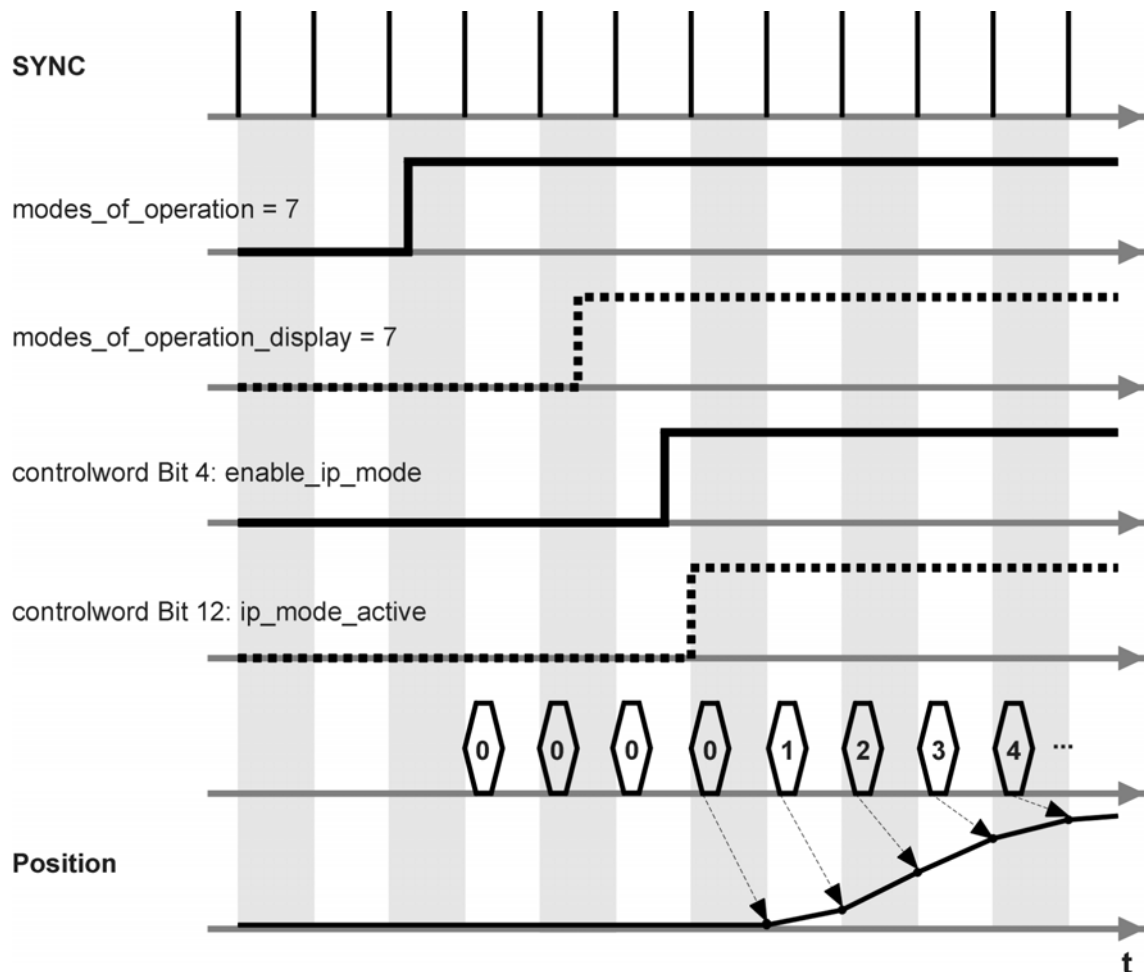


Abbildung 8.21: Aufsynchronisation und Datenfreigabe

Nr.	Ereignis	CAN-Objekt
1	SYNC- Nachrichten erzeugen	
2	Anforderung der Betriebsart ip:	6060 _h , modes_of_operation = 07
3	Warten bis Betriebsart eingenommen	6061 _h , modes_of_operation_display = 07
4	Auslesen der akt. Istposition	6064 _h , position_actual_value
5	Zurückschreiben als aktuelle Sollposition	60C1 _h _01 _h , ip_data_position
6	Start der Interpolation	6040 _h , controlword, enable_ip_mode
7	Quittierung durch Regler	6041 _h , statusword, ip_mode_active
8	Ändern der aktuellen Sollposition gemäß Trajektorie	60C1 _h _01 _h , ip_data_position

Nach Beendigung des synchronen Fahrvorgangs kann durch Löschen des Bits **enable_ip_mode** die weitere Auswertung von Lagewerten verhindert werden. Anschließend kann gegebenenfalls in eine andere Betriebsart umgeschaltet werden.

8.4.3.3 Unterbrechung der Interpolation im Fehlerfall

Wird eine laufende Interpolation (**ip_mode_active** gesetzt) durch das Auftreten eines Reglerfehlers unterbrochen, verhält sich der Antrieb zunächst so, wie für den jeweiligen Fehler spezifiziert (z.B. Wegnahme der Reglerfreigabe und Wechsel in den Zustand **SWITCH_ON_DISABLED**).

Die Interpolation kann dann nur durch eine erneute Aufsynchronisation fortgesetzt werden, da der Regler wieder in den Zustand **OPERATION_ENABLE** gebracht werden muss, wodurch das Bit **ip_mode_active** gelöscht wird.

8.5 Betriebsart Drehzahlregelung (Profile Velocity Mode)

8.5.1 Übersicht

Der drehzahlgeregelte Betrieb (Profile Velocity Mode) beinhaltet die folgenden Unterfunktionen:

- Sollwert-Erzeugung durch den Rampen-Generator
- Drehzahlerfassung über den Winkelgeber durch Differentiation
- Drehzahlregelung mit geeigneten Eingabe- und Ausgabesignalen
- Begrenzung des Drehmomenten-Sollwertes (**torque_demand_value**)
- Überwachung der Ist-Geschwindigkeit (**velocity_actual_value**) mit der Fenster-Funktion/Schwelle

Die Bedeutung der folgenden Parameter ist im Abschnitt 8.3: *Betriebsart Positionieren (Profile Position Mode)*, ab Seite 191 beschrieben: **profile_acceleration**, **profile_deceleration**, **quick_stop**.

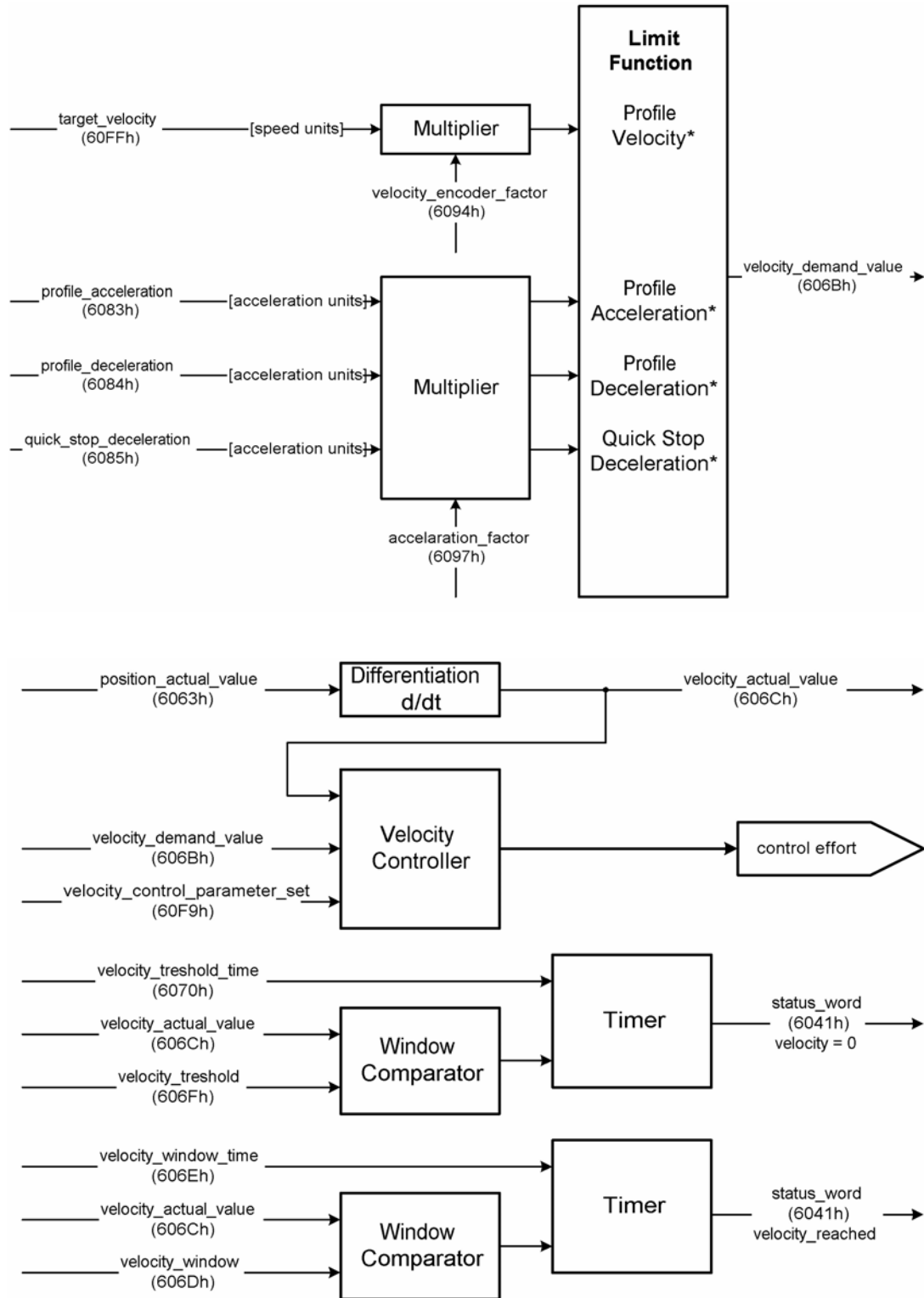


Abbildung 8.22: Struktur des drehzahlgeregelten Betriebs (Profile Velocity Mode)

8.5.2 Beschreibung der Objekte

8.5.2.1 In diesem Kapitel behandelte Objekte

Index	Objekt	Name	Typ	Attr.
6069 _h	VAR	velocity_sensor_actual_value	INT32	ro
606A _h	VAR	sensor_selection_code	INT16	rw
606B _h	VAR	velocity_demand_value	INT32	ro
202E _h	VAR	velocity_demand_sync_value	INT32	ro
606C _h	VAR	velocity_actual_value	INT32	ro
606D _h	VAR	velocity_window	UINT16	rw
606E _h	VAR	velocity_window_time	UINT16	rw
606F _h	VAR	velocity_threshold	UINT16	rw
6080 _h	VAR	max_motor_speed	UINT32	rw
60FF _h	VAR	target_velocity	INT32	rw

8.5.2.2 Betroffene Objekte aus anderen Abschnitten

Index	Objekt	Name	Typ	Abschnitt
6040 _h	VAR	controlword	INT16	7 Gerätesteuerung
6041 _h	VAR	statusword	UINT16	7 Gerätesteuerung
6063 _h	VAR	position_actual_value*	INT32	6.7 Lageregler
6071 _h	VAR	target_torque	INT16	8.7 Momentenregler
6072 _h	VAR	max_torque_value	UINT16	8.7 Momentenregler
607E _h	VAR	polarity	UINT8	6.3 Umrechnungsfaktoren
6083 _h	VAR	profile_acceleration	UINT32	8.3 Positionieren
6084 _h	VAR	profile_deceleration	UINT32	8.3 Positionieren
6085 _h	VAR	quick_stop_deceleration	UINT32	8.3 Positionieren
6086 _h	VAR	motion_profile_type	INT16	8.3 Positionieren
6094 _h	ARRAY	velocity_encoder_factor	UINT32	6.3 Umrechnungsfaktoren

8.5.2.3 Objekt 6069_h: velocity_sensor_actual_value

Mit dem Objekt **velocity_sensor_actual_value** kann der Wert eines möglichen Geschwindigkeitsgebers in internen Einheiten ausgelesen werden. Bei der SERVOTEC S2-Familie kann kein separater Drehzahlgeber angeschlossen werden. Zur Bestimmung des Drehzahl-Istwertes sollte daher grundsätzlich das Objekt **606C_h** verwendet werden.

Index	6069_h
Name	velocity_sensor_actual_value
Object Code	VAR
Data Type	INT32

Access	ro
PDO Mapping	yes
Units	U / 4096 min
Value Range	--
Default Value	--

8.5.2.4 Objekt 606A_h: sensor_selection_code

Mit diesem Objekt kann der Geschwindigkeitssensor ausgewählt werden. Zur Zeit ist kein separater Geschwindigkeitssensor vorgesehen. Deshalb ist nur der standardmäßige Winkelgeber anwählbar.

Index	606A_h
Name	sensor_selection_code
Object Code	VAR
Data Type	INT16

Access	rw
PDO Mapping	yes
Units	--
Value Range	0
Default Value	0

8.5.2.5 Objekt 606B_h: velocity_demand_value

Mit diesem Objekt kann der aktuelle Drehzahl Sollwert des Drehzahlreglers ausgelesen werden. Auf diesen wirkt der Sollwert vom Rampen-Generator bzw. des Fahrkurven-Generators. Bei aktiviertem Lageregler wird außerdem dessen Korrekturgeschwindigkeit addiert.

