

CiA Draft Standard Proposal DSP-402



Device Profile for

Drives and Motion Control

Version 1.1

Date: October 8, 1998

© CAN in Automation e.V.

not recommended for implementation
may be changed without notification

History

Changes made compared to revision 1.0
(old pages in brackets):

- page 57 (page 56)
Bit 7 *Fault Reset* set to 0 except for the transition *Fault Reset*. See explanation page 54
State Transition 15.
- page 193 (page 183)
Object 6058_h, Subindex 1: *vl_frequency_motor_min_amount*.
- page 24 (page 23)
Object 1000_h: *Device Type*.
Adaption to DS301 for multi profile devices.
Additional information bits 16..23 **bit-encoded**.
- page 62 (page 61)
controlword bit 13, Profile Velocity mode: ***max_slippage_error***. See explanation
page 161.
- page 161 (page 152)
Object 60F8_h:
When the *max_slippage* has been reached, the corresponding bit 13 *max_slippage_error*
in the status message will be set to one.
- page 78, 79 (page 76, 77)
Object 6089_h, Object 608A_h: ***software_position_limit*** added.
- page 124, 125, 126, 128 (page 120, 121, 122, 123)
Object 6062_h, Object 6064_h, Object 6065_h, Object 6067_h
Units changed to *position units*.
- page 100, 101, 102 (page 98, 99, 100)
Object 607F_h, Object 6081_h, Object 6082_h
Value Range changed to 0..($2^{31}-1$).
- page 57 (page 56)
controlword bit 6, Profile Position Mode: defined
0: absolute
1: relative
- page 94 (page 92)
Figure 15: *position_demand_value* is index 60F2_h.
- page 111, 114 (page 108, 111)
Homing Methods: there is a new structure.
Object 6098_h is adapted to the new structure.
- page 120 (page 117)
Figure 31: Object 6063_h *position_actual_value** is a normalised parameter.
- page 178 (page 170)
Figure 46: *vl_velocity_target* changed to ***vl_target_velocity*** see object 6042_h.

- page 38, 55 (page 37, 54)
Object 603F_h, Object 6040_h
Data Type changed to **Unsigned16**, therefore *Value Range* changed to 0..65553.
- page 129
Object 60F4_h: **New**
following_error_actual_value: This object represents the actual value of the following error.
- page 19-26
Index attached.
- page 153 (page 144)
Figure 40: *target_velocity* indexnumber (60FF_h) added. *Unit* changed to *velocity units*.
- page 154 (page 146)
Output Data Description wrong text deleted.
- page 156 (page 148)
velocity_sensor_actual_value formatted.
- page 156 (page 148)
velocity_actual_value changed to *velocity_***sensor***_actual_value*.
- page 26,28 (page 25, 27)
4th RPDO second entry is **target_velocity** Index 60FF_h.
- page 78, 79 (page 76, 77)
target_velocity added.

Table of Contents

1	Scope	6
2	References	8
3	Definitions, Acronyms and Abbreviation	9
4	Overview	10
4.1	Access to the Drive	10
4.2	Architecture of the Drive	11
5	Operating Principle	15
5.1	Introduction	15
5.2	Standardisation via Profiling	15
5.3	The Object Dictionary	16
5.3.1	Index and sub-index usage	17
6	Emergency Messages	19
6.1	Principle	19
6.2	Error Code Meanings	19
7	Predefinitions	24
7.1	Naming conventions	24
7.2	Predefined Objects	24
7.2.1	Object 1000 _h : Device Type	24
7.2.2	Object 1001 _h : Error Register	25
7.2.3	Object 67FF _h : Single Device Type	25
7.3	PDO Mapping	25
7.3.1	Receive PDOs	26
7.3.2	Transmit PDOs	30
8	Object Dictionary	34
9	Common Entries in the Object Dictionary	35
9.1	General Information	35
9.1.1	Motor Data	35
9.1.2	Drive Data	35
9.2	Object Dictionary Entries	36
9.2.1	Objects defined in this Chapter	36
9.3	Object Description	37
9.3.1	Object 6007 _h : abort_connection_option_code	37
9.3.2	Object 603F _h : error_code	38
9.3.3	Object 6402 _h : motor_type	38
9.3.4	Object 6403 _h : motor_catalogue_number	39
9.3.5	Object 6404 _h : motor_manufacturer	40

9.3.6	Object 6405 _h : http_motor_catalog_address	40
9.3.7	Object 6406 _h : motor_calibration_date	41
9.3.8	Object 6407 _h : motor_service_period	41
9.3.9	Object 6410 _h : motor_data	42
9.3.10	Object 6502 _h : supported_drive_modes	43
9.3.11	Object 6503 _h : drive_catalogue_number	44
9.3.12	Object 6504 _h : drive_manufacturer	44
9.3.13	Object 6505 _h : http_drive_catalog_address	45
9.3.14	Object 6510 _h : drive_data	45
9.3.15	Object 60FD _h : digital_inputs	46
9.3.16	Object 60FE _h : digital_outputs	47
10	Device Control	49
10.1	General Information	49
10.1.1	Statemachine	50
10.2	Object Dictionary Entries	55
10.2.1	Objects defined in this Chapter	55
10.3	Object Description	55
10.3.1	Object 6040 _h : controlword	55
10.3.2	Object 6041 _h : statusword	58
10.3.3	Object 605B _h : shutdown_option_code	63
10.3.4	Object 605C _h : disable_operation_option_code	64
10.3.5	Object 605A _h : quick_stop_option_code	65
10.3.6	Object 605D _h : stop_option_code	66
10.3.7	Object 605E _h : fault_reaction_option_code	67
10.3.8	Object 6060 _h : modes_of_operation	68
10.3.9	Object 6061 _h : modes_of_operation_display	69
10.4	Functional Description	69
10.4.1	Modes of Operation Function	69
10.4.2	Drive Disabling Function	71
10.4.3	Quick Stop Function	71
10.4.4	Stop Function	72
10.4.5	Fault Reaction	72
11	Factor Group	74
11.1	General Information	74
11.1.1	Factors	74
11.1.2	Relationship between Physical and Internal Units	74
11.2	Object Dictionary Entries	77
11.2.1	Objects defined in this Chapter	77
11.3	Object Description	78
11.3.1	Object 6089 _h : position_notation_index	78
11.3.2	Object 608A _h : position_dimension_index	79
11.3.3	Object 608B _h : velocity_notation_index	80
11.3.4	Object 608C _h : velocity_dimension_index	81
11.3.5	Object 608D _h : acceleration_notation_index	82
11.3.6	Object 608E _h : acceleration_dimension_index	83
11.3.7	Object 608F _h : position_encoder_resolution	84
11.3.8	Object 6090 _h : velocity_encoder_resolution	85
11.3.9	Object 6091 _h : gear_ratio	86
11.3.10	Object 6092 _h : feed_constant	87

11.3.11 Object 6093 _h : position_factor	88
11.3.12 Object 6094 _h : velocity_encoder_factor	89
11.3.13 Object 6095 _h : velocity_factor_1	90
11.3.14 Object 6096 _h : velocity_factor_2	91
11.3.15 Object 6097 _h : acceleration_factor	92
11.3.16 Object 607E _h : polarity	93

12 Profile Position Mode **94**

12.1 General Information	94
12.1.1 Input Data Description	95
12.1.2 Output Data Description	96
12.2 Object Dictionary Entries	96
12.2.1 Objects defined in this Chapter	96
12.2.2 Objects defined in other Chapters	96
12.3 Object Description	97
12.3.1 Object 607A _h : target_position	97
12.3.2 Object 607B _h : position_range_limit	98
12.3.3 Object 607D _h : software_position_limit	99
12.3.4 Object 607F _h : max_profile_velocity	100
12.3.5 Object 6080 _h : max_motor_speed	101
12.3.6 Object 6081 _h : profile_velocity	101
12.3.7 Object 6082 _h : end_velocity	102
12.3.8 Object 6083 _h : profile_acceleration	102
12.3.9 Object 6084 _h : profile_deceleration	103
12.3.10 Object 6085 _h : quick_stop_deceleration	103
12.3.11 Object 6086 _h : motion_profile_type	104
12.3.12 Object 60C5 _h : max_acceleration	105
12.3.13 Object 60C6 _h : max_deceleration	106
12.4 Functional Description	106

13 Homing Mode **109**

13.1 General Information	109
13.1.1 Input Data Description	109
13.1.2 Output Data Description	109
13.1.3 Internal States	109
13.2 Object Dictionary Entries	110
13.2.1 Objects defined in this Chapter	110
13.2.2 Objects defined in other Chapters	110
13.3 Object Description	110
13.3.1 Object 607C _h : home_offset	110
13.3.2 Object 6098 _h : homing_method	111
13.3.3 Object 6099 _h : homing_speeds	112
13.3.4 Object 609A _h : homing_acceleration	113
13.4 Functional Description	113
13.4.1 Homing Methods	114
13.4.2 Homing Mode Sequence	119

14 Position Control Function **120**

14.1 General information	120
14.1.1 Input Data Description	122

14.1.2	Output Data Description	122
14.2	Object Dictionary Entries	123
14.2.1	Objects defined in this Chapter	123
14.2.2	Objects defined in other Chapters	123
14.3	Object Description	124
14.3.1	Object 6062 _h : position_demand_value	124
14.3.2	Object 6063 _h : position_actual_value*	124
14.3.3	Object 6064 _h : position_actual_value	125
14.3.4	Object 6065 _h : following_error_window	126
14.3.5	Object 6066 _h : following_error_time_out	127
14.3.6	Object 6067 _h : position_window	128
14.3.7	Object 6068 _h : position_window_time	129
14.3.8	Object 60F4 _h : following_error_actual_value	129
14.3.9	Object 60FA _h : control_effort	130
14.3.10	Object 60FB _h : position_control_parameter_set	131
14.3.11	Object 60FC _h : position_demand_value*	132
14.4	Functional Description	133
15	Interpolated Position Mode	134
15.1	General Information	134
15.1.1	Input Data Description	136
15.1.2	Output Data Description	136
15.1.3	Internal States	137
15.1.4	State Transitions of the Internal States	137
15.2	Object Dictionary Entries	138
15.2.1	Objects defined in this Chapter	138
15.2.2	Objects defined in other Chapters	139
15.3	Object Descriptions	140
15.3.1	Object 60C0 _h : interpolation_submode_select	140
15.3.2	Object 60C1 _h : interpolation_data_record	140
15.3.3	Object 60C2 _h : interpolation_time_period	142
15.3.4	Object 60C3 _h : interpolation_sync_definition	143
15.3.5	Object 60C4 _h : interpolation_data_configuration	144
15.4	Functional Description	147
15.4.1	Interpolated Position Mode	147
15.4.2	Linear Interpolated Position Mode with several Axes	147
15.4.3	Buffer Strategies for the Interpolated Position Mode	149
16	Profile Velocity Mode	151
16.1	General Information	151
16.1.1	Input Data Description	154
16.1.2	Output Data Description	154
16.2	Object Dictionary Entries	154
16.2.1	Objects defined in this Chapter	154
16.2.2	Objects defined in other Chapters	155
16.3	Object Description	155
16.3.1	position_encoder	155
16.3.2	Object 6069 _h : velocity_sensor_actual_value	156
16.3.3	Object 606A _h : sensor_selection_code	156
16.3.4	Object 606B _h : velocity_demand_value	157