Index	606B_h
Name	velocity_demand_value
Object Code	VAR
Data Type	INT32

Access	ro
PDO Mapping	yes
Units	speed units
Value Range	--
Default Value	--

8.5.2.6 Objekt 202E_h: velocity_demand_sync_value

Über dieses Objekt kann die Soll-Drehzahl des Synchronisationsgeber ausgelesen werden. Diese wird durch das Objekt **2022_h synchronization_encoder_select** (siehe Abschnitt 6.11: *Soll-/Istwertaufschaltung, ab Seite 122*) definiert. Dieses Objekt wird in benutzerdefinierten Einheiten angegeben.

Index	202E_h
Name	velocity_demand_sync_value
Object Code	VAR
Data Type	INT32

Ab Firmware 3.2.0.1.1

Access	ro
PDO Mapping	no
Units	velocity units
Value Range	--
Default Value	--

8.5.2.7 Objekt 606C_h: velocity_actual_value

Über das Objekt **velocity_actual_value** kann der Drehzahl-Istwert ausgelesen werden.

Index	606C_h
Name	velocity_actual_value
Object Code	VAR
Data Type	INT32

Access	ro
PDO Mapping	yes
Units	speed units
Value Range	--
Default Value	--

8.5.2.8 Objekt 2074_h: velocity_actual_value_filtered

Über das Objekt **velocity_actual_value_filtered** kann ein gefilterter Drehzahl- Istwert ausgelesen werden, der allerdings nur zu Anzeigezwecken verwendet werden sollte. Im Gegensatz zu **velocity_actual_value** wird **velocity_actual_value_filtered** nicht zur Regelung, wohl aber für den Durchdrehschutz des Reglers verwendet. Die Filterzeitkonstante kann über das Objekt **2073_h** (**velocity_display_filter_time**) eingestellt werden (siehe Abschnitt 6.6.2.2: *Objekt 2073_h: velocity_display_filter_time*, Seite 97).

Index	2074_h
Name	velocity_actual_value_filtered
Object Code	VAR
Data Type	INT32

Ab Firmware 3.5.x.1.1

Access	ro
PDO Mapping	yes
Units	speed units
Value Range	--
Default Value	--

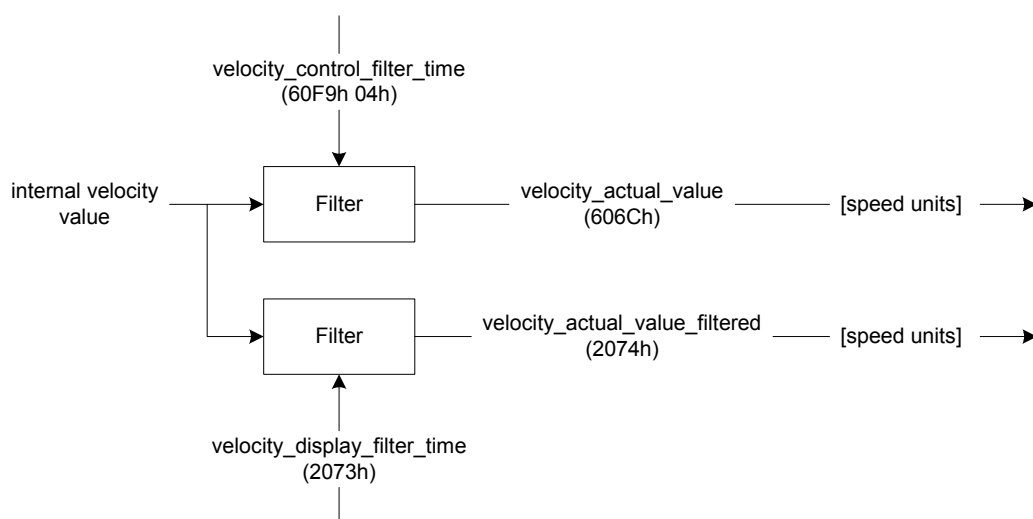


Abbildung 8.23: Ermittlung von velocity_actual_value und velocity_actual_value_filtered

8.5.2.9 Objekt 606D_h: velocity_window

Das Objekt **velocity_window** dient zur Einstellung des Fensterkomparators. Dieser vergleicht den Drehzahl-Istwert mit der vorgegebenen Endgeschwindigkeit (Objekt **60FF_h**: **target_velocity**). Ist die Differenz eine bestimmte Zeitdauer kleiner als hier angegeben, so wird das Bit 10 **target_reached** im Objekt **statusword** gesetzt (siehe auch Abschnitt 8.5.2.10: *Objekt 606E_h: velocity_window_time*, Seite 215).

Index	606D_h
Name	velocity_window
Object Code	VAR
Data Type	UINT16

Access	rw
PDO Mapping	yes
Units	speed units
Value Range	0...65536 min ⁻¹
Default Value	4 min ⁻¹

8.5.2.10 Objekt 606E_h: velocity_window_time

Das Objekt **velocity_window_time** dient neben dem Objekt **606D_h: velocity_window** der Einstellung des Fensterkomparators. Die Drehzahl muss die hier spezifizierte Zeit innerhalb des **velocity_window** liegen, damit das Bit 10 **target_reached** im Objekt **statusword** gesetzt wird.

Index	606E_h
Name	velocity_window_time
Object Code	VAR
Data Type	UINT16

Access	rw
PDO Mapping	yes
Units	ms
Value Range	0...4999
Default Value	0

8.5.2.11 Objekt 606F_h: velocity_threshold

Das Objekt **velocity_threshold** gibt an, ab welchem Drehzahl-Istwert der Antrieb als stehend angesehen wird. Wenn der Antrieb den hier vorgegebenen Drehzahlwert für einen bestimmten Zeitraum überschreitet, wird im **statusword** das Bit 12 (velocity = 0) gelöscht. Der Zeitraum wird durch das Objekt **velocity_threshold_time** bestimmt.

Index	606F_h
Name	velocity_threshold
Object Code	VAR
Data Type	UINT16

Access	rw
PDO Mapping	yes
Units	speed units
Value Range	0...65536 min ⁻¹
Default Value	10

8.5.2.12 Objekt 6070_h: velocity_threshold_time

Das Objekt **velocity_threshold_time** gibt an, wie lange der Antrieb den vorgegebenen Drehzahlwert überschreiten darf, bevor im **statusword** das Bit 12 (velocity = 0) gelöscht wird.

Index	6070_h
Name	velocity_threshold_time
Object Code	VAR
Data Type	UINT16

Access	rw
PDO Mapping	yes
Units	ms
Value Range	0...4999
Default Value	0

8.5.2.13 Objekt 6080_h: max_motor_speed

Das Objekt **max_motor_speed** gibt die höchste erlaubte Drehzahl für den Motor in min⁻¹. Das Objekt wird benutzt, um den Motor zu schützen und kann dem Motordatenblatt entnommen werden. Der Drehzahl-Sollwert wird auf diesen Wert begrenzt.

Index	6080_h
Name	max_motor_speed
Object Code	VAR
Data Type	UINT16

Access	rw
PDO Mapping	yes
Units	min ⁻¹
Value Range	0... 32768 min ⁻¹
Default Value	32768 min ⁻¹

8.5.2.14 Objekt 60FF_h: target_velocity

Das Objekt **target_velocity** ist die Sollwertvorgabe für den Rampen-Generator.

Index	60FF_h
Name	target_velocity
Object Code	VAR
Data Type	INT32

Access	rw
PDO Mapping	yes
Units	speed units
Value Range	--
Default Value	--

8.6 Drehzahl-Rampen

Wird als **modes_of_operation** *profile_velocity_mode* gewählt, wird grundsätzlich auch die Sollwerttrappe aktiviert. Somit ist es möglich über die Objekte **profile_acceleration** und **profile_deceleration** eine sprungförmige Sollwertänderung auf eine bestimmte Drehzahländerungen pro Zeit zu begrenzen. Der Regler ermöglicht es, nicht nur unterschiedliche Beschleunigungen für Bremsen und Beschleunigungen anzugeben, sondern noch zusätzlich nach positiver und negativer Drehzahl zu unterscheiden.

Die folgende Abbildung verdeutlicht dieses Verhalten:

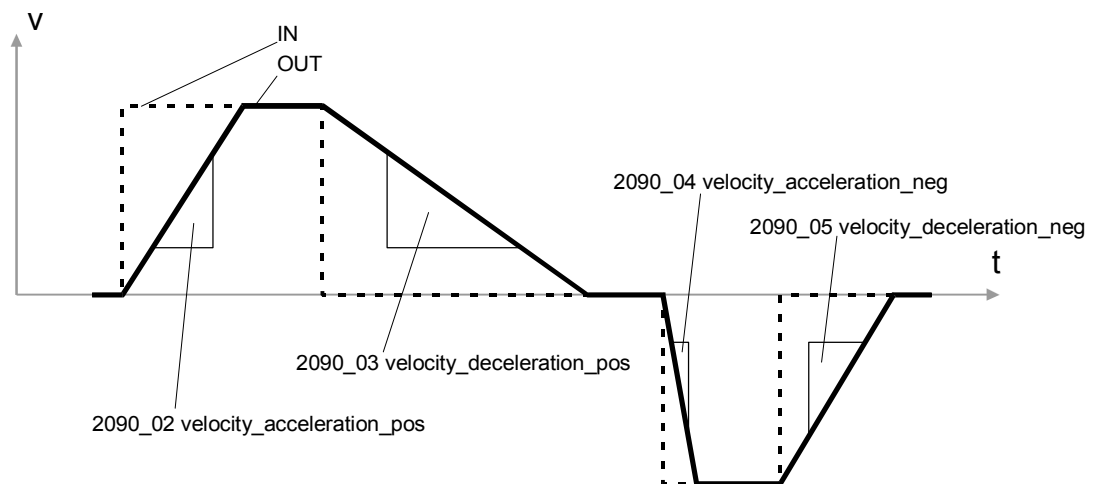


Abbildung 8.24: Drehzahlrampen

Um diese 4 Beschleunigungen einzeln parametrieren zu können, ist die Objektgruppe **velocity_ramps** vorhanden. Es ist zu beachten, dass die Objekte **profile_acceleration** und **profile_deceleration** die gleichen internen Beschleunigungen verändern, wie die **velocity_ramps**. Wird die **profile_acceleration** geschrieben, werden gemeinsam **velocity_acceleration_pos** und **velocity_acceleration_neg** geändert, wird die **profile_deceleration** geschrieben, werden gemeinsam **velocity_deceleration_pos** und **velocity_deceleration_neg** geändert. Mit dem Objekt **velocity_ramps_enable** lässt sich festlegen, ob die Sollwerte über den Rampengenerator geführt werden, oder nicht.

Index	2090 _h
Name	velocity_ramps
Object Code	RECORD
No. of Elements	5

Ab Firmware 3.0.x.1.1

Ab Firmware 3.0.x.1.1

Sub-Index	01_h
Description	velocity_ramps_enable
Data Type	UINT8
Access	rw
PDO Mapping	no
Units	--
Value Range	0: Sollwert NICHT über den Rampengenerator 1: Sollwert über den Rampengenerator
Default Value	1

Sub-Index	02_h
Description	velocity_acceleration_pos
Data Type	INT32
Access	rw
PDO Mapping	no
Units	acceleration units
Value Range	--
Default Value	14 100 min ⁻¹ /s

Sub-Index	03_h
Description	velocity_deceleration_pos
Data Type	INT32
Access	rw
PDO Mapping	no
Units	acceleration units
Value Range	--
Default Value	14 100 min ⁻¹ /s

Sub-Index	04_h
Description	velocity_acceleration_neg
Data Type	INT32
Access	rw
PDO Mapping	no
Units	acceleration units
Value Range	--
Default Value	14 100 min ⁻¹ /s

Sub-Index	05_h
Description	velocity_deceleration_neg
Data Type	INT32
Access	rw
PDO Mapping	no
Units	acceleration units
Value Range	--
Default Value	14 100 min ⁻¹ /s

8.7 Betriebsart Momentenregelung (Profile Torque Mode)

8.7.1 Übersicht

Dieses Kapitel beschreibt den drehmomentengeregelten Betrieb. Diese Betriebsart erlaubt es, dass dem Regler ein externer Momenten-Sollwert **target_torque** vorgegeben wird, welcher durch den integrierten Rampen-Generator geglättet werden kann. Somit ist es möglich, dass dieser Regler auch für Bahnsteuerungen eingesetzt werden kann, bei denen sowohl der Lageregler als auch der Drehzahlregler auf einen externen Rechner verlagert sind.

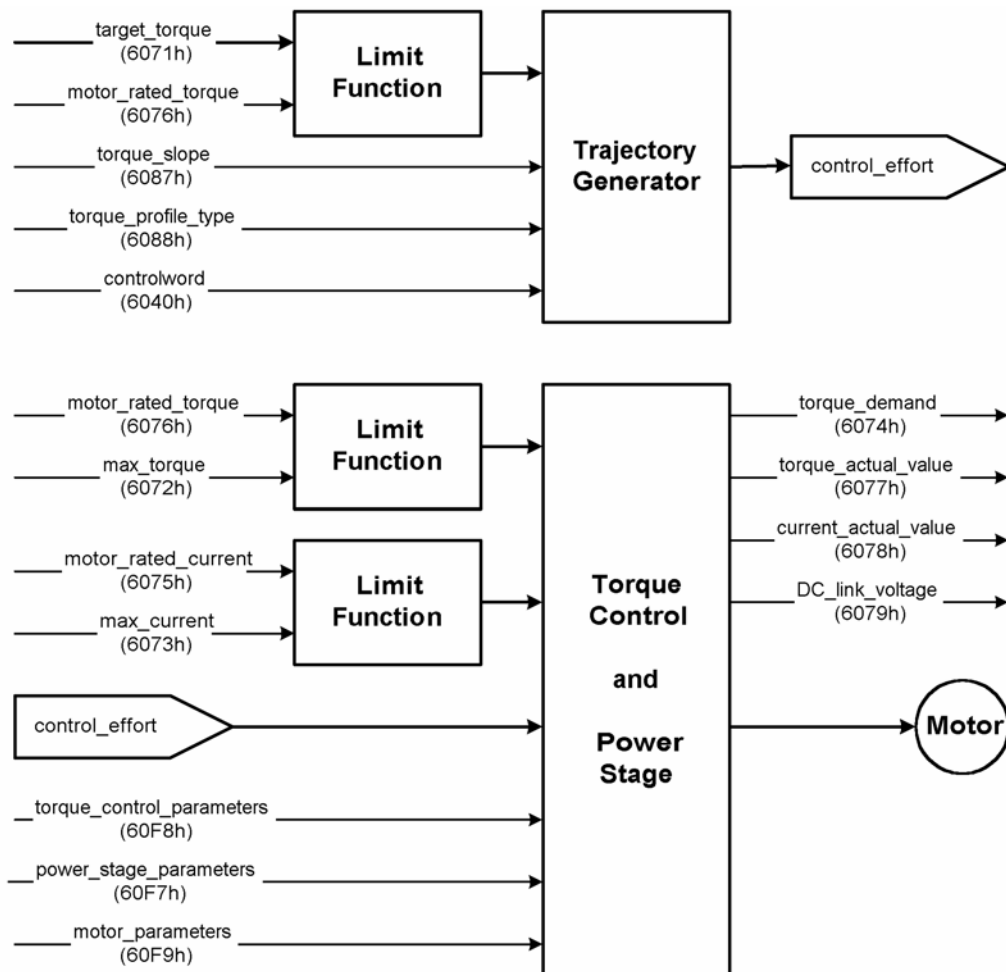


Abbildung 8.25: Struktur des drehmomentgeregelten Betriebs

Für den Rampengenerator müssen die Parameter Rampensteilheit **torque_slope** und Rampenform **torque_profile_type** vorgegeben werden.

Wenn im **controlword** das Bit 8 **halt** gesetzt wird, senkt der Rampen-Generator das Drehmoment bis auf Null ab. Entsprechend erhöht er es wieder auf das Sollmoment **target_torque**, wenn das Bit 8 wieder gelöscht wird. In beiden Fällen berücksichtigt der Rampen-Generator die Rampensteilheit **torque_slope** und die Rampenform **torque_profile_type**.

Alle Definitionen innerhalb dieses Dokumentes beziehen sich auf drehbare Motoren. Wenn lineare Motoren benutzt werden, müssen sich alle „Drehmoment“-Objekte statt dessen auf eine „Kraft“ beziehen. Der Einfachheit halber sind die Objekte nicht doppelt vertreten und ihre Namen sollten nicht verändert werden.

Die Betriebsarten Positionierbetrieb (Profile Position Mode) und Drehzahlregler (Profile Velocity Mode) benötigen für ihre Funktion den Momentenregler. Deshalb ist es immer notwendig, diesen zu parametrieren.

8.7.2 Beschreibung der Objekte

8.7.2.1 In diesem Kapitel behandelte Objekte

Index	Objekt	Name	Typ	Attr.
6071 _h	VAR	target_torque	INT16	rw
6072 _h	VAR	max_torque	UINT16	rw
6074 _h	VAR	torque_demand_value	INT16	ro
6076 _h	VAR	motorRatedTorque	UINT32	rw
6077 _h	VAR	torque_actual_value	INT16	ro
6078 _h	VAR	current_actual_value	INT16	ro
6079 _h	VAR	DC_link_circuit_voltage	UINT32	ro
6087 _h	VAR	torque_slope	UINT32	rw
6088 _h	VAR	torque_profile_type	INT16	rw
60F7 _h	RECORD	power_stage_parameters		rw
60F6 _h	RECORD	torque_control_parameters		rw

8.7.2.2 Betroffene Objekte aus anderen Kapiteln

Index	Objekt	Name	Typ	Kapitel
6040 _h	VAR	controlword	INT16	7 Gerätesteuerung
60F9 _h	RECORD	motor_parameters		6.5 Stromregler u. Motoranpassung
6075 _h	VAR	motorRatedCurrent	UINT32	6.5 Stromregler u. Motoranpassung
6073 _h	VAR	max_current	UINT16	6.5 Stromregler u. Motoranpassung

8.7.2.3 Objekt 6071_h: target_torque

Dieser Parameter ist im drehmomentengeregelten Betrieb (Profile Torque Mode) der Eingabewert für den Drehmomentenregler. Er wird in Tausendstel des Nennmomentes (Objekt 6076_h) angegeben.

Index	6071_h
Name	target_torque
Object Code	VAR
Data Type	INT16

Access	rw
PDO Mapping	yes
Units	motorRatedTorque / 1000
Value Range	-32768...32768
Default Value	0

8.7.2.4 Objekt 6072_h: max_torque

Dieser Wert stellt das höchstzulässige Drehmoment des Motors dar. Es wird in Tausendstel des Nennmomentes (Objekt 6076_h) angegeben. Wenn zum Beispiel kurzzeitig eine zweifache Überlastung des Motors zulässig ist, so ist hier der Wert 2000 einzutragen.



Das Objekt 6072_h: max_torque korrespondiert mit dem Objekt 6073_h: max_current und darf erst beschrieben werden, wenn zuvor das Objekt 6075_h: motorRatedCurrent mit einem gültigen Wert beschrieben wurde.

Index	6072_h
Name	max_torque
Object Code	VAR
Data Type	UINT16

Access	rw
PDO Mapping	yes
Units	motorRatedTorque / 1000
Value Range	1000...65536
Default Value	2023

8.7.2.5 Objekt 6074h: torque_demand_value

Über dieses Objekt kann das aktuelle Sollmoment in Tausendstel des Nennmoments (**6076_h**) ausgelesen werden. Berücksichtigt sind hierbei die internen Begrenzungen des Reglers (Stromgrenzwerte und I²T-Überwachung).

Index	6074_h
Name	torque_demand_value
Object Code	VAR
Data Type	INT16

Access	ro
PDO Mapping	yes
Units	motorRatedTorque / 1000
Value Range	--
Default Value	--

8.7.2.6 Objekt 6076_h: motorRatedTorque

Dieses Objekt gibt das Nennmoment des Motors an. Dieses kann dem Typenschild des Motors entnommen werden. Es ist in der Einheit 0.001 Nm einzugeben.

Index	6076_h
Name	motorRatedTorque
Object Code	VAR
Data Type	UINT32

Access	rw
PDO Mapping	yes
Units	0.001 Nm
Value Range	--
Default Value	296

8.7.2.7 Objekt 6077_h: torque_actual_value

Über dieses Objekt kann der Drehmomenten-Istwert des Motors in Tausendstel des Nennmomentes (Objekt 6076_h) ausgelesen werden.

Index	6077 _h
Name	torque_actual_value
Object Code	VAR
Data Type	INT16

Access	ro
PDO Mapping	yes
Units	motorRatedTorque / 1000
Value Range	--
Default Value	--

8.7.2.8 Objekt 6078_h: current_actual_value

Über dieses Objekt kann der Strom-Istwert des Motors in Tausendstel des Nennstromes (Objekt 6075_h) ausgelesen werden.

Index	6078 _h
Name	current_actual_value
Object Code	VAR
Data Type	INT16

Access	ro
PDO Mapping	yes
Units	motorRatedCurrent / 1000
Value Range	--
Default Value	--

8.7.2.9 Objekt 6079_h: dc_link_circuit_voltage

Über dieses Objekt kann die Zwischenkreisspannung des Reglers ausgelesen werden. Die Spannung wird in der Einheit Millivolt angegeben.

Index	6079_h
Name	dc_link_circuit_voltage
Object Code	VAR
Data Type	UINT32

Access	ro
PDO Mapping	yes
Units	mV
Value Range	--
Default Value	--

8.7.2.10 Objekt 6087_h: torque_slope

Dieser Parameter beschreibt die Änderungsgeschwindigkeit der Sollwertrampe. Diese ist in Tausendstel vom Nennmoment pro Sekunde anzugeben. Beispielsweise wird der Drehmomenten-Sollwert **target_torque** von 0 Nm auf den Wert **motorRatedTorque** erhöht. Wenn der Ausgangswert der zwischengeschalteten Drehmomentenrampe diesen Wert in einer Sekunde erreichen soll, dann ist in diesem Objekt der Wert 1000 einzuschreiben.

Index	6087_h
Name	torque_slope
Object Code	VAR
Data Type	UINT32

Access	rw
PDO Mapping	yes
Units	motorRatedTorque / 1000 s
Value Range	--
Default Value	E310F94 _h

8.7.2.11 Objekt 6088_h: torque_profile_type

Mit dem Objekt **torque_profile_type** wird vorgegeben, mit welcher Kurvenform ein Sollwertsprung ausgeführt wird. Zur Zeit ist in diesem Regler nur die lineare Rampe implementiert, so dass dieses Objekt nur mit dem Wert 0 beschrieben werden kann.