16.3.5	Object 606C _h : velocity_actual_value	158
16.3.6	Object 606D _h : velocity_window	158
16.3.7	Object 606E _h : velocity_window_time	159
16.3.8	Object 606F _h : velocity_threshold	159
16.3.9	Object 6070 _h : velocity_threshold_time	160
16.3.10	Object 60FF _h : target_velocity	160
16.3.11	Object 60F8 _h : max_slippage	161
16.3.12	Object 60F9 _h : velocity_control_parameter_set	162

17 Profile Torque Mode **163**

17.1	General Information	163
17.1.1	Structure of the Profile Torque Mode	164
17.2	Object Dictionary Entries	165
17.2.1	Objects defined in this Chapter	165
17.2.2	Objects defined in other chapters	165
17.3	Object Description	166
17.3.1	Object 6071 _h : target_torque	166
17.3.2	Object 6072 _h : max_torque	166
17.3.3	Object 6073 _h : max_current	167
17.3.4	Object 6074 _h : torque_demand_value	167
17.3.5	Object 6075 _h : motorRatedCurrent	168
17.3.6	Object 6076 _h : motorRatedTorque	168
17.3.7	Object 6077 _h : torque_actual_value	169
17.3.8	Object 6078 _h : current_actual_value	169
17.3.9	Object 6079 _h : DC_link_circuit_voltage	170
17.3.10	Object 6087 _h : torque_slope	170
17.3.11	Object 6088 _h : torque_profile_type	171
17.3.12	Object 60F7 _h : power_stage_parameters	172
17.3.13	Object 60F6 _h : torque_control_parameters	173

18 Velocity Mode **174**

18.1	General Description	174
18.1.1	Input Data Description	175
18.1.2	Output Data Description	175
18.1.3	Structure of the Velocity Mode	175
18.1.4	Subfunction Description	178
18.2	Object Dictionary Entries	179
18.2.1	Objects defined in this Chapter	179
18.2.2	Objects defined in other Chapters	179
18.3	Object Description	180
18.3.1	Object 6042 _h : vl_target_velocity	180
18.3.2	Object 6043 _h : vl_velocity_demand	181
18.3.3	Object 6053 _h : vl_percentage_demand	182
18.3.4	Object 6054 _h : vl_actual_percentage	183
18.3.5	Object 6055 _h : vl_manipulated_percentage	184
18.3.6	Object 604E _h : vl_velocity_reference	185
18.3.7	Object 604C _h : vl_dimension_factor	186
18.3.8	Object 604B _h : vl_setpoint_factor	188
18.3.9	Object 604D _h : vl_pole_number	189
18.3.10	Object 6046 _h : vl_velocity_min_max_amount	190
18.3.11	Object 6047 _h : vl_velocity_min_max	191
18.3.12	Object 6058 _h : vl_frequency_motor_min_max_amount	193

18.3.13 Object 6059 _h : vl_frequency_motor_min_max	195
18.3.14 Object 6056 _h : vl_velocity_motor_min_max_amount	197
18.3.15 Object 6057 _h : vl_velocity_motor_min_max	198
18.3.16 Object 6048 _h : vl_velocity_acceleration	200
18.3.17 Object 6049 _h : vl_velocity_deceleration	202
18.3.18 Object 604A _h : vl_velocity_quick_stop	204
18.3.19 Object 604F _h : vl_ramp_function_time	205
18.3.20 Object 6050 _h : vl_slow_down_time	206
18.3.21 Object 6051 _h : vl_quick_stop_time	207
18.3.22 Object 6044 _h : vl_control_effort	208
18.3.23 Object 6045 _h : vl_manipulated_velocity	208
18.3.24 Object 6052 _h : vl_nominal_percentage	209
18.4 Functional Description	210
18.4.1 Percentage Function	210
18.4.2 Factor Function and Reverse Factor Function	210
18.4.3 Pole Number Function	211
18.4.4 Velocity Limit Function	212
18.4.5 Velocity Motor Limit Function	212
18.4.6 Ramp Function	213
18.4.7 Ramp Min Function	214
18.4.8 Reference Calculation	215
18.4.9 Closed Open Loop Control Function	215
19 Appendix	19-1
A Object Dictionary by Chapter	19-1
A.1 Common Entries in the Object Dictionary	19-1
A.2 Device Control	19-2
A.3 Factor Group	19-3
A.4 Profile Position Mode	19-5
A.5 Homing Mode	19-6
A.6 Position Control Function	19-7
A.7 Interpolated Position Mode	19-8
A.8 Profile Velocity Mode	19-9
A.9 Profile Torque Mode	19-10
A.10 Velocity Mode	19-11
B Object Dictionary by Index	19-13
C Object Dictionary by Name	19-18
D Definition of Dimension Indices	19-23
D.1 Dimension/Notation Index Table	19-23
D.2 Examples for Notation Indices	19-25
E Index	19-26

1 Scope

This document represents the standardised CANopen Device Profile for digital controlled motion products like servo controllers, frequency converters or stepper motors.

All the above devices use communication techniques which conform to those described in the CiA Draft Standard DS-301 (CAL based communication profile for industrial systems). This document should be consulted in parallel to this profile.

2 References

- /1/: ISO 7498, 1984, Information Processing Systems - Open Systems Interconnection - Basic Reference Model
- /2/: ISO 11898, November 1993, Road Vehicles, Interchange of Digital Information - Controller Area Network (CAN) for high-speed Communication
- /3/: CiA/DS 102, CAN Physical Layer for Industrial Applications, April 1994
- /4/: CiA/DS 201, CAN Reference Model, Version 1.1, Feb. 1996
- /5/: CiA/DS 202-1, CMS Service Specification, Version 1.1, Feb. 1996
- /6/: CiA/DS 202-2, CMS Protocol Specification, Version 1.1, Feb. 1996
- /7/: CiA/DS 202-3, CMS Encoding Rules, Version 1.1, Feb. 1996
- /8/: CiA/DS 203-1, NMT Service Specification, Version 1.1, Feb. 1996
- /9/: CiA/DS 203-2, NMT Protocol Specification, Version 1.1, Feb. 1996
- /10/: CiA/DS 204-1, DBT Service Specification, Version 1.1, Feb. 1996
- /11/: CiA/DS 204-2, DBT Protocol Specification, Version 1.1, Feb. 1996
- /12/: CiA/DS 207, Application Layer Naming Specification, Version 1.1, Feb. 1996
- /13/: CiA/DS 205-1, LMT Service Specification, Version 1.1, Feb. 1996
- /14/: CiA/DS 205-2, LMT Protocol Specification, Version 1.1, Feb. 1996
- /15/: CiA/DS 206, Application Specific Data Types, Version 1.1, Feb. 1996
- /16/: CiA/DS 301, CAL-based Communication Profile, Version 3.0, Oct. 1996
- /17/: CiA/DS 401, CANopen Device Profile for I/O Modules, Version 1.4, Dec. 1996
- /18/: DRIVECOM Profil Antriebstechnik/Profil 21
- /19/: DRIVECOM Profil Antriebstechnik/Servo 22, Jan. 1994

3 Definitions, Acronyms and Abbreviation

CAN	<u>C</u> ontroller <u>A</u> rea <u>N</u> etwork
CiA	<u>C</u> AN in <u>A</u> utomation e. V. international users and manufactorers group.
CMS	<u>C</u> AN based <u>M</u> essage <u>S</u> pecification. One of the service elements of the application layer in the CAN Reference Model.
COB	<u>C</u> ommunication <u>O</u> bject (CAN Message). A unit of transportation in a CAN network. Data must be sent across a network inside a COB.
COB-ID	<u>C</u> OB- <u>I</u> dentifier. Identifies a COB uniquely in a network. The identifier determines the priority of that COB in the MAC sub-layer too.
DBT	<u>D</u> istrib <u>u</u> tor. One of the service elements of the application in the CAN Reference Model. It's the responsibility of the DBT to distribute COB-IDs to the COBs that are used by the CMS.
LMT	<u>L</u> ayer <u>M</u> anagement. One of the service elements of the application in the CAN Reference Model. It serves to configure parameters of each layer in the CAN Reference Model.
MAC	<u>M</u> edium <u>A</u> ccess <u>C</u> ontrol. One of the sub-layers of the Datalink Layer in the CAN Reference Model that controls who gets access to the medium to send a message.
NMT	<u>N</u> etwork <u>M</u> anagement. One of the service elements of the application in the CAN Reference Model. It performs initialisation, configuration and error handling in a CAN network.
PDO	<u>P</u> rocess <u>D</u> ata <u>O</u> bject. Object for data exchange between several devices.
SDO	<u>S</u> ervice <u>D</u> ata <u>O</u> bject. Peer to peer communication with access to the Object dictionary of a device.
pp	Profile Position Mode
pv	Profile Velocity Mode
vl	Velocity Mode
hm	Homing Mode
ip	Interpolated Position Mode
tq	Profile Torque Mode
all	mandatory for all modes
ce	Common Entries in the Object Dictionary
dc	Device Control
pc	Position Control Function

4 Overview

4.1 Access to the Drive

The access from the CAN network to the drive is done through data objects.

Data Objects of the Drive

PDO	SDO	IDO
Process Data Object	Service Data Object	Internal Data Object
described in chapters 9 to 18	described in chapter 7	manufacturer specific normally not accessible

Figure 1: Data Objects of the Drive

Process Data Object (PDO): PDOs are messages in an unconfirmed service (see /16/). They are used for the transfer of real-time data to and from the drive. The transfer is fast, because it is performed with no protocol overhead what means to transport eight application data bytes in one CAN-frame. The PDOs correspond to entries in the Object dictionary described in chapters 9 to 18. The data type and mapping of these objects into a PDO is described in chapter 7.

Service Data Object (SDO): SDOs are messages in a confirmed service with a kind of handshake (see /16/). They are used for the access to entries of the Object dictionary. Especially the configuration for the requested behaviour of the drive adapted to the various possible applications is done by these objects.

Internal Data Object (IDO): The internal data objects represent the adaptation of the manufacturer and device specific functionality to this profile. Normally these objects are not directly accessible; nevertheless a manufacturer can give the user access to the IDOs by SDO services.