Index	6088 _h
Name	torque_profile_type
Object Code	VAR
Data Type	INT16

Access	rw
PDO Mapping	yes
Units	--
Value Range	0
Default Value	0

Wert	Bedeutung
0	Lineare Rampe


```

#define vendor_id (0x101801 + UINT32 + cRD }
#define product_code (0x101802 + UINT32 + cRD }
#define revision_number (0x101803 + UINT32 + cRD }
#define serial_number (0x101804 + UINT32 + cRD }
#define server_sdo_parameter (0x120000 + UINT8 + cRD }
#define cob_id_client_server (0x120001 + UINT32 + cRD }
#define cob_id_server_client (0x120002 + UINT32 + cRD }
#define receive_pdo_parameter_rpdo1 (0x140000 + UINT8 + cRD }
#define cob_id_used_by_pdo_rpdo1 (0x140001 + UINT32 + cRD + cWR }
#define transmission_type_rpdo1 (0x140002 + UINT8 + cRD + cWR }
#define receive_pdo_parameter_rpdo2 (0x140100 + UINT8 + cRD }
#define cob_id_used_by_pdo_rpdo2 (0x140101 + UINT32 + cRD + cWR }
#define transmission_type_rpdo2 (0x140102 + UINT8 + cRD + cWR }
#define receive_pdo_parameter_rpdo3 (0x140200 + UINT8 + cRD }
#define cob_id_used_by_pdo_rpdo3 (0x140201 + UINT32 + cRD + cWR }
#define transmission_type_rpdo3 (0x140202 + UINT8 + cRD + cWR }
#define receive_pdo_parameter_rpdo4 (0x140300 + UINT8 + cRD }
#define cob_id_used_by_pdo_rpdo4 (0x140301 + UINT32 + cRD + cWR }
#define transmission_type_rpdo4 (0x140302 + UINT8 + cRD + cWR }
#define receive_pdo_mapping_rpdo1 (0x160000 + UINT8 + cRD + cWR }
#define first_mapped_object_rpdo1 (0x160001 + UINT32 + cRD + cWR }
#define second_mapped_object_rpdo1 (0x160002 + UINT32 + cRD + cWR }
#define third_mapped_object_rpdo1 (0x160003 + UINT32 + cRD + cWR }
#define fourth_mapped_object_rpdo1 (0x160004 + UINT32 + cRD + cWR }
#define receive_pdo_mapping_rpdo2 (0x160100 + UINT8 + cRD + cWR }
#define first_mapped_object_rpdo2 (0x160101 + UINT32 + cRD + cWR }
#define second_mapped_object_rpdo2 (0x160102 + UINT32 + cRD + cWR }
#define third_mapped_object_rpdo2 (0x160103 + UINT32 + cRD + cWR }
#define fourth_mapped_object_rpdo2 (0x160104 + UINT32 + cRD + cWR }
#define receive_pdo_mapping_rpdo3 (0x160200 + UINT8 + cRD + cWR }
#define first_mapped_object_rpdo3 (0x160201 + UINT32 + cRD + cWR }
#define second_mapped_object_rpdo3 (0x160202 + UINT32 + cRD + cWR }
#define third_mapped_object_rpdo3 (0x160203 + UINT32 + cRD + cWR }
#define fourth_mapped_object_rpdo3 (0x160204 + UINT32 + cRD + cWR }
#define receive_pdo_mapping_rpdo4 (0x160300 + UINT8 + cRD + cWR }
#define first_mapped_object_rpdo4 (0x160301 + UINT32 + cRD + cWR }
#define second_mapped_object_rpdo4 (0x160302 + UINT32 + cRD + cWR }
#define third_mapped_object_rpdo4 (0x160303 + UINT32 + cRD + cWR }
#define fourth_mapped_object_rpdo4 (0x160304 + UINT32 + cRD + cWR }
#define transmit_pdo_parameter_tpdo1 (0x180000 + UINT8 + cRD }
#define cob_id_used_by_pdo_tpdo1 (0x180001 + UINT32 + cRD + cWR }
#define transmission_type_tpdo1 (0x180002 + UINT8 + cRD + cWR }
#define inhibit_time_tpdo1 (0x180003 + UINT16 + cRD + cWR }
#define transmit_pdo_parameter_tpdo2 (0x180100 + UINT8 + cRD }
#define cob_id_used_by_pdo_tpdo2 (0x180101 + UINT32 + cRD + cWR }
#define transmission_type_tpdo2 (0x180102 + UINT8 + cRD + cWR }
#define inhibit_time_tpdo2 (0x180103 + UINT16 + cRD + cWR }
#define transmit_pdo_parameter_tpdo3 (0x180200 + UINT8 + cRD }
#define cob_id_used_by_pdo_tpdo3 (0x180201 + UINT32 + cRD + cWR }
#define transmission_type_tpdo3 (0x180202 + UINT8 + cRD + cWR }
#define inhibit_time_tpdo3 (0x180203 + UINT16 + cRD + cWR }
#define transmit_pdo_parameter_tpdo4 (0x180300 + UINT8 + cRD }
#define cob_id_used_by_pdo_tpdo4 (0x180301 + UINT32 + cRD + cWR }
#define transmission_type_tpdo4 (0x180302 + UINT8 + cRD + cWR }
#define inhibit_time_tpdo4 (0x180303 + UINT16 + cRD + cWR }
#define transmit_pdo_mapping_tpdo1 (0x1A0000 + UINT8 + cRD + cWR }
#define first_mapped_object_tpdo1 (0x1A0001 + UINT32 + cRD + cWR }
#define second_mapped_object_tpdo1 (0x1A0002 + UINT32 + cRD + cWR }
#define third_mapped_object_tpdo1 (0x1A0003 + UINT32 + cRD + cWR }
#define fourth_mapped_object_tpdo1 (0x1A0004 + UINT32 + cRD + cWR }
#define transmit_pdo_mapping_tpdo2 (0x1A0100 + UINT8 + cRD + cWR }
#define first_mapped_object_tpdo2 (0x1A0101 + UINT32 + cRD + cWR }
#define second_mapped_object_tpdo2 (0x1A0102 + UINT32 + cRD + cWR }
#define third_mapped_object_tpdo2 (0x1A0103 + UINT32 + cRD + cWR }
#define fourth_mapped_object_tpdo2 (0x1A0104 + UINT32 + cRD + cWR }
#define transmit_pdo_mapping_tpdo3 (0x1A0200 + UINT8 + cRD + cWR }
#define first_mapped_object_tpdo3 (0x1A0201 + UINT32 + cRD + cWR }
#define second_mapped_object_tpdo3 (0x1A0202 + UINT32 + cRD + cWR }
#define third_mapped_object_tpdo3 (0x1A0203 + UINT32 + cRD + cWR }
#define fourth_mapped_object_tpdo3 (0x1A0204 + UINT32 + cRD + cWR }
#define transmit_pdo_mapping_tpdo4 (0x1A0300 + UINT8 + cRD + cWR }
#define first_mapped_object_tpdo4 (0x1A0301 + UINT32 + cRD + cWR }
#define second_mapped_object_tpdo4 (0x1A0302 + UINT32 + cRD + cWR }
#define third_mapped_object_tpdo4 (0x1A0303 + UINT32 + cRD + cWR }
#define fourth_mapped_object_tpdo4 (0x1A0304 + UINT32 + cRD + cWR }
#define manufacturer_statuswords (0x200000 + UINT8 + cRD }
#define manufacturer_statusword_1 (0x200001 + UINT32 + cRD + cPDO }
#define manufacturer_status_masks (0x200500 + UINT8 + cRD }

```

```

#define manufacturer_status_mask_1      (0x200501 + UINT32 + cRD + cWR + cPDO }
#define manufacturer_status_invert     (0x200A00 + UINT8 + cRD }
#define manufacturer_status_invert_1   (0x200A01 + UINT32 + cRD + cWR + cPDO }
#define last_warning_code               (0x200F00 + UINT16 + cRD + cPDO }
#define tpdo1_transmit_mask             (0x201400 + UINT8 + cRD }
#define tpdo1_transmit_mask_low         (0x201401 + UINT32 + cRD + cWR }
#define tpdo1_transmit_mask_high        (0x201402 + UINT32 + cRD + cWR }
#define tpdo2_transmit_mask             (0x201500 + UINT8 + cRD }
#define tpdo2_transmit_mask_low         (0x201501 + UINT32 + cRD + cWR }
#define tpdo2_transmit_mask_high        (0x201502 + UINT32 + cRD + cWR }
#define tpdo3_transmit_mask             (0x201600 + UINT8 + cRD }
#define tpdo3_transmit_mask_low         (0x201601 + UINT32 + cRD + cWR }
#define tpdo3_transmit_mask_high        (0x201602 + UINT32 + cRD + cWR }
#define tpdo4_transmit_mask             (0x201700 + UINT8 + cRD }
#define tpdo4_transmit_mask_low         (0x201701 + UINT32 + cRD + cWR }
#define tpdo4_transmit_mask_high        (0x201702 + UINT32 + cRD + cWR }
#define encoder_emulation_data         (0x201A00 + UINT8 + cRD }
#define encoder_emulation_resolution    (0x201A01 + INT32 + cRD + cWR }
#define encoder_emulation_offset       (0x201A02 + INT16 + cRD + cWR }
#define commutation_encoder_select     (0x201F00 + INT16 + cRD + cWR }
#define position_controller_resolution  (0x202000 + UINT32 + cRD + cWR }
#define position_encoder_selection     (0x202100 + INT16 + cRD + cWR }
#define synchronisation_encoder_selection (0x202200 + INT16 + cRD + cWR }
#define synchronisation_filter_time     (0x202300 + UINT32 + cRD + cWR }
#define encoder_x2a_data_field          (0x202400 + UINT8 + cRD }
#define encoder_x2a_resolution          (0x202401 + UINT32 + cRD }
#define encoder_x2a_numerator           (0x202402 + INT16 + cRD + cWR }
#define encoder_x2a_divisor             (0x202403 + INT16 + cRD + cWR }
#define encoder_x10_data_field          (0x202500 + UINT8 + cRD }
#define encoder_x10_resolution          (0x202501 + UINT32 + cRD + cWR }
#define encoder_x10_numerator           (0x202502 + INT16 + cRD + cWR }
#define encoder_x10_divisor             (0x202503 + INT16 + cRD + cWR }
#define encoder_x10_counter             (0x202504 + UINT32 + cRD + cPDO }
#define encoder_x2b_data_field          (0x202600 + UINT8 + cRD }
#define encoder_x2b_resolution          (0x202601 + UINT32 + cRD + cWR }
#define encoder_x2b_numerator           (0x202602 + INT16 + cRD + cWR }
#define encoder_x2b_divisor             (0x202603 + INT16 + cRD + cWR }
#define encoder_x2b_counter             (0x202604 + UINT32 + cRD + cPDO }
#define encoder_emulation_resolution    (0x202800 + INT32 + cRD + cWR }
#define position_demand_sync_value     (0x202D00 + INT32 + cRD }
#define velocity_demand_sync_value     (0x202E00 + INT32 + cRD }
#define synchronisation_selector_data   (0x202F00 + UINT8 + cRD }
#define synchronisation_main            (0x202F07 + UINT16 + cRD + cWR }
#define set_position_absolute           (0x203000 + INT32 + cWR }
#define torque_feed_forward             (0x203A00 + UINT32 + cRD + cWR }
#define homing_timeout                  (0x204500 + UINT16 + cRD + cWR }
#define sample_data                     (0x204A00 + UINT8 + cRD }
#define sample_mode                     (0x204A01 + UINT16 + cRD + cWR }
#define sample_status                   (0x204A02 + UINT8 + cRD + cPDO }
#define sample_status_mask              (0x204A03 + UINT8 + cRD + cWR + cPDO }
#define sample_control                  (0x204A04 + UINT8 + cWR + cPDO }
#define sample_position_rising_edge     (0x204A05 + INT32 + cRD + cPDO }
#define sample_position_falling_edge    (0x204A06 + INT32 + cRD + cPDO }
#define velocity_display_filter_time    (0x207300 + UINT32 + cRD + cWR }
#define velocity_actual_value_filtered  (0x207400 + INT32 + cRD + cPDO }
#define velocity_message                 (0x207800 + UINT8 + cRD }
#define message_target_velocity         (0x207801 + INT32 + cRD + cWR }
#define message_velocity_window         (0x207802 + INT16 + cRD + cWR }
#define velocity_ramps                   (0x209000 + UINT8 + cRD }
#define velocity_rampe_enable           (0x209001 + UINT8 + cRD + cWR }
#define velocity_acceleration_pos       (0x209002 + INT32 + cRD + cWR }
#define velocity_deceleration_pos       (0x209003 + INT32 + cRD + cWR }
#define velocity_acceleration_neg       (0x209004 + INT32 + cRD + cWR }
#define velocity_deceleration_neg       (0x209005 + INT32 + cRD + cWR }
#define error_management                (0x210000 + UINT8 + cRD }
#define error_number                     (0x210001 + UINT8 + cRD + cWR }
#define error_reaction_code             (0x210002 + UINT8 + cRD + cWR }
#define read_write_ko_nr                (0x220000 + UINT32 + cRD + cWR }
#define read_ko                          (0x220400 + UINT32 + cRD }
#define write_ko                         (0x221400 + UINT32 + cWR }
#define read_ko_record                  (0x221500 + UINT8 + cRD }
#define read_ko_demand_value            (0x221501 + UINT32 + cRD }
#define read_ko_actual_value            (0x221502 + UINT32 + cRD }
#define read_ko_minimum                 (0x221503 + UINT32 + cRD }
#define read_ko_maximum                 (0x221504 + UINT32 + cRD }
#define analog_input_voltage            (0x240000 + UINT8 + cRD }
#define analog_input_voltage_ch_0      (0x240001 + INT16 + cRD }
#define analog_input_voltage_ch_1      (0x240002 + INT16 + cRD }

```

```

#define analog_input_voltage_ch_2          (0x240003 + INT16 + cRD
#define analog_input_offset                (0x240100 + UINT8 + cRD
#define analog_input_offset_ch_0          (0x240101 + INT32 + cRD + cWR
#define analog_input_offset_ch_1          (0x240102 + INT32 + cRD + cWR
#define analog_input_offset_ch_2          (0x240103 + INT32 + cRD + cWR
#define current_limitation                 (0x241500 + UINT8 + cRD
#define limit_current_input_channel        (0x241501 + INT8 + cRD + cWR
#define limit_current                     (0x241502 + INT32 + cRD + cWR
#define speed_limitation                   (0x241600 + UINT8 + cRD
#define limit_speed_input_channel          (0x241601 + INT8 + cRD + cWR
#define limit_speed                       (0x241602 + INT32 + cRD + cWR
#define digital_outputs_state_mapping      (0x242000 + UINT8 + cRD
#define dig_out_state_mapp_dout_1         (0x242001 + UINT8 + cRD + cWR
#define dig_out_state_mapp_dout_2         (0x242002 + UINT8 + cRD + cWR
#define dig_out_state_mapp_dout_3         (0x242003 + UINT8 + cRD + cWR
#define dig_out_state_mapp_ea88_0_low     (0x242011 + UINT32 + cRD + cWR
#define dig_out_state_mapp_ea88_0_high    (0x242012 + UINT32 + cRD + cWR
#define digital_inputs_low_byte           (0x2C0A00 + UINT8 + cRD + cPDO
#define error_code                        (0x603F00 + UINT16 + cRD + cPDO
#define controlword                       (0x604000 + UINT16 + cRD + cWR + cPDO
#define statusword                        (0x604100 + UINT16 + cRD + cPDO
#define pole_number                       (0x604D00 + UINT8 + cRD + cWR + cPDO
#define quick_stop_option_code            (0x605A00 + INT16 + cRD + cWR
#define shutdown_option_code              (0x605B00 + INT16 + cRD + cWR
#define disable_operation_option_code     (0x605C00 + INT16 + cRD + cWR
#define stop_option_code                  (0x605D00 + INT16 + cRD + cWR
#define fault_reaction_option_code        (0x605E00 + INT16 + cRD + cWR
#define modes_of_operation                 (0x606000 + INT8 + cRD + cWR + cPDO
#define modes_of_operation_display        (0x606100 + INT8 + cRD + cPDO
#define position_demand_value             (0x606200 + INT32 + cRD + cPDO
#define position_actual_value*            (0x606300 + INT32 + cRD + cPDO
#define position_actual_value             (0x606400 + INT32 + cRD + cPDO
#define following_error_window            (0x606500 + UINT32 + cRD + cWR + cPDO
#define following_error_time_out          (0x606600 + UINT16 + cRD + cWR + cPDO
#define position_window                   (0x606700 + UINT32 + cRD + cWR + cPDO
#define position_window_time              (0x606800 + UINT16 + cRD + cWR + cPDO
#define velocity_sensor_actual_value      (0x606900 + INT32 + cRD + cPDO
#define sensor_selection_code             (0x606A00 + INT16 + cRD + cWR + cPDO
#define velocity_demand_value             (0x606B00 + INT32 + cRD + cPDO
#define velocity_actual_value             (0x606C00 + INT32 + cRD + cPDO
#define velocity_window                   (0x606D00 + UINT16 + cRD + cWR + cPDO
#define velocity_window_time              (0x606E00 + UINT16 + cRD + cWR + cPDO
#define velocity_threshold                 (0x606F00 + UINT16 + cRD + cWR + cPDO
#define velocity_threshold_time           (0x607000 + UINT16 + cRD + cWR + cPDO
#define target_torque                     (0x607100 + INT16 + cRD + cWR + cPDO
#define max_torque                        (0x607200 + UINT16 + cRD + cWR + cPDO
#define max_current                       (0x607300 + UINT16 + cRD + cWR + cPDO
#define torque_demand_value               (0x607400 + INT16 + cRD + cPDO
#define motor_rated_current               (0x607500 + UINT32 + cRD + cWR + cPDO
#define motor_rated_torque                (0x607600 + UINT32 + cRD + cWR + cPDO
#define torque_actual_value               (0x607700 + INT16 + cRD + cPDO
#define current_actual_value              (0x607800 + INT16 + cRD + cPDO
#define dc_link_circuit_voltage           (0x607900 + UINT32 + cRD + cPDO
#define target_position                   (0x607A00 + INT32 + cRD + cWR + cPDO
#define position_range_limit              (0x607B00 + UINT8 + cRD
#define min_position_range_limit          (0x607B01 + INT32 + cRD + cWR + cPDO
#define max_position_range_limit          (0x607B02 + INT32 + cRD + cWR + cPDO
#define home_offset                       (0x607C00 + INT32 + cRD + cWR + cPDO
#define software_position_limit           (0x607D00 + UINT8 + cRD
#define min_position_limit                 (0x607D01 + INT32 + cRD + cWR + cPDO
#define max_position_limit                 (0x607D02 + INT32 + cRD + cWR + cPDO
#define polarity                           (0x607E00 + UINT8 + cRD + cWR + cPDO
#define max_motor_speed                   (0x608000 + UINT16 + cRD + cWR + cPDO
#define profile_velocity                   (0x608100 + UINT32 + cRD + cWR + cPDO
#define end_velocity                       (0x608200 + UINT32 + cRD + cWR + cPDO
#define profile_acceleration               (0x608300 + UINT32 + cRD + cWR + cPDO
#define profile_deceleration               (0x608400 + UINT32 + cRD + cWR + cPDO
#define quick_stop_deceleration           (0x608500 + UINT32 + cRD + cWR + cPDO
#define motion_profile_type                (0x608600 + INT16 + cRD + cWR + cPDO
#define torque_slope                       (0x608700 + UINT32 + cRD + cWR + cPDO
#define torque_profile_type                (0x608800 + INT16 + cRD + cWR + cPDO
#define position_notation_index           (0x608900 + INT8 + cRD + cWR + cPDO
#define position_dimension_index           (0x608A00 + UINT8 + cRD + cWR + cPDO
#define velocity_notation_index           (0x608B00 + INT8 + cRD + cWR + cPDO
#define velocity_dimension_index           (0x608C00 + UINT8 + cRD + cWR + cPDO
#define acceleration_notation_index        (0x608D00 + INT8 + cRD + cWR + cPDO
#define acceleration_dimension_index       (0x608E00 + UINT8 + cRD + cWR + cPDO
#define position_encoder_resolution        (0x608F00 + UINT8 + cRD

```

```

#define encoder_increments          (0x608F01 + UINT32 + cRD + cWR + cPDO }
#define motor_revolutions           (0x608F02 + UINT32 + cRD + cWR + cPDO }
#define velocity_encoder_resolution (0x609000 + UINT8 + cRD }
#define encoder_increments_per_second (0x609001 + UINT32 + cRD + cWR + cPDO }
#define motor_revolutions_per_second (0x609002 + UINT32 + cRD + cWR + cPDO }
#define gear_ratio                   (0x609100 + UINT8 + cRD }
#define motor_revolutions           (0x609101 + UINT32 + cRD + cWR + cPDO }
#define shaft_revolutions           (0x609102 + UINT32 + cRD + cWR + cPDO }
#define feed_constant                (0x609200 + UINT8 + cRD }
#define feed                         (0x609201 + UINT32 + cRD + cWR + cPDO }
#define shaft_revolutions           (0x609202 + UINT32 + cRD + cWR + cPDO }
#define position_factor              (0x609300 + UINT8 + cRD }
#define numerator                    (0x609301 + UINT32 + cRD + cWR + cPDO }
#define divisor                      (0x609302 + UINT32 + cRD + cWR + cPDO }
#define velocity_encoder_factor      (0x609400 + UINT8 + cRD }
#define numerator                    (0x609401 + UINT32 + cRD + cWR + cPDO }
#define divisor                      (0x609402 + UINT32 + cRD + cWR + cPDO }
#define velocity_factor_1            (0x609500 + UINT8 + cRD }
#define numerator                    (0x609501 + UINT32 + cRD + cWR + cPDO }
#define divisor                      (0x609502 + UINT32 + cRD + cWR + cPDO }
#define velocity_factor_2            (0x609600 + UINT8 + cRD }
#define numerator                    (0x609601 + UINT32 + cRD + cWR + cPDO }
#define divisor                      (0x609602 + UINT32 + cRD + cWR + cPDO }
#define acceleration_factor          (0x609700 + UINT8 + cRD }
#define numerator                    (0x609701 + UINT32 + cRD + cWR + cPDO }
#define divisor                      (0x609702 + UINT32 + cRD + cWR + cPDO }
#define homing_method                (0x609800 + INT8 + cRD + cWR + cPDO }
#define homing_speeds                (0x609900 + UINT8 + cRD }
#define speed_during_search_for_switch (0x609901 + UINT32 + cRD + cWR + cPDO }
#define speed_during_search_for_zero (0x609902 + UINT32 + cRD + cWR + cPDO }
#define homing_acceleration          (0x609A00 + UINT32 + cRD + cWR + cPDO }
#define interpolation_submode_select  (0x60C000 + INT16 + cRD + cWR + cPDO }
#define interpolation_data_record     (0x60C100 + UINT8 + cRD }
#define ip_data_position             (0x60C101 + INT32 + cRD + cWR + cPDO }
#define ip_data_controlword          (0x60C102 + UINT8 + cRD + cWR + cPDO }
#define interpolation_time_period     (0x60C200 + UINT8 + cRD }
#define ip_time_units                (0x60C201 + UINT8 + cRD + cWR + cPDO }
#define ip_time_index                (0x60C202 + INT8 + cRD + cWR + cPDO }
#define interpolation_sync_definition (0x60C300 + UINT8 + cRD }
#define synchronize_on_group         (0x60C301 + UINT8 + cRD + cWR + cPDO }
#define ip_sync_every_n_event        (0x60C302 + UINT8 + cRD + cWR + cPDO }
#define interpolation_data_configuration (0x60C400 + UINT8 + cRD + cWR + cPDO }
#define max_buffer_size              (0x60C401 + UINT32 + cRD }
#define actual_size                  (0x60C402 + UINT32 + cRD + cWR + cPDO }
#define buffer_organisation          (0x60C403 + UINT8 + cRD + cWR + cPDO }
#define buffer_position              (0x60C404 + UINT16 + cRD + cWR + cPDO }
#define size_of_data_record          (0x60C405 + UINT8 + cWR + cPDO }
#define buffer_clear                  (0x60C406 + UINT8 + cWR + cPDO }
#define torque_control_parameters    (0x60F600 + UINT8 + cRD }
#define torque_control_gain          (0x60F601 + UINT16 + cRD + cWR }
#define torque_control_time           (0x60F602 + UINT16 + cRD + cWR }
#define velocity_control_parameter_set (0x60F900 + UINT8 + cRD }
#define velocity_control_gain        (0x60F901 + UINT16 + cRD + cWR }
#define velocity_control_time        (0x60F902 + UINT16 + cRD + cWR }
#define velocity_control_filter_time (0x60F904 + UINT16 + cRD + cWR }
#define control_effort                (0x60FA00 + INT32 + cRD + cPDO }
#define position_control_parameter_set (0x60FB00 + UINT8 + cRD }
#define position_control_gain        (0x60FB01 + UINT16 + cRD + cWR }
#define position_control_time        (0x60FB02 + UINT16 + cRD + cWR }
#define position_control_v_max       (0x60FB04 + UINT32 + cRD + cWR }
#define position_error_tolerance_window (0x60FB05 + UINT32 + cRD + cWR }
#define digital_inputs                (0x60FD00 + UINT32 + cRD + cPDO }
#define digital_outputs              (0x60FE00 + UINT8 + cRD }
#define digital_outputs_data          (0x60FE01 + UINT32 + cRD + cWR + cPDO }
#define digital_outputs_mask         (0x60FE02 + UINT32 + cRD + cWR + cPDO }
#define target_velocity              (0x60FF00 + INT32 + cRD + cWR + cPDO }
#define motor_type                    (0x640200 + UINT16 + cRD + cPDO }
#define motor_data                   (0x641000 + UINT8 + cRD }
#define iit_time_motor                (0x641003 + UINT16 + cRD + cWR }
#define iit_ratio_motor              (0x641004 + UINT16 + cRD }
#define phase_order                  (0x641010 + UINT16 + cRD + cWR }
#define encoder_offset_angle         (0x641011 + INT16 + cRD + cWR + cPDO }
#define motor_temperature_sensor_polarity (0x641014 + INT16 + cRD + cWR + cPDO }
#define supported_drive_modes        (0x650200 + UINT32 + cRD + cPDO }
#define drive_data                    (0x651000 + UINT8 + cRD }
#define serial_number                 (0x651001 + UINT32 + cRD }
#define drive_code                    (0x651002 + UINT32 + cRD }
#define user_variable_not_saved      (0x651003 + INT16 + cRD + cWR }

```

```

#define user_variable_saved          (0x651004 + INT16 + cRD + cWR }
#define enable_logic                 (0x651010 + UINT16 + cRD + cWR }
#define limit_switch_polarity        (0x651011 + INT16 + cRD + cWR }
#define limit_switch_selector         (0x651012 + INT16 + cRD + cWR }
#define homing_switch_selector        (0x651013 + INT16 + cRD + cWR }
#define homing_switch_polarity        (0x651014 + INT16 + cRD + cWR }
#define limit_switch_deceleration     (0x651015 + INT32 + cRD + cWR }
#define brake_delay_time              (0x651018 + UINT16 + cRD + cWR }
#define automatic_brake_delay         (0x651019 + UINT16 + cRD + cWR }
#define position_range_limit_enable   (0x651020 + UINT16 + cRD + cWR }
#define position_error_switch_off_limit (0x651022 + UINT32 + cRD + cWR }
#define motor_temperature              (0x65102E + INT16 + cRD + cWR } + cPDO }
#define max_motor_temperature         (0x65102F + INT16 + cRD + cWR }
#define pwm_frequency                 (0x651030 + UINT16 + cRD + cWR }
#define power_stage_temperature       (0x651031 + INT16 + cRD + cWR } + cPDO }
#define max_power_stage_temperature   (0x651032 + INT16 + cRD }
#define nominal_dc_link_circuit_voltage (0x651033 + UINT32 + cRD }
#define actual_dc_link_circuit_voltage (0x651034 + UINT32 + cRD + cWR } + cPDO }
#define max_dc_link_circuit_voltage   (0x651035 + UINT32 + cRD }
#define min_dc_link_circuit_voltage    (0x651036 + UINT32 + cRD + cWR }
#define enable_dc_link_undervoltage_error (0x651037 + UINT16 + cRD + cWR }
#define iit_error_enable               (0x651038 + UINT16 + cRD + cWR }
#define enable_enhanced_modulation    (0x65103A + UINT16 + cRD + cWR }
#define iit_ratio_servo                (0x65103D + UINT16 + cRD + cWR } + cPDO }
#define nominal_current                (0x651040 + UINT32 + cRD }
#define peak_current                   (0x651041 + UINT32 + cRD }
#define drive_serial_number            (0x6510A0 + UINT32 + cRD }
#define drive_type                     (0x6510A1 + UINT32 + cRD }
#define drive_revision                 (0x6510A2 + UINT32 + cRD }
#define encoder_serial_number          (0x6510A3 + UINT32 + cRD }
#define encoder_type                   (0x6510A4 + UINT32 + cRD }
#define encoder_revision               (0x6510A5 + UINT32 + cRD }
#define module_serial_number           (0x6510A6 + UINT32 + cRD }
#define module_type                    (0x6510A7 + UINT32 + cRD }
#define module_revision                (0x6510A8 + UINT32 + cRD }
#define firmware_main_version          (0x6510A9 + UINT32 + cRD }
#define firmware_custom_version        (0x6510AA + UINT32 + cRD }
#define mdc_version                    (0x6510AB + UINT32 + cRD }
#define firmware_type                  (0x6510AC + UINT32 + cRD }
#define km_release                     (0x6510AD + UINT32 + cRD }
#define cycletime_current_controller    (0x6510B0 + UINT32 + cRD }
#define cycletime_velocity_controller   (0x6510B1 + UINT32 + cRD }
#define cycletime_position_controller   (0x6510B2 + UINT32 + cRD }
#define cycletime_trajectory_generator  (0x6510B3 + UINT32 + cRD }
#define device_current                 (0x6510B8 + UINT32 + cRD }
#define commissioning_state            (0x6510C0 + UINT32 + cRD + cWR }
#define compatibility_control           (0x6510F0 + UINT16 + cRD + cWR }

/*****
/* 'magic' word for loading parameter */
*****/
#define cLOAD 0x64616F6C
#define cSAVE 0x65766173

/*****
/* modes of operation */
*****/
#define cPositionMode 0x01
#define cVelocityMode 0x03
#define cTorqueMode 0x04
#define cHomingMode 0x06
#define cInterpolatedMode 0x07
#define cUnknownMode 0xFF

/*****
/* digital inputs */
*****/
#define cEND0 0x00000001 /* negativer Endschalter */
#define cEND1 0x00000002 /* positiver Endschalter */
#define cHOME_SAMPLE 0x00000004 /* Home / Sample */
#define cLOCK 0x00000008 /* Regler- oder Endstufenfreigabe fehlt */
#define cPOS0 0x01000000 /* Digital Input Target selector */
#define cPOS1 0x02000000 /* Digital Input Target selector */
#define cPOS2 0x04000000 /* Digital Input Target selector */
#define cPOS3 0x08000000 /* Digital Input Target selector */
#define cSTART 0x10000000
#define cSAMPLE 0x20000000