4.2 Architecture of the Drive

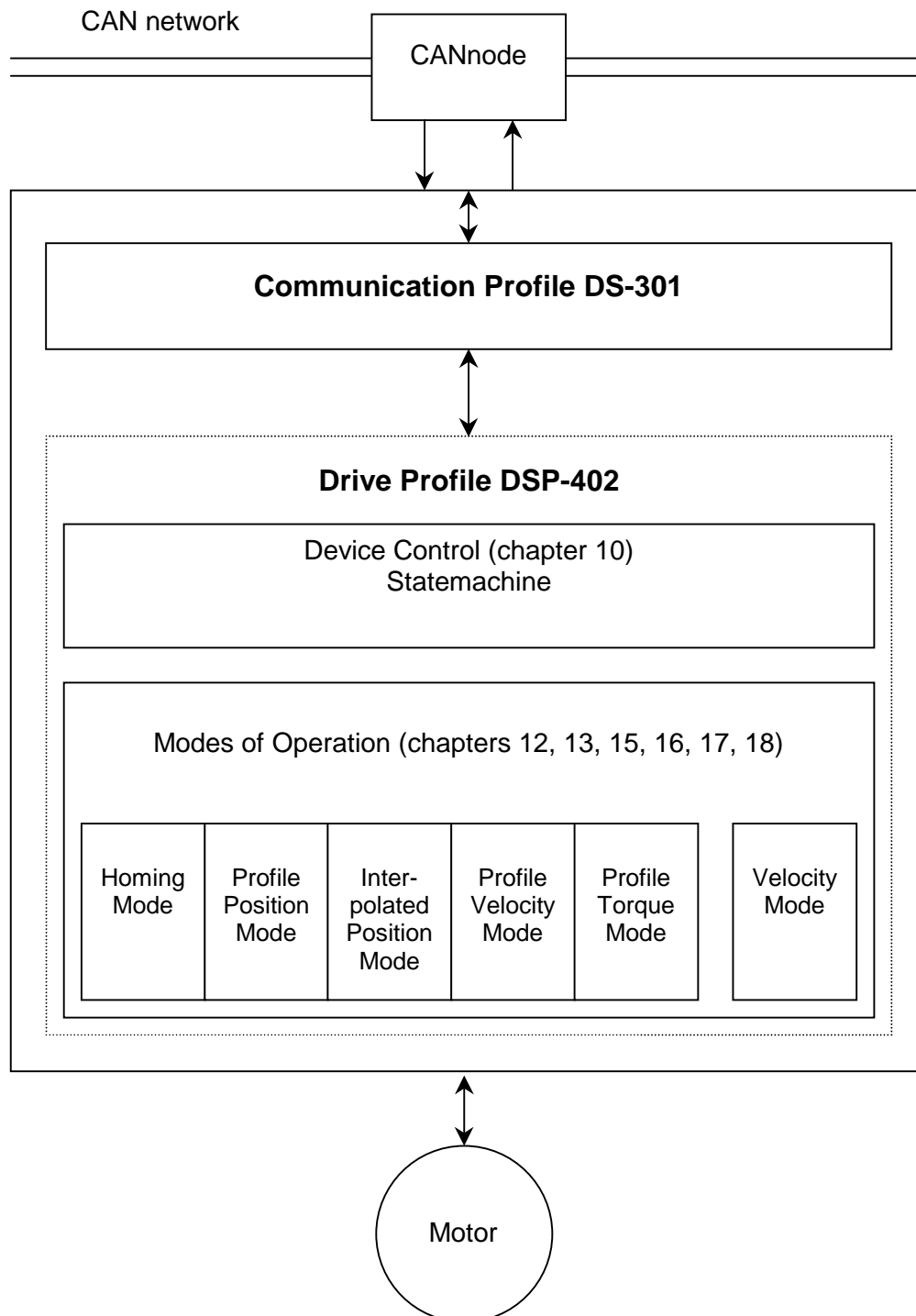


Figure 2: Communication Architecture

Device Control: The starting and stopping of the drive and several mode specific commands are executed by the statemachine. This is described in chapter 10. The mode specific actions are described in chapter 12 to 18.

Modes of Operation: The operation mode defines the behaviour of the drive. The following modes are defined in this profile:

Homing Mode (chapter 13)

This chapter describes the various methods to find a home position (also: reference point, datum, zero point).

Profile Position Mode (chapter 12)

The positioning of the drive is defined in this mode. Speed, position and acceleration can be limited and profiled moves using a Trajectory Generator are possible as well.

Interpolated Position Mode (chapter 15)

This chapter describes the time interpolation of single axles and the spatial interpolation of co-ordinated axles. Synchronisation mechanisms and interpolation data buffers are covered by this chapter.

Profile Velocity Mode (chapter 16)

The Profile Velocity Mode is used to control the velocity of the drive with no special regard of the position. It supplies limit functions and trajectory generation.

Profile Torque Mode (chapter 17)

In this chapter the torque control with all related parameters is described.

Velocity Mode (chapter 18)

Many frequency inverters use this simple mode to control the velocity of the drive with limits and ramp functions.

The Velocity Mode (chapter 18) is rather separated from the other modes and does not interfere with them so much. For this reason, the naming of object dictionary entries differs a little bit from the other chapters.

The manufacturer commits in the manual which modes are supported by his device.

If more than one mode is supported, then the manufacturer also defines whether the change of operation mode is allowed while the drive is moving or only when the drive is stopped.

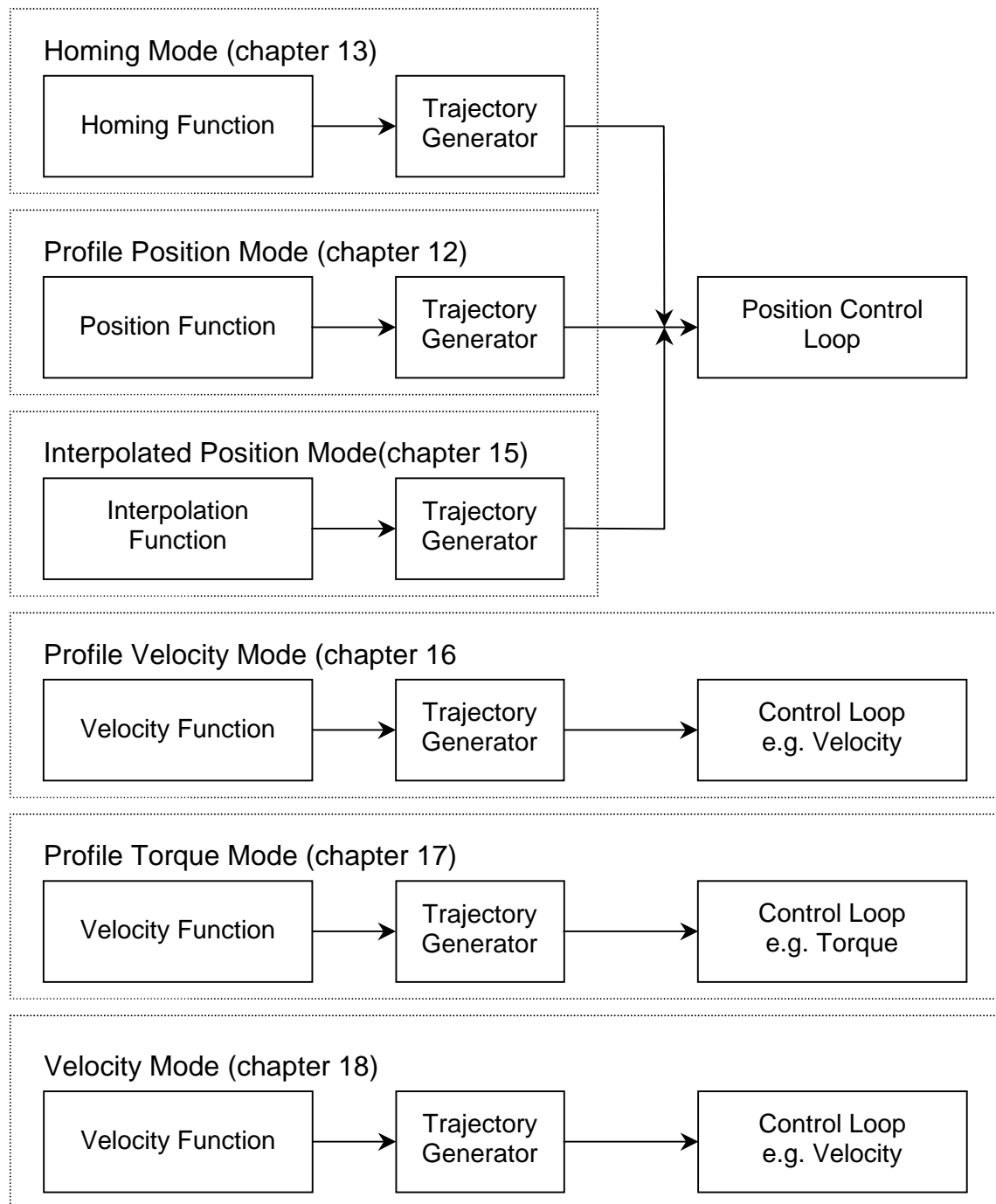


Figure 3: Functional Architecture

Trajectory Generator: The chosen operation mode and the corresponding parameters (objects) define the input of the Trajectory Generator. The Trajectory Generator supplies the control loop(s) with the demand values. They are generally mode specific.

Each Mode may use its own Trajectory Generator. A general description of its functionality is given in chapter 12, which is related to the Profile Position Mode.