```

```

/* ----- Definition nach Steckerbenamung ----- */
#define cDIN0          cPOS0
#define cDIN1          cPOS1
#define cDIN2          cPOS2
#define cDIN3          cPOS3
#define cDIN5          cLOCK
#define cDIN6          cEND0
#define cDIN7          cEND1
#define cDIN8          cSTART
#define cDIN9          cSAMPLE

/*****
/*      controlword
*****/
#define cwSHUT_DOWN          0x0006
#define cwSWITCH_ON          0x0007
#define cwDISABLE_VOLTAGE   0x0000
#define cwQUICK_STOP         0x0002
#define cwDISABLE_OPERATION 0x0007
#define cwENABLE_OPERATION   0x000F
#define cwFAULT_RESET       0x0080 /* Fault Reset mit steigender Flanke */

#define cwNEW_SET_POINT      0x0010
#define cwSTART_HOMING_OPERATION 0x0010
#define cwENABLE_IP_MODE     0x0010
#define cwCHANGE_SET_IMMEDIATELY 0x0020
#define cwABSOLUTE_RELATIV    0x0040
#define cwHOLD                0x0100

/*****
/*      statusword
*****/
/* state definition */
#define cdNOT_READY_TO_SWITCH_ON          0x0000
#define cdSWITCHED_ON_DISABLED           0x0040
#define cdREADY_TO_SWITCH_ON             0x0021
#define cdSWITCHED_ON                    0x0023
#define cdOPERATION_ENABLED              0x0027
#define cdFAULT                          0x000F
#define cdFAULT_REACTION_ACTIVE          0x000F
#define cdQUICK_STOP_ACTIVE              0x0007

/* Bits of status word */
#define swVOLTAGE_DISABLED                0x0010
#define swSWITCH_ON_DISABLED              0x0040
#define swWARNING                         0x0080
#define swREMOTE                          0x0200
#define swTARGET_REACHED                  0x0400
#define swINTERNAL_LIMIT_ACTIVE           0x0800
#define swSET_POINT_ACKNOWLEDGE           0x1000
#define swSPEED0                          0x1000
#define swHOMING_ATTAINED                 0x1000
#define swIP_MODE_ACTIVE                  0x1000
#define swFOLLOWING_ERROR                 0x2000
#define swHOMING_ERROR                    0x2000

/* position modes */
#define pCONTINUOUS                       0x0000
#define pIMMEDIATE                        0x0020
#define pABSOLUTE                          0x0000
#define pRELATIVE                          0x0040

/* motion profile types */
#define mpLINEAR                           0

```

10 Stichwortverzeichnis

7

7-Segment-Anzeige	
'A' in der	150

A

A in 7-Segment-Anzeige	150
acceleration_factor	74
actual_dc_link_circuit_voltage	82
actual_size	203
Aktuelle Zwischenkreisspannung	82
analog_input_offset	129
analog_input_offset_ch_0	129
analog_input_offset_ch_1	129
analog_input_offset_ch_2	129
analog_input_voltage	128
analog_input_voltage_ch_0	128
analog_input_voltage_ch_1	128
analog_input_voltage_ch_2	129
Analoge Eingänge	128
Eingangsspannung Kanal 0	128
Eingangsspannung Kanal 1	128
Eingangsspannung Kanal 2	129
Eingangsspannungen	128
Offsetspannung Kanal 0	129
Offsetspannung Kanal 1	129
Offsetspannung Kanal 2	129
Offsetspannungen	129
Anschlag	188, 189
Anschlussbelegung	26
Antrieb referenziert	168
Anzahl gemappter Objekte	40
Auswahl der Istwert Lage	124
Auswahl der Synchronisationsquelle	125

B

Beschleunigung	
bei der Referenzfahrt	183
beim Positionieren	194
Brems- (Positionieren)	194
Schnellstop-(Positionieren)	195
Betriebsart	177, 178
Ändern der	177
Drehzahlregelung	208
Einstellen der	176
Lesen der	178
Momentenregeln	220
Positionieren	191
Referenzfahrt	179
brake_delay_time	143
Bremse	
Verzögerungszeit	143
Bremsverzögerungszeit	143
buffer_clear	204
buffer_organisation	203
buffer_position	204

C

CAN-Interface	
Anschlußbelegung	26
Kenndaten des	227
cob_id_sync	45
cob_id_used_by_pdo	39
commissioning_state	150
commutation_encoder_select	123
commutation_valid	168
compatibility_control	65
control_effort	105
controlword	159
Bitbelegung	159
Kommandos	160
Objektbeschreibung	159

Controlword für Interpolationsdaten.....	200
current_actual_value	224
current_limitation	111
cycletime_current_controller.....	148
cycletime_position_controller.....	149
cycletime_tracectory_generator.....	149
cycletime_velocity_controller	149

D

dc_link_circuit_voltage.....	225
Default-Parameter laden.....	63
Device Control	154
dig_out_state_mapp_dout_1	133
dig_out_state_mapp_dout_2	133
dig_out_state_mapp_dout_3	133
dig_out_state_mapp_ea88_0_high.....	134
dig_out_state_mapp_ea88_0_low.....	134
digital_inputs.....	131
digital_outputs	132
digital_outputs_data.....	132
digital_outputs_mask.....	132
digital_outputs_state_mapping	133
Digitale Ausgänge	
Mapping von DOUT1	133
Mapping von DOUT1...DOUT4 von EA88_0 ..	134
Mapping von DOUT2	133
Mapping von DOUT3	133
Mapping von DOUT5...DOUT8 von EA88_0 ..	134
Digitale Eingänge	132
Mapping	133
Maske	132
Zustände.....	132
Digitale Eingänge	131
disable_operation_option_code.....	174
divisor	
acceleration_factor.....	74
position_factor	70
velocity_encoder_factor	72
Drehzahlanzeige	
Filter.....	97
Drehzahlanzeige, gefiltert.....	214
Drehzahlbegrenzter Momentenbetrieb	113
Drehzahlbegrenzung	113

Quelle	113
Skalierung.....	113
Sollwert.....	113
Drehzahl-Istwert	213
Drehzahlregelung	208
Drehzahl-Sollwert	212
Geschwindigkeitssensor-Auswahl.....	211
Max. Motordrehzahl.....	217
Sollgeschwindigkeit.....	217
Stillstandsschwelle.....	216
Stillstandsschwellenzeit	216
Zielfenster	215
Zielfensterzeit	215
Zielgeschwindigkeit.....	217
Drehzahlregler	95
Filterzeitkonstante.....	96
Parameter	96
Verstärkung	96
Zeitkonstante	96
Drehzahl-Sollwert	212
drive_data.....	78, 90, 92, 107, 109, 135, 143, 146
drive_serial_number	146
drive_type.....	146
Durchdrehschutz	95

E

Eingänge, analoge.....	128
Einstellen der Betriebsart	176
EMERGENCY	47
EMERGENCY-Message.....	47
Aufbau der	46
enable_dc_link_undervoltage_error	83
enable_enhanced_modulation.....	79
enable_logic	78
encoder_emulation_data	121
encoder_emulation_offset	121
encoder_emulation_resolution.....	121
encoder_offset_angle	91
encoder_x10_counter.....	119
encoder_x10_data_field	118
encoder_x10_divisor	118
encoder_x10_numerator	118
encoder_x10_resolution	118

encoder_x2a_data_field	115
encoder_x2a_divisor.....	115
encoder_x2a_numerator.....	115
encoder_x2a_resolution	115
encoder_x2b_counter	117
encoder_x2b_data_field	116
encoder_x2b_divisor.....	117
encoder_x2b_numerator.....	116
encoder_x2b_resolution	116
end_velocity.....	193
Endgeschwindigkeit	193
Endschalter.....	135, 184, 186
Nothalt-Rampe.....	137
Polarität.....	135
Tauschen der.....	136
Endstufenfreigabe	78
Endstufenparameter	78
Freigabelogik	78
Gerätenennspannung	81
Gerätenennstrom.....	84
max. Zwischenkreisspannung.....	82
Maximale Temperatur.....	81
Maximalstrom	85
min. Zwischenkreisspannung.....	83
PWM-Frequenz.....	79
Temperatur	80
Zwischenkreisspannung	82
Endstufen-Temperatur.....	80
Error Control Protocol	
Heartbeat.....	54, 55
Node guarding	57
error_management.....	152
error_number.....	152
error_reaction_code.....	152
Erweiterte Sinusmodulation	79

F

Factor Group	68
acceleration_factor.....	74
polarity	76
position_factor	70
velocity_encoder_factor.....	72
Fahrkurven-Generator	191

Fault	157
Fault Reaction Active.....	157
fault_reaction_option_code.....	175
Fehler	
'A' in 7-Segment-Anzeige	150
Reglerfehler	47
SDO-Fehlermeldungen.....	33
Fehlermanagement	152
Fehlernummer	152
Fehlerreaktion	152
Fehlerregister	47
Filterzeitkonstante Synchrondrehzahl.....	127
firmware_custom_version.....	147
firmware_main_version.....	147
firmware_type	148
first_mapped_object	40
Following_error.....	98
following_error_time_out	105
following_error_window	104
fourth_mapped_object.....	41
Freigabelogik.....	78

G

Gerätenennspannung.....	81
Gerätenennstrom.....	84
Gerätesteuerung.....	154
Gerätetyp.....	146
Geschwindigkeit	
bei der Referenzfahrt	182
beim Positionieren	193
End- (Positionieren).....	193
Geschwindigkeitssensor-Auswahl	211
Grenzwert Schleppfehler	107
guard_time	59

H

Heartbeat.....	54, 55
Herstellercode	144
Herstellerspezifische Statuswort- Invertierung 1.	172
Herstellerspezifische Statuswort-Maske 1	171
Herstellerspezifisches Statuswort 1.....	168

home_offset.....	180
homing mode	
home_offset.....	180
homing_acceleration.....	183
homing_method.....	181
homing_speeds.....	182
Homing Mode.....	179
homing_acceleration.....	183
homing_method.....	181
homing_speeds.....	182
homing_switch_polarity.....	136
homing_switch_selector.....	137
homing_timeout.....	183

I

I ² t-Auslastung.....	89
I ² t-Zeit.....	89
Identifizierung des Geräts.....	144
identity_object.....	144
iit_error_enable.....	90
iit_ratio_motor.....	89
iit_time_motor.....	89
iit-Fehler auslösen.....	90
inhibit_time.....	39
Inkrementalgeberemulation	
Auflösung.....	121
Offset.....	121
interpolation_data_configuration.....	203
interpolation_data_record.....	200
interpolation_submode_select.....	199
interpolation_sync_definition.....	202
interpolation_time_period.....	201
Interpolations-Daten.....	200
Interpolations-Typ.....	199
ip_data_controlword.....	200
ip_data_position.....	200
ip_sync every n event.....	202
ip_time_index.....	201
ip_time_units.....	201
is_referenced.....	168

Istposition setzen.....	110
Istwert	
Lage in position_units (position_actual_value)104	
Moment (torque_actual_value).....	224

K

km_release.....	147
Kommutiergeberselektion.....	123
Kommutierlage gültig.....	168
Korrekturgeschwindigkeit.....	102

L

Lage-Istwert (position units).....	104
Lageregler.....	98
Ausgang des.....	105
Parameter.....	102
Totbereich.....	103
Verstärkung.....	102
Zeitkonstante.....	102
Lagereglerausgang.....	105
Lageregler-Parameter.....	102
Lagereglerverstärkung.....	102
Lagereglerzeitkonstante.....	102
Lagesollwert (position units).....	103
Lagewert Interpolation.....	200
last_warning_code.....	153
Letzte Warnung.....	153
limit_current.....	111, 113
limit_current_input_channel.....	111
limit_speed_input_channel.....	113
limit_switch_deceleration.....	137
limit_switch_polarity.....	135
limit_switch_selector.....	136

M

manufacturer_status_invert.....	172
manufacturer_status_invert_1.....	172
manufacturer_status_mask_1.....	171
manufacturer_status_masks.....	171

manufacturer_statusword_1	168
Bitbelegung	168
manufacturer_statuswords	168
Mappingparameter für PDOs	40
Max. Motor-Temperatur	93
max_buffer_size	203
max_current	88
max_dc_link_circuit_voltage	82
max_motor_speed	217
max_motor_temperature	93
max_position_range_limit	108
max_power_stage_temperature	81
max_torque	222
Maximale Endstufentemperatur	81
Maximale Motordrehzahl	217
Maximale Zwischenkreisspannung	82
Maximales Moment	222
Maximalstrom	85
min_dc_link_circuit_voltage	83
min_position_range_limit	108
Minimale Zwischenkreisspannung	83
modes_of_operation	177
modes_of_operation_display	178
Momentenbegrenzter Drehzahlbetrieb	111
Momentenbegrenzung	111
Quelle	111
Skalierung	112
Sollwert	111
Momenten-Istwert	224
Momentenregeln	220
Momentenregelung	
Max. Moment	222
Momenten-Istwert	224
Nennmoment	223
Sollmoment	222
Sollwertprofil	226
Stromsollwert	223
Zielmoment	222
motion_profile_type	195
motor_data	89, 91
motorRatedCurrent	87
motorRatedTorque	223
motorTemperature	92
motorTemperatureSensorPolarity	92

Motoranpassung	86
MotorNennstrom	87
Motorparameter	
I ² t-Zeit	89
Nennstrom	87
Pol(paar)zahl	88
Resolveroffsetwinkel	91
Spitzenstrom	88
Motorspitzenstrom	88
Motor-Temperatur	92

N

Nennmoment des Motors	223
Nennstrom	
Motor	87
Netzwerkmanagement	52
Neue Position anfahren	196
NMT-Service	52
Nodeguarding	57
guard_time	59
life_time_factor	59
nominal_current	84
nominal_dc_link_circuit_voltage	81
Not Ready to Switch On	157
Nullimpuls	189
Nullpunkt-Offset	180
number_of_mapped_objects	40
numerator	
acceleration_factor	74
position_factor	70
velocity_encoder_factor	72

O

Objekte	
Objekt 100 _h	61
Objekt 1001 _h	61
Objekt 1002 _h	61
Objekt 1003 _h	50
Objekt 1003 _h _01 _h	50
Objekt 1003 _h _02 _h	50
Objekt 1003 _h _03 _h	50

Objekt 1003 _{h_04h}	50	Objekt 200A _h	172
Objekt 1005 _h	45	Objekt 200A _{h_01h}	172
Objekt 1006 _h	61	Objekt 200F _h	153
Objekt 1007 _h	61	Objekt 2014 _h	42
Objekt 100C _h	59	Objekt 2015 _h	42
Objekt 100D _h	59	Objekt 2016 _h	42
Objekt 1010 _h	64	Objekt 2017 _h	42
Objekt 1010 _{h_01h}	64	Objekt 201A _h	121
Objekt 1011 _h	63	Objekt 201A _{h_01h}	121
Objekt 1011 _{h_01h}	63	Objekt 201A _{h_02h}	121
Objekt 1017 _h	56	Objekt 201F _h	123
Objekt 1018 _h	144	Objekt 2021 _h	124
Objekt 1018 _{h_01h}	144	Objekt 2022 _h	125
Objekt 1018 _{h_02h}	145	Objekt 2023 _h	127
Objekt 1018 _{h_03h}	145	Objekt 2024 _h	115
Objekt 1018 _{h_04h}	145	Objekt 2024 _{h_01h}	115
Objekt 1400 _h	43	Objekt 2024 _{h_02h}	115
Objekt 1401 _h	43	Objekt 2024 _{h_03h}	115
Objekt 1402 _h	43	Objekt 2025 _h	118
Objekt 1403 _h	43	Objekt 2025 _{h_01h}	118
Objekt 1600 _h	43	Objekt 2025 _{h_02h}	118
Objekt 1601 _h	43	Objekt 2025 _{h_03h}	118
Objekt 1602 _h	43	Objekt 2025 _{h_04h}	119
Objekt 1603 _h	43	Objekt 2026 _h	116
Objekt 1800 _h	39, 41	Objekt 2026 _{h_01h}	116
Objekt 1800 _{h_01h}	39	Objekt 2026 _{h_02h}	116
Objekt 1800 _{h_02h}	39	Objekt 2026 _{h_03h}	117
Objekt 1800 _{h_03h}	39	Objekt 2026 _{h_04h}	117
Objekt 1801 _h	41	Objekt 2028 _h	121
Objekt 1802 _h	41	Objekt 202D _h	103
Objekt 1803 _h	42	Objekt 202E _h	212
Objekt 1A00 _h	40, 41	Objekt 202F _h	126
Objekt 1A00 _{h_00h}	40	Objekt 202F _{h_07h}	126
Objekt 1A00 _{h_01h}	40	Objekt 2030 _h	110
Objekt 1A00 _{h_02h}	40	Objekt 2045 _h	183
Objekt 1A00 _{h_03h}	40	Objekt 204A _h	139
Objekt 1A00 _{h_04h}	41	Objekt 204A _{h_01h}	139
Objekt 1A01 _h	41	Objekt 204A _{h_02h}	139
Objekt 1A02 _h	41	Objekt 204A _{h_03h}	140
Objekt 1A03 _h	42	Objekt 204A _{h_04h}	140
Objekt 2000 _h	168	Objekt 204A _{h_05h}	141
Objekt 2000 _{h_01h}	168	Objekt 204A _{h_06h}	141
Objekt 2005 _h	171	Objekt 2073 _h	97
Objekt 2005 _{h_01h}	171	Objekt 2074 _h	214

Objekt 2090 _h	218	Objekt 6069 _h	211
Objekt 2090 _{h_01h}	219	Objekt 606A _h	211
Objekt 2090 _{h_02h}	219	Objekt 606B _h	212
Objekt 2090 _{h_03h}	219	Objekt 606C _h	213
Objekt 2090 _{h_04h}	219	Objekt 606D _h	215
Objekt 2090 _{h_05h}	219	Objekt 606E _h	215
Objekt 2100 _h	152	Objekt 606F _h	216
Objekt 2100 _{h_01h}	152	Objekt 6070 _h	216
Objekt 2100 _{h_02h}	152	Objekt 6071 _h	222
Objekt 2400 _h	128	Objekt 6072 _h	222
Objekt 2400 _{h_01h}	128	Objekt 6073 _h	88
Objekt 2400 _{h_02h}	128	Objekt 6074 _h	223
Objekt 2400 _{h_03h}	129	Objekt 6075 _h	87
Objekt 2401 _h	129	Objekt 6076 _h	223
Objekt 2401 _{h_01h}	129	Objekt 6077 _h	224
Objekt 2401 _{h_02h}	129	Objekt 6078 _h	224
Objekt 2401 _{h_03h}	129	Objekt 6079 _h	225
Objekt 2415 _h	111	Objekt 607A _h	192
Objekt 2415 _{h_01h}	111	Objekt 607B _h	108
Objekt 2415 _{h_02h}	111	Objekt 607B _{h_01h}	108
Objekt 2416 _h	113	Objekt 607B _{h_02h}	108
Objekt 2416 _{h_01h}	113	Objekt 607C _h	180
Objekt 2416 _{h_02h}	113	Objekt 607E _h	76
Objekt 2420 _h	133	Objekt 6080 _h	217
Objekt 2420 _{h_01h}	133	Objekt 6081 _h	193
Objekt 2420 _{h_02h}	133	Objekt 6082 _h	193
Objekt 2420 _{h_03h}	133	Objekt 6083 _h	194
Objekt 2420 _{h_11h}	134	Objekt 6084 _h	194
Objekt 2420 _{h_12h}	134	Objekt 6085 _h	195
Objekt 6040 _h	159	Objekt 6086 _h	195
Objekt 6041 _h	164	Objekt 6087 _h	225
Objekt 604D _h	88	Objekt 6088 _h	226
Objekt 605A _h	174	Objekt 6093 _h	70
Objekt 605B _h	173	Objekt 6093 _{h_01h}	70
Objekt 605C _h	174	Objekt 6093 _{h_02h}	70
Objekt 605E _h	175	Objekt 6094 _h	72
Objekt 6060 _h	177	Objekt 6094 _{h_01h}	72
Objekt 6061 _h	178	Objekt 6094 _{h_02h}	72
Objekt 6062 _h	103	Objekt 6097 _h	74
Objekt 6064 _h	104	Objekt 6097 _{h_01h}	74
Objekt 6065 _h	104	Objekt 6097 _{h_02h}	74
Objekt 6066 _h	105	Objekt 6098 _h	181
Objekt 6067 _h	106	Objekt 6099 _h	182
Objekt 6068 _h	106	Objekt 6099 _{h_01h}	182

Objekt 6099 _h _02 _h	182	Objekt 6510 _h .78, 90, 92, 107, 109, 135, 143, 146	
Objekt 609A _h	183	Objekt 6510 _h _10 _h	78
Objekt 60C0 _h	199	Objekt 6510 _h _11 _h	135
Objekt 60C1 _h	200	Objekt 6510 _h _12 _h	136
Objekt 60C1 _h _01 _h	200	Objekt 6510 _h _13 _h	137
Objekt 60C1 _h _02 _h	200	Objekt 6510 _h _14 _h	136
Objekt 60C2 _h	201	Objekt 6510 _h _15 _h	137
Objekt 60C2 _h _01 _h	201	Objekt 6510 _h _18 _h	143
Objekt 60C2 _h _02 _h	201	Objekt 6510 _h _20 _h	109
Objekt 60C3 _h	202	Objekt 6510 _h _22 _h	107
Objekt 60C3 _h _01 _h	202	Objekt 6510 _h _2E _h	92
Objekt 60C3 _h _02 _h	202	Objekt 6510 _h _2F _h	93
Objekt 60C4 _h	203	Objekt 6510 _h _30 _h	79
Objekt 60C4 _h _01 _h	203	Objekt 6510 _h _31 _h	80
Objekt 60C4 _h _02 _h	203	Objekt 6510 _h _32 _h	81
Objekt 60C4 _h _03 _h	203	Objekt 6510 _h _33 _h	81
Objekt 60C4 _h _04 _h	204	Objekt 6510 _h _34 _h	82
Objekt 60C4 _h _05 _h	204	Objekt 6510 _h _35 _h	82
Objekt 60C4 _h _06 _h	204	Objekt 6510 _h _36 _h	83
Objekt 60F6 _h	94	Objekt 6510 _h _37 _h	83
Objekt 60F6 _h _01 _h	94	Objekt 6510 _h _38 _h	90
Objekt 60F6 _h _02 _h	94	Objekt 6510 _h _3A _h	79
Objekt 60F9 _h	96	Objekt 6510 _h _40 _h	84
Objekt 60F9 _h _01 _h	96	Objekt 6510 _h _41 _h	85
Objekt 60F9 _h _02 _h	96	Objekt 6510 _h _A0 _h	146
Objekt 60F9 _h _04 _h	96	Objekt 6510 _h _A1 _h	146
Objekt 60FA _h	105	Objekt 6510 _h _A9 _h	147
Objekt 60FB _h	102	Objekt 6510 _h _AA _h	147
Objekt 60FB _h _01 _h	102	Objekt 6510 _h _AC _h	148
Objekt 60FB _h _02 _h	102	Objekt 6510 _h _AD _h	147
Objekt 60FB _h _04 _h	102	Objekt 6510 _h _B0 _h	148
Objekt 60FB _h _05 _h	103	Objekt 6510 _h _B1 _h	149
Objekt 60FD _h	131	Objekt 6510 _h _B2 _h	149
Objekt 60FE _h	132	Objekt 6510 _h _B3 _h	149
Objekt 60FE _h _01 _h	132	Objekt 6510 _h _C0 _h	150
Objekt 60FE _h _02 _h	132	Objekt 6510 _h _F0 _h	65
Objekt 60FF _h	217	Offset des Winkelgebers	91
Objekt 6410 _h	89, 91	Operation enable	157
Objekt 6410 _h _03 _h	89		
Objekt 6410 _h _04 _h	89		
Objekt 6410 _h _10 _h	90		
Objekt 6410 _h _11 _h	91		
Objekt 6410 _h _11 _h	91		
Objekt 6410 _h _14 _h	92		