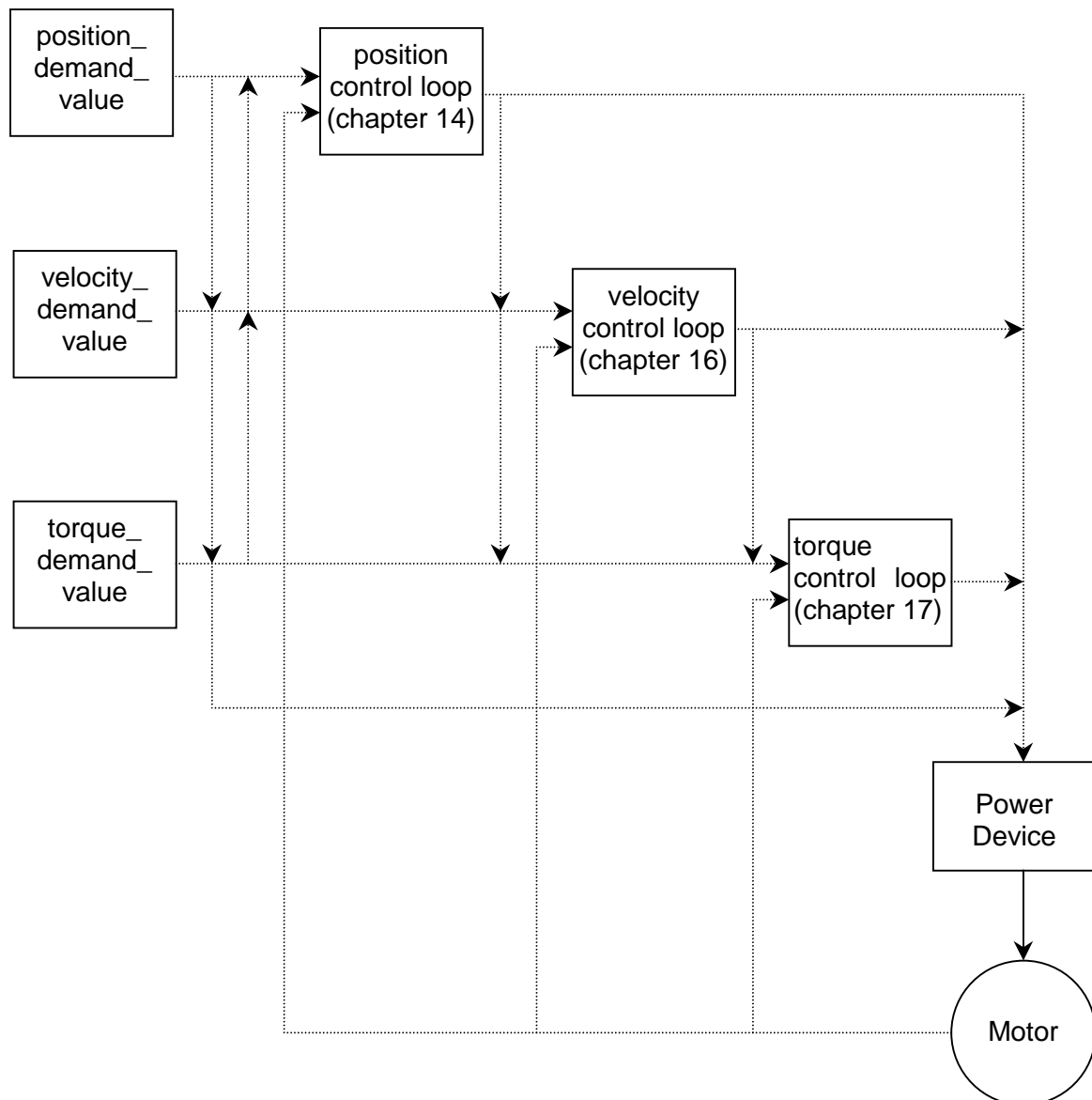


Figure 4: Possible Structures of the Control Loop

Control Loop: The implementation of the control loop is highly manufacturer specific and not described in this profile. Possible control loop structures are shown in the picture above.

The control loop can be open or closed and it can be operation mode specific or fixed. The objects which are described in chapter 12 to 18 must be implemented, if the corresponding mode is supported and if they are mandatory. But it is allowed that the manufacturer uses objects of the velocity controller in the Profile Position Mode; for example the control loop structure consists of a position controller producing a velocity demand value and a velocity controller using this as a demand value.

5 Operating Principle

5.1 Introduction

The purpose of this profile is, to give drives an understandable and unique behaviour on the CAN network. The CANopen Device Profile for Drives and Motion Control is built on top of a CAN communication profile, called CANopen, describing the basic communication mechanisms common to all devices at the CAN-network.

The purpose of drive units is to connect axle controllers or other motion control products to the CAN bus. They can receive configuration information what is done via service data objects normally for I/O configurations, limit parameters for scaling or application specific parameters. At run time, data can be obtained from the drive unit via CAN bus by either polling or event driven (interrupt).

The motion control products have a process data object mapping for real time operation, which may be configured using service data objects (see /16/). This communication channel is used to interchange real-time data like setpoints or actual values like a *position_actual_value* e.g.

5.2 Standardisation via Profiling

The two principal advantages of the profile approach for device specification are in the areas of system integration and device standardisation.

If two independent device manufacturers design products that have to communicate, then both manufacturers must be provided with a device specification from the other one. These specifications will widely differ in formal and terminological aspects from one company to another. The concept of device profiling provides a standard for producing such specifications. By adopting this approach, all manufacturers will specify their devices in a similar fashion, what greatly reduces the effort involved in system integration.

The other obvious advantage of the profile approach for device specification is, that it can be used to guide manufacturers into producing standardised devices. The advantages of standardised devices are numerous. Perhaps most important is the idea, that a standardised device decouples a system integrator from a specific supplier. If one supplier cannot meet special application demands, a system designer can use devices from another supplier with reduced effort. On the other hand the device manufacturers are not forced any more to implement private protocols for each customer.

A device profile defines a 'standard' device. This standard device represents really basic functionality, every device within this device class must support. This mandatory functionality is necessary to ensure, that at least simple non-manufacturer-specific operation of a device is possible. For example the standard drive unit provides a 'quickstop' function to stop a drive. This function is defined as mandatory, such that any drive unit supporting the CANopen Device Profile for Drives and Motion Control, can be halted using the same message.

The concept of device standardisation is extended by the notion of optional functionality defined within the standardised device profile. Such optional functionality does not have to

be implemented by all manufacturers. However, if a manufacturer implements such functionality he must do so in a fixed manner.

Providing optional functionality is a very powerful mechanism to ensure all manufacturers implementing particular functionality in a defined fashion. For example, the device profile covers multi-axes modules as well, which are still not very common. By defining a standardised access to the different axes, interchanging devices from different manufacturers becomes easier.

The device profiles provide a mechanism by which manufacturers wishing to implement truly manufacturer specific functionality can do so as well. This is clearly necessary since it would be impossible to anticipate all possible device functionality and define this in the optional category of each device class. This concept guarantees that the standard device profiles are 'future-proof'.

By defining mandatory device characteristics, basic network operation is guaranteed. By defining optional device features a degree of defined flexibility can be built in. By leaving 'hooks' for manufacturer specific functionality, manufacturers will not be constrained to an out-of-date standard.

5.3 The Object Dictionary

The most important part of a device profile is the object dictionary description. The object dictionary is essentially a grouping of objects accessible via the network in an ordered pre-defined fashion. Each object within the dictionary is addressed using a 16-bit index so that the object dictionary may contain a maximum of 65536 entries.

The overall layout of the standard object dictionary is shown below. This layout closely conforms with device profiles for other fieldbus systems :

Index (hex)	Object
0000	not used
0001-001F	Static Data Types
0020-003F	Complex Data Types
0040-005F	Manufacturer Specific Data Types
0060-0FFF	Reserved for further use
1000-1FFF	Communication Profile Area
2000-5FFF	Manufacturer Specific Profile Area
6000-9FFF	Standardised Device Profile Area
A000-FFFF	Reserved for further use

Table 1: Object Dictionary Structure

The static data types at indices 0001_h through 001F_h contain type definitions for standard data types like boolean, integer, floating point, string, etc. These entries are included for reference only, they cannot be read or written.

Complex data types at indices 0020_h through 003F_h are pre-defined structures that are composed out of standard data types and are common to all devices.

Manufacturer Specific Data Types at indices 0040_h through 005F_h are also structures composed of standard data types but are specific to a particular device.

The Communication Profile Area at indices 1000_h through 1FFF_h contains the parameters for the communication profile on the CAN network. These entries are common to all devices.

The standardised device profile area at indices 6000_h through 9FFF_h contains all data objects common to a class of devices that can be read or written via the network. The drives profile uses entries from 6000_h to 9FFF_h to describe the drive parameters and the drive functionality. Within this range up to 8 axles can be realised. Additional it is possible to describe optional I/O modules combined with the drive. These I/O modules must conform to DS-401 (see /17/) and can be implemented instead of an axle. For standard drives only the range 6000_h to 67FF_h is mandatory. There are also two reserved areas at indices 060_h through 0FFF_h and A000_h through FFFF_h for future use by the communication or drive profile.

For multi axles devices the object range 6000_h to 67FF_h is shifted as follows:

6000 _h to 67FF _h	axle 0
6800 _h to 6FFF _h	axle 1
7000 _h to 77FF _h	axle 2
7800 _h to 7FFF _h	axle 3
8000 _h to 87FF _h	axle 4
8800 _h to 8FFF _h	axle 5
9000 _h to 97FF _h	axle 6
9800 _h to 9FFF _h	axle 7

5.3.1 Index and sub-index usage

A 16-bit index is used to address all entries within the object dictionary. In case of a simple variable this references the value of this variable directly. In case of records and arrays however, the index addresses the whole data structure. To allow individual elements of structures of data to be accessed via the network a sub-index has been defined. For single object dictionary entries such as an Unsigned8, Boolean, Integer32 etc. the value for the sub-index is always zero. For complex object dictionary entries such as arrays or records with multiple data fields the sub-index refers to fields within a data-structure pointed to by the main index. Index counting starts with one. For example in the chapter Factor Group exists the object 608F_h named *position_encoder_resolution*. Because this may be a fraction, two integers in an array are used to describe it. The drive uses the two values in the following manner:

$$\text{position_encoder_resolution} = \frac{\text{encoder_increments}}{\text{motor_revolutions}}$$

The sub-index concept can be used to access these individual fields which may be of different data type as shown below:

Main Index	Sub Index	Variable Accessed	Data Type
648F	0	Number of elements	Unsigned8
	1	Encoder_increments	Unsigned32
	2	Motor_revolutions	Unsigned32

Table 2: Usage of index and sub-index

6 Emergency Messages

6.1 Principle

Emergency messages are triggered by internal errors in the device and they are assigned the highest possible priority to ensure that they get access to the bus without latency. By default, the emergency messages contain an error field with pre-defined error numbers and additional information.

The error number is of Unsigned32 type. The lower two bytes contain the error code, the upper two bytes may contain additional error information. The high byte of the error code is used for an error classification while the low byte contains the error number for this class (see also /16/).

Error numbers from $xx00_h$ to $xx7F_h$ are defined in the communication profile DS-301 or in this profile DSP402. Not defined error numbers within this range are reserved. Error numbers between $xx80_h$ and $xxFF_h$ can be used manufacturer specific.

6.2 Error Code Meanings

Error Code (hex)	Meaning	Defined By
0000	no error	Comm. Prof.
1000	generic error	Comm. Prof.
2000	current	Comm. Prof.
2100	current on device input side	
2110	short circuit/earth leakage	Drives Prof.
2120	earth leakage	Drives Prof.
2121	earth leakage phase L1	Drives Prof.
2122	earth leakage phase L2	Drives Prof.
2123	earth leakage phase L3	Drives Prof.
2130	short circuit	Drives Prof.
2131	short circuit phases L1-L2	Drives Prof.
2132	short circuit phases L2-L3	Drives Prof.
2133	short circuit phases L3-L1	Drives Prof.
2200	internal current	Drives Prof.
2211	internal current No.1	Drives Prof.
2212	internal current No.2	Drives Prof.
2213	over-current in ramp function	Drives Prof.
2214	over-current in the sequence	Drives Prof.
2220	continuous over current	Drives Prof.
2221	continuous over current No.1	Drives Prof.
2222	continuous over current No.2	Drives Prof.
2230	short circuit/earth leakage	
2240	earth leakage	Drives Prof.
2250	short circuit	Drives Prof.

Error Code (hex)	Meaning	Defined By
2300	current on device output side	Drives Prof.
2310	continuous over current	Drives Prof.
2311	continuous over current No.1	Drives Prof.
2312	continuous over current No.2	Drives Prof.
2320	short circuit/earth leakage	Drives Prof.
2330	earth leakage	Drives Prof.
2331	earth leakage phase U	Drives Prof.
2332	earth leakage phase V	Drives Prof.
2333	earth leakage phase W	Drives Prof.
2340	short circuit	Drives Prof.
2341	short circuit phases U-V	Drives Prof.
2342	earth leakage phase V-W	Drives Prof.
2343	earth leakage phase W-U	Drives Prof.
3000	voltage	Comm. Prof.
3100	mains voltage	Drives Prof.
3110	mains over-voltage	Drives Prof.
3111	mains over-voltage phase L1	Drives Prof.
3112	mains over-voltage phase L2	Drives Prof.
3113	mains over-voltage phase L3	Drives Prof.
3120	mains under-voltage	Drives Prof.
3121	mains under-voltage phase L1	Drives Prof.
3122	mains under-voltage phase L2	Drives Prof.
3123	mains under-voltage phase L3	Drives Prof.
3130	phase failure	Drives Prof.
3131	phase failure L1	Drives Prof.
3132	phase failure L2	Drives Prof.
3133	phase failure L3	Drives Prof.
3134	phase sequence	Drives Prof.
3140	mains frequency	Drives Prof.
3141	mains frequency too great	Drives Prof.
3142	mains frequency too small	Drives Prof.
3200	DC link voltage	Drives Prof.
3210	DC link over-voltage	Drives Prof.
3211	over-voltage No. 1	Drives Prof.
3212	over voltage No. 2	Drives Prof.
3220	DC link under-voltage	Drives Prof.
3221	under-voltage No. 1	Drives Prof.
3222	under-voltage No. 2	Drives Prof.
3230	load error	Drives Prof.
3300	output voltage	Drives Prof.
3310	output over-voltage	Drives Prof.
3311	output over-voltage phase U	Drives Prof.
3312	output over-voltage phase V	Drives Prof.
3313	output over-voltage phase W	Drives Prof.
3320	armature circuit	Drives Prof.
3321	armature circuit interrupted	Drives Prof.
3330	field circuit	Drives Prof.
3331	field circuit interrupted	Drives Prof.

Error Code (hex)	Meaning	Defined By
4000	temperature	Comm. Prof.
4100	ambient temperature	Drives Prof.
4110	excess ambient temperature	Drives Prof.
4120	too low ambient temperature	Drives Prof.
4130	temperature supply air	Drives Prof.
4140	temperature air outlet	Drives Prof.
4200	temperature device	Drives Prof.
4210	excess temperature device	Drives Prof.
4220	too low temperature device	Drives Prof.
4300	temperature drive	Drives Prof.
4310	excess temperature drive	Drives Prof.
4320	too low temperature drive	Drives Prof.
4400	temperature supply	Drives Prof.
4410	excess temperature supply	Drives Prof.
4420	too low temperature supply	Drives Prof.
5000	device hardware	Comm. Prof.
5100	supply	Drives Prof.
5110	supply low voltage	Drives Prof.
5111	U1 = supply +/- 15V	Drives Prof.
5112	U2 = supply +24 V	Drives Prof.
5113	U3 = supply +5 V	Drives Prof.
5114	U4 = manufacturer specific	Drives Prof.
5115	U5 = manufacturer specific	Drives Prof.
5116	U6 = manufacturer specific	Drives Prof.
5117	U7 = manufacturer specific	Drives Prof.
5118	U8 = manufacturer specific	Drives Prof.
5119	U9 = manufacturer specific	Drives Prof.
5120	supply intermediate circuit	Drives Prof.
5200	control	Drives Prof.
5210	measurement circuit	Drives Prof.
5220	computing circuit	Drives Prof.
5300	operating unit	Drives Prof.
5400	power section	Drives Prof.
5410	output stages	Drives Prof.
5420	chopper	Drives Prof.
5430	input stages	Drives Prof.
5440	contacts	Drives Prof.
5441	contact 1 = manufacturer specific	Drives Prof.
5442	contact 2 = manufacturer specific	Drives Prof.
5443	contact 3 = manufacturer specific	Drives Prof.
5444	contact 4 = manufacturer specific	Drives Prof.
5445	contact 5 = manufacturer specific	Drives Prof.
5450	fuses	Drives Prof.
5451	S1 = L1	Drives Prof.
5452	S2 = L2	Drives Prof.
5453	S3 = L3	Drives Prof.
5454	S4 = manufacturer specific	Drives Prof.
5455	S5 = manufacturer specific	Drives Prof.
5456	S6 = manufacturer specific	Drives Prof.
5457	S7 = manufacturer specific	Drives Prof.

Error Code (hex)	Meaning	Defined By
5458	S8 = manufacturer specific	Drives Prof.
5459	S9 = manufacturer specific	Drives Prof.
5500	data storage	Drives Prof.
5510	RAM	Drives Prof.
5520	EPROM	Drives Prof.
5530	EEPROM	Drives Prof.
6000	device software	Comm. Prof.
6010	software reset (watchdog)	Drives Prof.
6100	internal software	Drives Prof.
6200	user software	Drives Prof.
6300	data record	Drives Prof.
6301	data record No. 1	Drives Prof.
	Drives Prof.
630F	data record No.15	Drives Prof.
6310	loss of parameters	Drives Prof.
6320	parameter error	Drives Prof.
7000	additional modules	Comm. Prof.
7100	power	Drives Prof.
7110	brake chopper	Drives Prof.
7111	failure brake chopper	Drives Prof.
7112	over current brake chopper	Drives Prof.
7113	protective circuit brake chopper	Drives Prof.
7120	motor	Drives Prof.
7121	motor blocked	Drives Prof.
7122	motor error or commutation malfunc.	Drives Prof.
7123	motor tilted	Drives Prof.
7200	measurement circuit	Drives Prof.
7300	sensor	Drives Prof.
7301	tacho fault	Drives Prof.
7302	tacho wrong polarity	Drives Prof.
7303	resolver 1 fault	Drives Prof.
7304	resolver 2 fault	Drives Prof.
7305	incremental sensor 1 fault	Drives Prof.
7306	incremental sensor 2 fault	Drives Prof.
7307	incremental sensor 3 fault	Drives Prof.
7310	speed	Drives Prof.
7320	position	Drives Prof.
7400	computation circuit	Drives Prof.
7500	communication	Drives Prof.
7510	serial interface No. 1	Drives Prof.
7520	serial interface No. 2	Drives Prof.
7600	data storage	Drives Prof.

Error Code (hex)	Meaning	Defined By
8000	monitoring	Comm. Prof.
8100	communication	Drives Prof.
8110	process data monitoring	Drives Prof.
8120	host monitoring	Drives Prof.
8200	control	Drives Prof.
8300	torque control	Drives Prof.
8311	excess torque	Drives Prof.
8312	difficult start up	Drives Prof.
8313	standstill torque	Drives Prof.
8321	insufficient torque	Drives Prof.
8331	torque fault	Drives Prof.
8400	velocity speed controller	Drives Prof.
8500	position controller	Drives Prof.
8600	positioning controller	Drives Prof.
8611	following error	Drives Prof.
8612	reference limit	Drives Prof.
8700	sync controller	Drives Prof.
8800	winding controller	Drives Prof.
9000	external error	Comm. Prof.
F000	additional functions	Comm. Prof.
F001	deceleration	Drives Prof.
F002	sub-synchronous run	Drives Prof.
F003	stroke operation	Drives Prof.
F004	control	Drives Prof.
FF00	device specific	Comm. Prof.
.....		
FFFF		

Table 3: Error Codes

7 Predefinitions

7.1 Naming conventions

The first three characters of a CMS name is a device profile identification (see /16/) and is defined for this device profile as: <402> (according to the number of this standard).

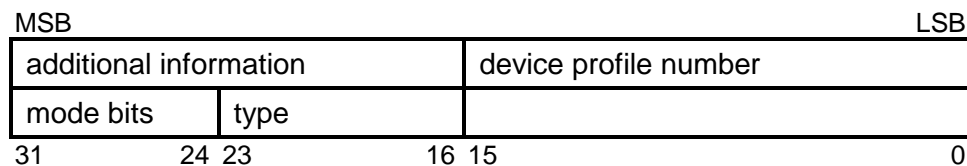
7.2 Predefined Objects

The default values for communication objects 1000_h to 1FFF_h which are not defined by the communication profile (see /16/) are mentioned below.

7.2.1 Object 1000_h: Device Type

The object at index 1000_h describes the type of a device and its functionality.

For multi device modules the additional information parameter contains FFF_h and the device profile number referenced by object 1000_h is the device profile of the first device in the object dictionary. All other devices of a multiple device module identify their profiles at object 67FF_h + x * 800_h with x = internal number of the device (0..7).



For devices in this device profile the following assignment exists:

Device Profile Number:	402
additional information	
drive type bit encoded: bits 16...23	Bit 16 = 1 Frequency Converter Bit 17 = 1 Servo Drive Bit 18 = 1 Stepper Motor Bit 23 = 1 I/O module (only multi device modules)
manufacturer specific bits 24...31	0

Table 4: Structure of the Device Type Entry in the Object Dictionary

7.2.2 Object 1001_h: Error Register

All bits are defined as in /16/. The device specific bit in the status word is used by the CANopen Device Profile for Drives and Motion Control. The error code can be read from the predefined error field at object 1003_h and to be compatible with device profiles for drives available for other fieldbus systems from object 603F_h as well.

7.2.3 Object 67FF_h: Single Device Type

The object at index 67FF_h and multiples with an offset of 800_h describe the type of each device within one drive unit and its functionality. The object structure is the same as defined in Object 1000_h.

For a multi device module, there must be defined the *device_type* at index 1000_h with the value FFFF_h instead.

7.3 PDO Mapping

A drive supporting more than one mode will mostly use more than one standard PDO. Therefore a lot of PDOs are predefined in respect to the different possible modes of operation for drives.

The hereafter described PDO distribution should be used for every axle of a multi-device module with an offset of 64, e.g. the first PDO of the second axle gets the number 65. In this way a system with a maximum of 8 axles is supported.

It is open to a manufacturer to specify additional entries in the mapping table or define absolutely new PDO mappings and it is also open to a user to change these default settings by changing the mapping structure, if the module supports variable mapping on these PDOs.