P

Parameter einstellen.....	61
Parametersatz sichern.....	64
Parametersätze	

Defaultwerte laden	63	Anzahl eingetragener Objekte	43
Laden und Speichern	61	COB-ID used by PDO	43
Parametersatz sichern	64	first mapped object	43
Parametrierstatus	150	fourth mapped object	43
PDO	30, 35	Identifier	43
1. eingetragenes Objekt	40	number of mapped objects	43
2. eingetragenes Objekt	40	second mapped object	43
3. eingetragenes Objekt	40	third mapped object	43
4. eingetragenes Objekt	41	transmission type	43
RPDO1		Übertragungstyp	43
1. eingetragenes Objekt	43	RPDO4	
2. eingetragenes Objekt	43	1. eingetragenes Objekt	43
3. eingetragenes Objekt	43	2. eingetragenes Objekt	43
4. eingetragenes Objekt	43	3. eingetragenes Objekt	43
Anzahl eingetragener Objekte	43	4. eingetragenes Objekt	43
COB-ID used by PDO	43	Anzahl eingetragener Objekte	43
first mapped object	43	COB-ID used by PDO	43
fourth mapped object	43	first mapped object	43
Identifier	43	fourth mapped object	43
number of mapped objects	43	Identifier	43
second mapped object	43	number of mapped objects	43
third mapped object	43	second mapped object	43
transmission type	43	third mapped object	43
Übertragungstyp	43	transmission type	43
RPDO2		Übertragungstyp	43
1. eingetragenes Objekt	43	TPDO1	
2. eingetragenes Objekt	43	1. eingetragenes Objekt	41
3. eingetragenes Objekt	43	2. eingetragenes Objekt	41
4. eingetragenes Objekt	43	3. eingetragenes Objekt	41
Anzahl eingetragener Objekte	43	4. eingetragenes Objekt	41
COB-ID used by PDO	43	Anzahl eingetragener Objekte	41
first mapped object	43	COB-ID used by PDO	41
fourth mapped object	43	first mapped object	41
Identifier	43	fourth mapped object	41
number of mapped objects	43	Identifier	41
second mapped object	43	inhibit time	41
third mapped object	43	number of mapped objects	41
transmission type	43	second mapped object	41
Übertragungstyp	43	Sperrzeit	41
RPDO3		third mapped object	41
1. eingetragenes Objekt	43	transmission type	41
2. eingetragenes Objekt	43	Übertragungsmaske	42
3. eingetragenes Objekt	43	Übertragungstyp	41
4. eingetragenes Objekt	43	TPDO2	

1. eingetragenes Objekt	41	Identifizier	42
2. eingetragenes Objekt	41	inhibit time	42
3. eingetragenes Objekt	41	number of mapped objects	42
4. eingetragenes Objekt	41	second mapped object	42
Anzahl eingetragener Objekte	41	Sperrzeit	42
COB-ID used by PDO	41	third mapped object	42
first mapped object	41	transmission type	42
fourth mapped object	41	Übertragungsmaske	42
Identifizier	41	Übertragungstyp	42
inhibit time	41	PDO-Message	30, 35
number of mapped objects	41	peak_current	85
second mapped object	41	phase_order	90
Sperrzeit	41	Polarität Motortemperatursensor	92
third mapped object	41	polarity	76
transmission type	41	pole_number	88
Übertragungsmaske	42	Polpaarzahl	88
Übertragungstyp	41	Polzahl	88
TPDO3		position control function	98
1. eingetragenes Objekt	41	Position setzen	110
2. eingetragenes Objekt	41	position_actual_value	104
3. eingetragenes Objekt	41	position_control_gain	102
4. eingetragenes Objekt	41	position_control_parameter_set	102
Anzahl eingetragener Objekte	41	position_control_time	102
COB-ID used by PDO	41	position_control_v_max	102
first mapped object	41	position_demand_sync_value	103
fourth mapped object	41	position_demand_value	103
Identifizier	41	position_encoder_selection	124
inhibit time	41	position_error_switch_off_limit	107
number of mapped objects	41	position_error_tolerance_window	103
second mapped object	41	position_factor	70
Sperrzeit	41	position_range_limit	108
third mapped object	41	position_range_limit_enable	109
transmission type	41	Position_reached	99
Übertragungsmaske	42	position_window	106
Übertragungstyp	41	position_window_time	106
TPDO4		Positionier-Beschleunigung	194
1. eingetragenes Objekt	42	Positionier-Bremsbeschleunigung	194
2. eingetragenes Objekt	42	Positionieren	191, 196
3. eingetragenes Objekt	42	Beschleunigung beim	194
4. eingetragenes Objekt	42	Bremsbeschleunigung	194
Anzahl eingetragener Objekte	42	Endgeschwindigkeit	193
COB-ID used by PDO	42	Geschwindigkeit beim	193
first mapped object	42	Handshake	196
fourth mapped object	42	Schnellstop-Beschleunigung	195

Zielposition.....	192
Positionier-Geschwindigkeit.....	193
Positionierprofil	
Lineares.....	195
Ruckfreies.....	195
Sinus ²	195
Positionierung starten.....	196
Positionswert Interpolation.....	200
power_stage_temperature.....	80
pre_defined_error_field.....	50
producer_heartbeat_time.....	56
product_code.....	145
Produktcode.....	145
Profil Position Mode	
profile_deceleration.....	194
Profile Position Mode.....	191
end_velocity.....	193
motion_profile_type.....	195
profile_acceleration.....	194
profile_velocity.....	193
quick_stop_deceleration.....	195
target_position.....	192
Profile Torque Mode.....	220
current_actual_value.....	224
dc_link_circuit_voltage.....	225
max_torque.....	222
motorRatedTorque.....	223
target_torque.....	222
torque_actual_value.....	224
torque_demand_value.....	223
torque_profile_type.....	226
torque_slope.....	225
Profile Velocity Mode.....	208
max_motor_speed.....	217
sensor_selection_code.....	211
target_velocity.....	217
velocity_actual_value.....	213
velocity_actual_value_filtered.....	214
velocity_demand_value.....	212
velocity_display_filter_time.....	97
velocity_sensor.....	211
velocity_threshold.....	216
velocity_threshold_time.....	216
velocity_window.....	215

velocity_window_time.....	215
profile_acceleration.....	194
profile_deceleration.....	194
profile_velocity.....	193
pwm_frequency.....	79
PWM-Frequenz.....	79

Q

Quick Stop Active.....	157
quick_stop_deceleration.....	195
quick_stop_option_code.....	174

R

Ready to Switch On.....	157
ready_for_enab.....	168
Receive_PDO_1.....	43
Receive_PDO_2.....	43
Receive_PDO_3.....	43
Receive_PDO_4.....	43
Referenzfahrt.....	179
Steuerung der.....	190
Timeout.....	183
Referenzfahrt Methoden.....	184
Referenzfahrten	
Beschleunigung.....	183
Geschwindigkeiten.....	182
Kriechgeschwindigkeit.....	182
Methode.....	181
Nullpunkt-Offset.....	180
Suchgeschwindigkeit.....	182
Referenzfahrt-Methode.....	181
Referenzposition gültig.....	168
Referenzschalter.....	135, 137
Polarität.....	136
Reglerfreigabe.....	78
Freigabe möglich.....	168
Regler-Freigabelogik.....	78
resolver_offset_angle.....	91
Resolveroffsetwinkel.....	91
restore_all_default_parameters.....	63
restore_parameters.....	63

revision_number	145
Revisionsnummer CANopen	145
R-PDO 1	43
R-PDO 2	43
R-PDO 3	43
R-PDO 4	43

S

Sample

Modus	139
Status	139
Statusmaske	140
Steuerung	140
sample_control	140
sample_data	139
sample_mode	139
sample_position_falling_edge.....	141
sample_position_rising_edge	141
sample_status	139
sample_status_mask	140
SAMPLE-Eingang als Referenzschalter	137
Sampling-Position	
Fallende Flanke	141
Steigende Flanke	141
save_all_parameters	64
Schleppfehler.....	98
Definition	98
Fehlerfenster.....	104
Grenzwert- Überschreitung	107
Timeoutzeit	105
Schleppfehlerfenster.....	104
Schleppfehler-Timeoutzeit	105
Schnellstop-Beschleunigung	195
SDO.....	30, 31
Fehlermeldungen	33
SDO-Message	30, 31
second_mapped_object.....	40
sensor_selection_code	211
serial_number	145
Seriennummer des Geräts.....	146
set_position_absolute	110
shutdown_option_code	173
size_of_data_record	204

Skalierungsfaktoren.....	68
Beschleunigungsfaktor	74
Geschwindigkeitsfaktor	72
Positionsfaktor	70
Vorzeichenwahl	76
Sollgeschwindigkeit für Drehzahlregelung	217
Sollmoment (Momentenregelung)	222
Sollwert	
Drehzahl	212
Lage (position units)	103
Moment.....	222
Strom	223
Synchrondrehzahl (velocity units)	212
speed_during_search_for_switch	182
speed_during_search_for_zero	182
speed_limitation.....	113
Spitzenstrom	85
Motor	88
standard_error_field_0	50
standard_error_field_1	50
standard_error_field_2	50
standard_error_field_3	50
START-Eingang als Referenzschalter.....	137
State	
Fault.....	157
Fault Reaction Active.....	157
Not Ready to Switch On.....	157
Operation Enable.....	157
Quick Stop Active	157
Ready to Switch On	157
Switch On Disabled	157
Switched On	157
state diagram.....	155
statemachine	155
statusword	
Bitbelegung.....	164
Objektbeschreibung.....	164
Statuswort	
Herstellerspezifische Invertierung.....	172
Herstellerspezifische Maske	171
Herstellerspezifisches	168
Steuerung des Reglers.....	154
Stillstandschwelle bei Drehzahlregelung	216
Stillstandsschwellenzeit bei Drehzahlregelung ...	216

store_parameters	64
Strombegrenzung	111
Stromregler	86
Parameter	94
Verstärkung	94
Zeitkonstante	94
Stromsollwert	223
Switch On Disabled	157
Switched On	157
SYNC	45
Synchrondrehzahl (velocity units)	212
synchronisation_encoder_selection	125
synchronisation_filter_time	127
synchronisation_main	126
synchronisation_selector_data	126
SYNC-Message	45
synchronize_on_group	202

T

target_position	192
target_torque	221, 222
target_velocity	217
Temperatur	
Max. Motor	93
Motor	92
third_mapped_object	40
torque_actual_value	224
torque_control_gain	94
torque_control_parameters	94
torque_control_time	94
torque_demand_value	223
torque_profile_type	226
torque_slope	225
T-PDO 1	41
T-PDO 2	41
T-PDO 3	41
T-PDO 4	42
tpdo_1_transmit_mask	42
tpdo_2_transmit_mask	42
tpdo_3_transmit_mask	42
tpdo_4_transmit_mask	42
transfer_PDO_1	41
transfer_PDO_2	41

transfer_PDO_3	41
transfer_PDO_4	42
transmission_type	39
transmit_pdo_mapping	40
transmit_pdo_parameter	39
Typ der geladenen Firmware	148

U

Überschreitung Grenzwert Schleppfehler	107
Übertragungsart	39
Übertragungsparameter für PDOs	39
Überwachung der Kommunikation	54, 55, 57
Überwachungszeit Nodeguarding	59
Umrechnungsfaktoren	68
Beschleunigungsfaktor	74
Geschwindigkeitsfaktor	72
Positionsfaktor	70
Vorzeichenwahl	76
Unterspannungsüberwachung aktivieren	83
Unterspannungsüberwachung deaktivieren	83

V

velocity_acceleration_neg	219
velocity_acceleration_pos	219
velocity_actual_value	213
velocity_actual_value_filtered	214
velocity_control_filter_time	96
velocity_control_gain	96
velocity_control_parameter_set	96
velocity_control_time	96
velocity_deceleration_neg	219
velocity_deceleration_pos	219
velocity_demand_sync_value	212
velocity_demand_value	212
velocity_display_filter_time	97
velocity_encoder_factor	72
velocity_rampe_enable	219
velocity_ramps	218
velocity_sensor_actual_value	211
velocity_threshold	216
velocity_threshold_time	216

velocity_window.....	215
velocity_window_time.....	215
vendor_id.....	144
Verhalten bei Kommando 'disable operation'.....	174
Verhalten bei Kommando 'quick stop'.....	174
Verhalten bei Kommando 'shutdown'.....	173
Verkabelungshinweise.....	27
Versionsnummer der Firmware.....	147
Versionsnummer der kundenspez. Variante.....	147
Versionsnummer des KM-Release.....	147
Verstärkung des Stromreglers.....	94

W

Warnungen anzeigen.....	153
Winkelgeberoffset.....	91

X

X10	
Abtrieb.....	118
Antrieb.....	118
Auflösung.....	118
Zähler.....	119
X2A	
Abtrieb.....	115
Antrieb.....	115
Auflösung.....	115
X2B	
Abtrieb.....	117
Antrieb.....	116
Auflösung.....	116
Zähler.....	117

Z

Zeitkonstante des Stromreglers.....	94
-------------------------------------	----

Zielfenster	
Positionsfenster.....	106
Zeit.....	106
Zielfenster bei Drehzahlregelung.....	215
Zielfensterzeit.....	106
Zielfensterzeit bei Drehzahlregelung.....	215
Zielgeschwindigkeit für Drehzahlregelung.....	217
Zielmoment (Momentenregelung).....	222
Zielposition.....	192
Zielpositionsfenster.....	106
Zulässiges Moment.....	222
Zustand	
Fault.....	157
Fault Reaction Active.....	157
Not Ready to Switch On.....	157
Operation Enable.....	157
Quick Stop Active.....	157
Ready to Switch On.....	157
Switch On Disabled.....	157
Switched On.....	157
Zustandsdiagramm des Reglers.....	155
Zwischenkreis	
Überwachung des.....	83
Zwischenkreisspannung	
aktuelle.....	82
maximale.....	82
minimale.....	83
Zwischenkreisüberwachung.....	82, 83
Zykluszeit	
Drehzahlregler.....	149
Lageregler.....	149
Positioniersteuerung.....	149
Stromregler.....	148
Zykluszeit Heartbeat-Telegramme.....	56
Zykluszeit PDOs.....	39