7.3.1 Receive PDOs

PDO No.	Mapping Object Index	Mapping Object Name	M/O	Comment
1	6040 _h	controlword	M	controls the state machine
2	6040 _h 6060 _h	controlword modes_of_operation	O	controls the state machine and mode of operation
3	6040 _h 607A _h	controlword target_position	O	controls the state machine and the target position (pp)
4	6040 _h 60FF _h	controlword target_velocity	O	controls the state machine and the target velocity (pv)
5	6040 _h 6071 _h	controlword target_torque	O	controls the state machine and the target torque (tq)
6	6040 _h 6042 _h	controlword vl_target_velocity	O	controls the state machine and the nominal speed (vl)
7	6040 _h 60FE _h	controlword digital_outputs	O	controls the state machine and the digital outputs
8	6040 _h 6060 _h	controlword modes_of_operation	O	controls the state machine and mode of operation (Broadcast PDO)
9 - 20			O	reserved
21 - 64			O	manufacturer specific

7.3.1.1 1st Receive PDO

Index	Subindex	Comment	Default Value
1400 _h	0	number of entries	see /16/
	1	COB-ID used by PDO	see /16/
	2	transmission type	255
	3	inhibit time	see /16/
	4	CMS priority group	3

Index	Subindex	Comment	Default Value
1600 _h	0	number of mapped objects	1
	1	controlword	60400010 _h

7.3.1.2 2nd Receive PDO

Index	Subindex	Comment	Default Value
1401 _h	0	number of entries	see /16/
	1	COB-ID used by PDO	see /16/
	2	transmission type	255
	3	inhibit time	see /16/
	4	CMS priority group	3

Index	Subindex	Comment	Default Value
1601 _h	0	number of mapped objects	2
	1	controlword	60400010 _h
	2	modes_of_operation	60600008 _h

7.3.1.3 3rd Receive PDO

Index	Subindex	Comment	Default Value
1402 _h	0	number of entries	see /16/
	1	COB-ID used by PDO	not defined
	2	transmission type	255
	3	inhibit time	see /16/
	4	CMS priority group	3

Index	Subindex	Comment	Default Value
1602 _h	0	number of mapped objects	2
	1	controlword	60400010 _h
	2	target_position	607A0020 _h

7.3.1.4 4th Receive PDO

Index	Subindex	Comment	Default Value
1403 _h	0	number of entries	see /17/
	1	COB-ID used by PDO	not defined
	2	transmission type	255
	3	inhibit time	see /16/
	4	CMS priority group	3

Index	Subindex	Comment	Default Value
1603 _h	0	number of mapped objects	2
	1	controlword	60400010 _h
	2	profile_velocity	60FF0020 _h

7.3.1.5 5th Receive PDO

Index	Subindex	Comment	Default Value
1404 _h	0	number of entries	see /16/
	1	COB-ID used by PDO	not defined
	2	transmission type	255
	3	inhibit time	see /16/
	4	CMS priority group	3

Index	Subindex	Comment	Default Value
1604 _h	0	number of mapped objects	2
	1	controlword	60400010 _h
	2	target_torque	60710010 _h

7.3.1.6 6th Receive PDO

Index	Subindex	Comment	Default Value
1405 _h	0	number of entries	see /16/
	1	COB-ID used by PDO	not defined
	2	transmission type	255
	3	inhibit time	see /16/
	4	CMS priority group	3

Index	Subindex	Comment	Default Value
1605 _h	0	number of mapped objects	2
	1	controlword	60400010 _h
	2	nominal_speed_value	60420010 _h

7.3.1.7 7th Receive PDO

Index	Subindex	Comment	Default Value
1406 _h	0	number of entries	see /16/
	1	COB-ID used by PDO	not defined
	2	transmission type	255
	3	inhibit time	see /16/
	4	CMS priority group	3

Index	Subindex	Comment	Default Value
1606 _h	0	number of mapped objects	2
	1	controlword	60400010 _h
	2	digital_outputs	60FE0020 _h

7.3.1.8 8th Receive PDO

Index	Subindex	Comment	Default Value
1407 _h	0	number of entries	see /16/
	1	COB-ID used by PDO	not defined
	2	transmission type	255
	3	inhibit time	see /16/
	4	CMS priority group	3

Index	Subindex	Comment	Default Value
1607 _h	0	number of mapped objects	2
	1	controlword	60400010 _h
	2	modes_of_operation	60600008 _h

7.3.2 Transmit PDOs

The task of the transmit PDOs is the monitoring of the drives behaviour. The TPDO 1,2 and 7 are event driven. The other PDOs can be implemented as synchronous or remotely requested (RTR) PDO.

PDO No.	Mapping Object Index	Mapping Object Name	M/O	Comment
1	6041 _h	statusword	M	shows status
2	6041 _h 6061 _h	statusword modes_of_operation_display	O	shows status and the actual mode of operation
3	6041 _h 6064 _h	statusword position_actual_value	O	shows the status and the actual position (pp)
4	6041 _h 606C _h	statusword velocity_actual_value	O	shows the status and the actual velocity (pv)
5	6041 _h 6077 _h	statusword torque_actual_value	O	shows the status and the actual torque (tq)
6	6041 _h 6044 _h	statusword vl_control_effort	O	shows the status and the actual speed (vl)
7	6041 _h 60FD _h	statusword digital_inputs	O	shows the status and the digital inputs
8 – 20			O	reserved
21 - 64			O	manufacturer specific

7.3.2.1 1st Transmit PDO

Index	Subindex	Comment	Default Value
1800 _h	0	number of entries	see /16/
	1	COB-ID used by PDO	see /16/
	2	transmission type	255
	3	inhibit time	see /16/
	4	CMS priority group	3

Index	Subindex	Comment	Default Value
1A00 _h	0	number of mapped objects	1
	1	statusword	60410010 _h

7.3.2.2 2nd Transmit PDO

Index	Subindex	Comment	Default Value
1801 _h	0	number of entries	see /16/
	1	COB-ID used by PDO	see /16/
	2	transmission type	255
	3	inhibit time	see /16/
	4	CMS priority group	3

Index	Subindex	Comment	Default Value
1A01 _h	0	number of mapped objects	2
	1	statusword	60410010 _h
	2	modes_of_operation_display	60610008 _h

7.3.2.3 3rd Transmit PDO

Index	Subindex	Comment	Default Value
1802 _h	0	number of entries	see /16/
	1	COB-ID used by PDO	not defined
	2	transmission type	not defined
	3	inhibit time	see /16/
	4	CMS priority group	3

Index	Subindex	Comment	Default Value
1A02 _h	0	number of mapped objects	2
	1	statusword	60410010 _h
	2	position_actual_value	606400020 _h

7.3.2.4 4th Transmit PDO

Index	Subindex	Comment	Default Value
1803 _h	0	number of entries	see /16/
	1	COB-ID used by PDO	not defined
	2	transmission type	not defined
	3	inhibit time	see /16/
	4	CMS priority group	3

Index	Subindex	Comment	Default Value
1A03 _h	0	number of mapped objects	2
	1	statusword	60410010 _h
	2	velocity_actual_value	606C0020 _h

7.3.2.5 5th Transmit PDO

Index	Subindex	Comment	Default Value
1804 _h	0	number of entries	see /16/
	1	COB-ID used by PDO	not defined
	2	transmission type	not defined
	3	inhibit time	see /16/
	4	CMS priority group	3

Index	Subindex	Comment	Default Value
1A04 _h	0	number of mapped objects	2
	1	statusword	60410010 _h
	2	torque_actual_value	60770010 _h

7.3.2.6 6th Transmit PDO

Index	Subindex	Comment	Default Value
1805 _h	0	number of entries	see /16/
	1	COB-ID used by PDO	not defined
	2	transmission type	not defined
	3	inhibit time	see /16/
	4	CMS priority group	3

Index	Subindex	Comment	Default Value
1A05 _h	0	number of mapped objects	2
	1	statusword	60410010 _h
	2	vl_control_effort	60440010 _h

7.3.2.7 7th Transmit PDO

Index	Subindex	Comment	Default Value
1806 _h	0	number of entries	see /16/
	1	COB-ID used by PDO	not defined
	2	transmission type	255
	3	inhibit time	see /16/
	4	CMS priority group	3

Index	Subindex	Comment	Default Value
1A06 _h	0	number of mapped objects	2
	1	statusword	60410010 _h
	2	digital_inputs	60FD0020 _h

8 Object Dictionary

Each drive shares the dictionary entries from 6000_h to 63FF_h. These entries are common to all drive modules and each module implements only the dictionary parts which are relevant for its functions.

Drives having also digital or analog I/O are using dictionary entries from 8000_h to 83FF_h as described in /19/ for the objects from 6000_h to 63FF_h with an offset of 2000_h.

Meaning of the Table Rows:	
Index	the 16-bit index to the object dictionary used by a module to represent a special function, data or task
Name	short description of the usage
Object Code	object type which represents the data, e.g. VAR, ARRAY, RECORD, etc.
Data Type	data type which represents the information, e.g. Unsigned32, Unsigned8 etc.
Object Class	entries in this row indicates whether an object is mandatory or not: M this object is mandatory for all drives O this object is optional Dependent on the mode of operation
Access	description how the object might be accessed: ro read only wo write only rw read and write
PDO mapping	indicates the manner of PDO mapping for an object. No mapping is not allowed Possible mapping is allowed for the manufacturer Yes this object is mapped by default
Units	physical units of the object value
Value Range	the value range allowed and requested for an object
Default Value	default value of the object after device initialization.
Substitute Value	if the object doesn't exist in the object dictionary description, this value will be used for internal calculations
Device Mode Abbreviations:	
pp	mandatory (m), optional (o) or not used (-) for the Profile Position Mode
pv	mandatory (m), optional (o) or not used (-) for the Profile Velocity Mode
vl	mandatory (m), optional (o) or not used (-) for the Velocity Mode
hm	mandatory (m), optional (o) or not used (-) for the Homing Mode
ip	mandatory (m), optional (o) or not used (-) for the Interpolated Position Mode
tq	mandatory (m), optional (o) or not used (-) for the Profile Torque Mode
all	mandatory for all modes
Chapter Title Abbreviations	
ce	Common Entries in the Object Dictionary
dc	Device Control
pc	mandatory (m), optional (o) or not used (-) for the Position Control Function

9 Common Entries in the Object Dictionary

9.1 General Information

9.1.1 Motor Data

The objects 6402_h to 64FF_h serve as a database for motor parameters. The values are typically found on the motor's nameplate or the manufacturer's motor catalog and are used to maintain a service database within the controlling device of the drive. Most of the entries are typically entities from the manufacturer's motor catalog. Future drives should at least contain an entry to the electronically available catalog via a common net address, like a HTTP link to the manufacturers database, *http_motor_catalog_address*.

The objects 6402_h to 640F_h are highly recommended.

Some objects are available in the object dictionary of other fieldbus systems, so their indices are not in the default range from 6400_h to 64ff_h.

There is one manufacturer specific data RECORD at object 6410_h. It should contain as much as possible entries for the used motor. The structure of this record is described in the manufacturer's data sheet for the drive unit.

9.1.2 Drive Data

The objects 6500_h to 65FF_h serve as a database for drive parameters.

There is one manufacturer specific data RECORD at object 6510_h. It should contain as much as possible entries for the used drive. The structure of this record is described in the manufacturer's handbook. The data must be filled in while in commissioning. The values are typically found on the drive's datasheet or the manufacturer's drives catalog and are used to maintain a service database within the controlling device of the drive.

Most of the entries are typically entities from the manufacturer's drive catalog. Future drives should at least contain an entry to the electronically available catalog via a common net address, like a HTTP link to the manufacturers database, *http_drive_catalog_address*.

In /16/ three optional objects for a CANopen device are recommended:

Index	Name
1008 _h	manufacturer device name
1009 _h	manufacturer hardware version
100A _h	manufacturer software version

9.2 Object Dictionary Entries

9.2.1 Objects defined in this Chapter

Index	Object	Name	Type	Attr.	M/O
6007 _h	VAR	abort_connection_option_code	Integer16	rw	O
603F _h	VAR	error_code	Unsigned16	ro	O
6402 _h	VAR	motor_type	Unsigned16	rw	O
6403 _h	VAR	motor_catalogue_number	Visible String	rw	O
6404 _h	VAR	motor_manufacturer	Visible String	rw	O
6405 _h	VAR	http_motor_catalog_address	Visible String	rw	O
6406 _h	VAR	motor_calibration_date	Date	rw	O
6407 _h	VAR	motor_service_period	Unsigned32	rw	O
6410 _h	RECORD	motor_data	-	rw	O
6502 _h	VAR	supported_drive_modes	Unsigned32	ro	O
6503 _h	VAR	drive_catalogue_number	Visible String	ro	O
6504 _h	VAR	drive_manufacturer	Visible String	ro	O
6505 _h	VAR	http_drive_catalog_address	Visible String	rw	O
6510 _h	RECORD	drive_data	-	rw	O
60FD _h	VAR	digital_inputs	Unsigned32	rw	O
60FE _h	RECORD	digital_outputs	-	rw	O

9.3 Object Description

9.3.1 Object 6007_h: abort_connection_option_code

The content of this object selects the function to be performed when the connection to the network is lost.

Index	6007_h
Name	abort_connection_option_code
Object Code	VAR
Data type	Integer16

Value description

Object Class	M: -	O: all
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	-32768..32767	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	

Data Description

Option Code	Meaning of the Option Code
0	No action
1	Malfunction
2	Device control command „disable_voltage“
3	Device control command „quick_stop“
4..32767	reserved
-32768..-1	manufacturer specific

9.3.2 Object 603F_h: error_code

The *error_code* captures the code of the last error that occurred in the drive. It corresponds to the value of the lower 16 bits of object 1003_h *pre_defined_error_field*.

Index	603F_h
Name	error_code
Object Code	VAR
Data type	Unsigned16

Value description

Object Class	M: -	O: all
Access	ro	
PDO Mapping	Possible	
Units	-	
Value Range	0..65535	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	

9.3.3 Object 6402_h: motor_type

The type of motor driven by the controller.

Index	6402_h
Name	motor_type
Object Code	VAR
Data type	Unsigned16

Value Description

Object Class	M: -	O: -
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..65535	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

Data Description

Value	Motor Type
0	Non-Standard Motor
1	Phase Modulated DC Motor
2	Frequency Controlled DC Motor
3	PM Synchronous Motor
4	FC Synchronous Motor
5	Switched Reluctance Motor
6	Wound Rotor Induction Motor
7	Squirrel Cage Induction Motor
8	Stepper Motor
9	Micro-Step Stepper Motor
10	Sinusoidal PM BL Motor
11	Trapezoidal PM BL Motor

9.3.4 Object 6403_h: motor_catalogue_number

The manufacturer's motor catalog number (nameplate number).

Index	6403_h
Name	motor_catalogue_number
Object Code	VAR
Data type	Visible String

Value Description

Object Class	M: -	O: -
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	-	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

9.3.5 Object 6404_h: motor_manufacturer

The motor manufacturer's name.

Index	6404_h
Name	motor_manufacturer
Object Code	VAR
Data type	Visible String

Value Description

Object Class	M: -	O: -
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	-	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

9.3.6 Object 6405_h: http_motor_catalog_address

Index	6405_h
Name	http_motor_catalog_address
Object Code	VAR
Data type	Visible String

Value Description

Object Class	M: -	O: -
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	-	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

9.3.7 Object 6406_h: motor_calibration_date

Date of the last motorobject 6406_h inspection.

Index	6406 _h
Name	motor_calibration_date
Object Code	VAR
Data type	Date

Value Description

Object Class	M: -	O: -
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	-	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

9.3.8 Object 6407_h: motor_service_period

Value in hours of the nominal motor lifetime. The motor needs service after this time.

Index	6407 _h
Name	motor_service_period
Object Code	VAR
Data type	Unsigned32

Value Description

Object Class	M: -	O: -
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

9.3.9 Object 6410_h: motor_data

This object contains as much as possible information about the connected motor. The structure of this record is described in the drive manufacturer's handbook.

Index	6410_h
Name	motor_data
Object Code	RECORD
Number of Elements	0..255

Value Description

Sub-Index	01 _h ..255 _h	
Description	Manufacturer specific	
Object Class	M: -	O: all
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	-	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	
Data Type	manufacturer defined	

9.3.10 Object 6502_h: supported_drive_modes

A drive can support more then one and several distinct modes of operation. Many of them are described in this document. This object is read only.

Index	6502_h
Name	supported_drive_modes
Object Code	VAR
Data type	Unsigned32

Value Description

Object Class	M: -	O: -
Access	ro	
PDO Mapping	Possible	
Units	-	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

Data Description

Bit number	Description
0	Profile Position Mode
1	Velocity Mode
2	Profile Velocity Mode
3	Profile Torque Mode
4	reserved
5	Homing Mode
6	Interpolated Position Mode
7	reserved
8	reserved
9	reserved
10...15	reserved
16 ... 31	manufacturer specific

9.3.11 Object 6503_h: drive_catalogue_number

The manufacturer's drive catalog number (nameplate number).

Index	6503_h
Name	drive_catalogue_number
Object Code	VAR
Data type	Visible String

Value Description

Object Class	M: -	O: -
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	-	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

9.3.12 Object 6504_h: drive_manufacturer

The drive manufacturer's name.

Index	6504_h
Name	drive_manufacturer
Object Code	VAR
Data type	Visible String

Value Description

Object Class	M: -	O: -
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	-	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

9.3.13 Object 6505_h: http_drive_catalog_address

The internet address of the manufacturer.

Index	6505_h
Name	http_drive_catalog_address
Object Code	VAR
Data type	Visible String

Value Description

Object Class	M: -	O: -
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	-	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

9.3.14 Object 6510_h: drive_data

This object contains as much as possible information about the drive unit. The structure of this record is described in the drive manufacturer's handbook.

Index	6510_h
Name	drive_data
Object Code	RECORD
Number of Elements	0..255

Value Description

Sub-Index	01 _h ..255 _h	
Description	Manufacturer specific	
Object Class	M: -	O: all
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	-	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	
Data Type	manufacturer defined	

9.3.15 Object 60FD_h: digital_inputs

This index defines simple digital inputs for drives. The user may apply any signals to these inputs for special purposes like limit or reference switches.

Index	60FD _h
Name	digital_inputs
Object Code	VAR
Data type	Unsigned32

Value Description

Object Class	M: -	O: all
Access	ro	
PDO Mapping	Possible	
Units	-	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	

Data Description

Bit No	Digital Input Assignment
0	negative limit switch position switch is "active high"
1	positive limit switch position switch is "active high"
2	home switch
3	interlock (enable) switch is "active high"
4..15	reserved
16..31	manufacturer specific

9.3.16 Object 60FE_h: digital_outputs

This index defines simple digital outputs for drives.

Index	60FE_h
Name	digital_outputs
Object Code	RECORD
Number of Elements	2

Value Description

Sub-Index	01 _h	
Description	physical_outputs	
Object Class	M: -	O: all
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	
Data Type	Unsigned32	

Data Description

Bit No	Assigned Digital Outputs
0	set brake
1..15	reserved
16..31	manufacturer specific

This second sub-index describes a mask to specify which of the outputs shall be used, where a "1" selects and a "0" deselects an output.

Sub-Index	02 _h	
Description	bitmask	
Object Class	M: -	O: all
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0, 1	
Mandatory Range	0, 1	
Default Value	0	
Substitute Value	-	
Data Type	Unsigned32	

10 Device Control

10.1 General Information

The device control function block controls all functions of the drive (drive function and power section). It is divided into:

Device Control of the Statemachine

Operation Mode Function

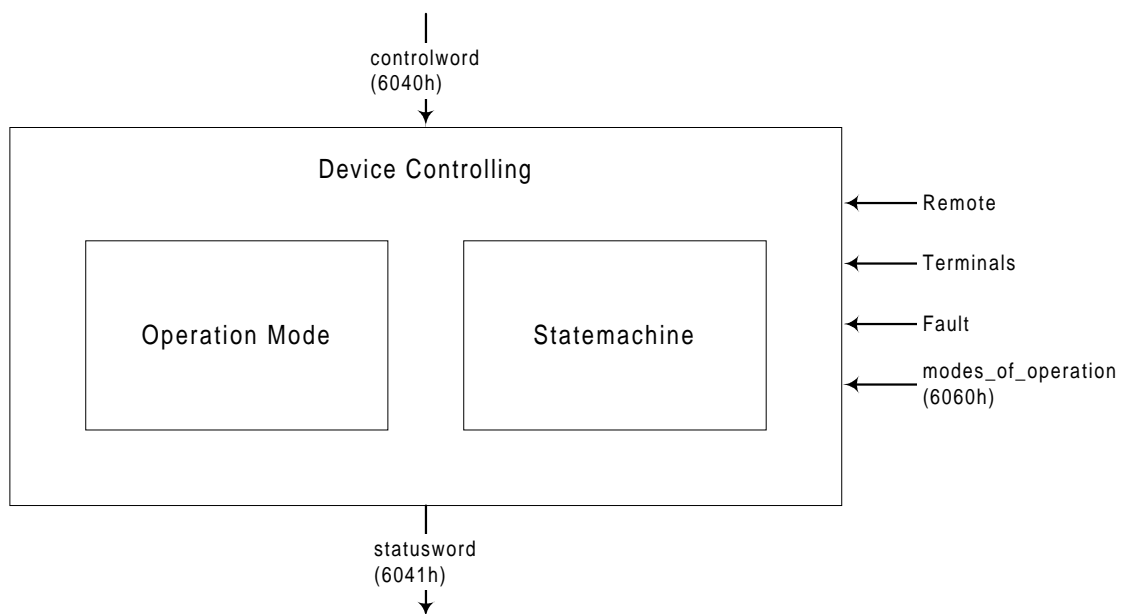


Figure 5: Device-Controlling

The state of the drive can be controlled by the *controlword*

The state of the drive is shown in the *statusword*

In remote mode the device is controlled directly from the CAN-network by Process Data Objects (PDOs) and Service Data Objects (SDOs).

The statemachine is controlled externally by the *controlword* and external signals. The write access to the *controlword* is controlled by the optional hardware signal 'Remote'. The statemachine is also controlled by internal signals like faults and *modes_of_operation*.

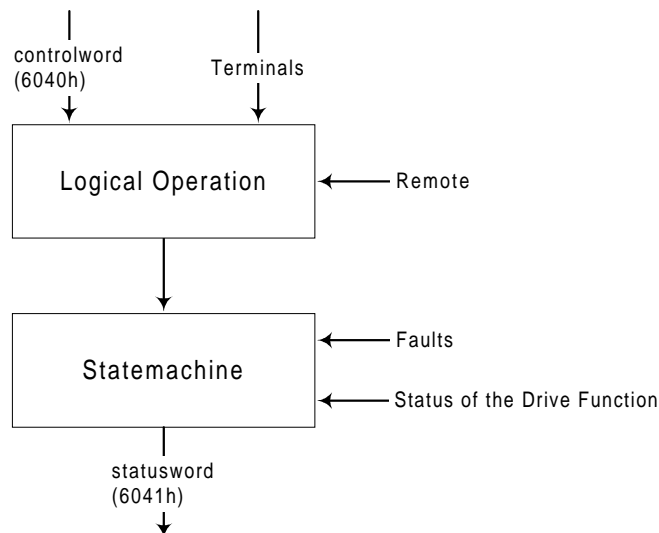


Figure 6: Remote Mode

10.1.1 Statemachine

The Statemachine describes the device status and the possible control sequence of the drive. A single state represents a special internal or external behaviour. The state of the drive also determines which commands are accepted. E.g. it is only possible to start a point-to-point move when the drive is in state OPERATION ENABLED.

States may be changed using the *controlword* and/or according to internal events. The current state can be read using the statusword.

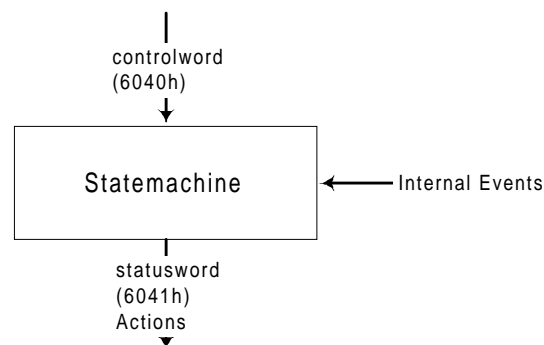


Figure 7: Statemachine

The state diagram in Figure 7 describes the state machine of the device with respect to control of the power electronics as a result of user commands and internal drive faults.

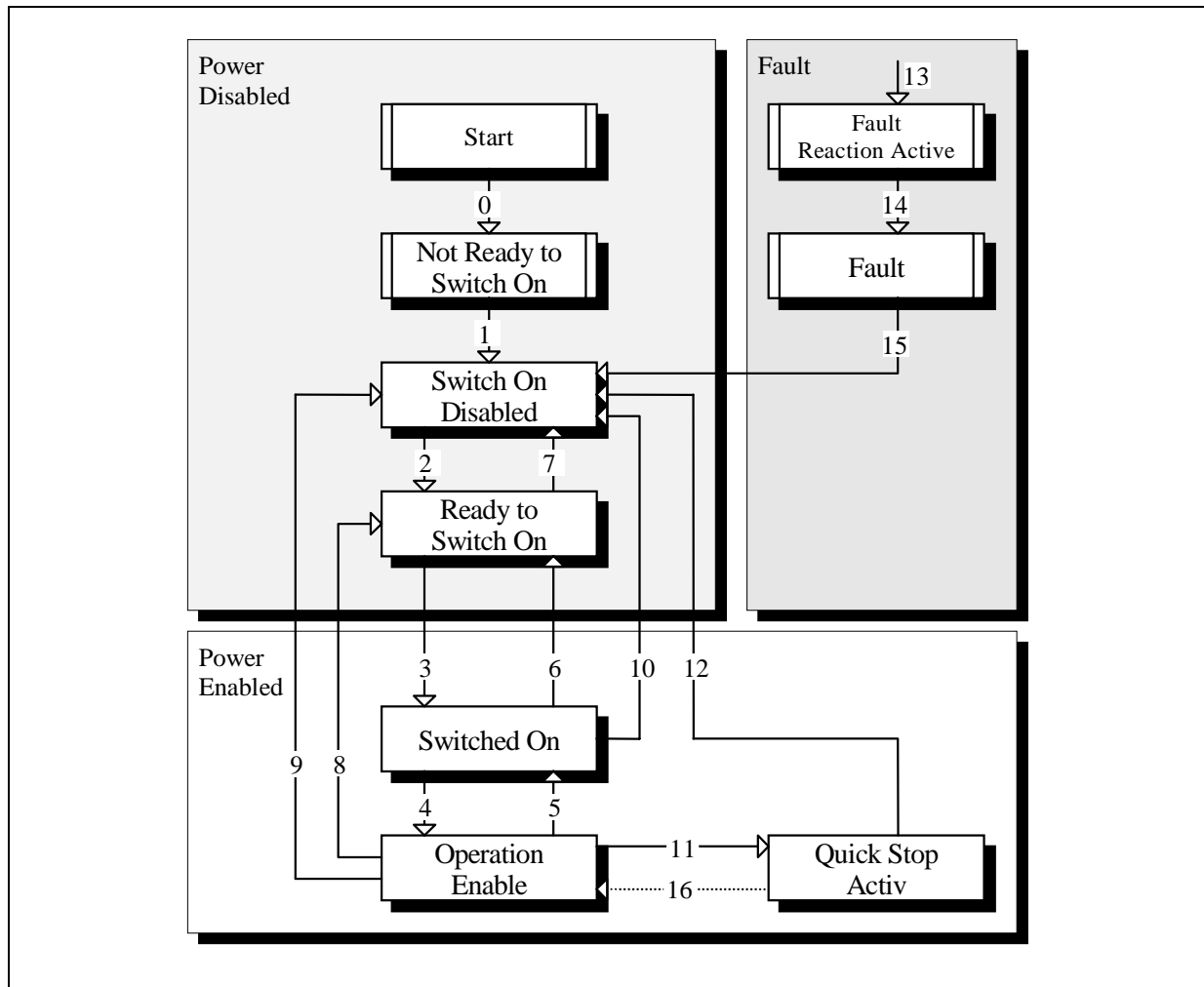


Figure 8: State Diagram

10.1.1.1 Drive States

The drive states may become more evident when considering the following (generic) block diagram of a drive:

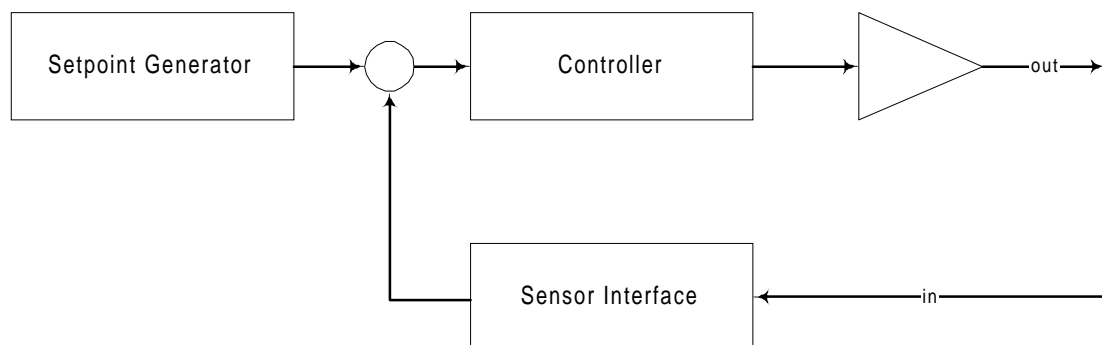


Figure 9: Generic Control Loop Block Diagram

The sensor interface and the “in”-terminal are only present in drives with a feedback path. Normally the setpoint generator, the controller and the power amplifier can be disabled.

The following states of the device are possible:

- *Not Ready to Switch On:*
Low level Power (e.g. $\pm 15\text{V}$, 5V) has been applied to the drive.
The drive is being initialized or is running self test.
A brake, if present, has to be applied in this state.
The drive function is disabled.
- *Switch On Disabled:*
Drive Initialisation is complete.
The drive parameters have been set up.
Drive parameters may be changed.
High Voltage may not be applied to the drive, (e.g. for safety reasons).
The drive function is disabled.
- *Ready to Switch On:*
High Voltage may be applied to the drive.
The drive parameters may be changed.
The drive function is disabled.
- *Switched On:*
High Voltage has been applied to the drive.
The Power Amplifier is ready.
The drive parameters may be changed.
The drive function is disabled.
- *Operation Enable:*
No faults have been detected.
The drive function is enabled and power is applied to the motor.
The drive parameters may be changed.
(This corresponds to normal operation of the drive.)
- *Quick Stop Active:*
The drive parameters may be changed.
The Quick Stop function is being executed.
The drive function is enabled and power is applied to the motor.

If the ‘Quick-Stop-Option-Code’ is switched to 5 (Stay in Quick-Stop), you can’t leave the Quick-Stop-State, but you can transmit to ‘Operation Enable’ with the command ‘Enable Operation’.
- *Fault Reaction Active:*
The drive parameters may be changed.
A non-fatal fault has occurred in the drive.
The Quick Stop function is being executed.
The drive function is enabled and power is applied to the motor.
- *Fault:*
The drive parameters may be changed.
A fault has occurred in the drive.
The drive function is disabled.

10.1.1.2 State Transitions of the Drive Supervisor

State Transitions are caused by internal events in the drive or by commands from the host via the controlword.

- *State Transition 0: Startup ⇒ Not Ready to Switch On*
Event: Reset.
Action: The drive self-tests and/or self-initialises.
- *State Transition 1: Not Ready to Switch On ⇒ Switch On Disabled*
Event: The drive has self-tested and/or initialised successfully.
Action: Activate communication and process data monitoring
- *State Transition 2: Switch On Disabled ⇒ Ready to Switch On*
Event: 'Shutdown' command received from host.
Action: None
- *State Transition 3: Ready to Switch On ⇒ Switched On*
Event: 'Switch On' command received from host.
Action: The power section is switched on if it is not already switched on.
- *State Transition 4: Switched On ⇒ Operation Enabled*
Event: 'Enable Operation' command received from host.
Action: The drive function is enabled.
- *State Transition 5: Operation Enabled ⇒ Switched On*
Event: 'Disable Operation' command received from host.
Action: The drive operation will be disabled.
- *State Transition 6: Switched On ⇒ Ready to Switch On*
Event: 'Shutdown' command received from host.
Action: The power section is switched off.
- *State Transition 7: Ready to Switch On ⇒ Switch On Disable*
Event: 'Quick stop' command received from host.
Action: None
- *State Transition 8: Operation Enable ⇒ Ready to Switch On*
Event: 'Shutdown' command received from host.
Action: The power section is switched off immediately, and the motor is free to rotate if unbraked
- *State Transition 9: Operation Enable ⇒ Switch On Disable*
Event: 'Disable Voltage' command received from host.
Action: The power section is switched off immediately, and the motor is free to rotate if unbraked
- *State Transition 10: Switched On ⇒ Switched On Disable*
Event: 'Disable Voltage' or 'Quick Stop' command received from host.
Action: The power section is switched off immediately, and the motor is free to rotate if unbraked
- *State Transition 11: Operation Enabled ⇒ Quick Stop Active*
Event: 'Quick Stop' command received from host.
Action: The Quick Stop function is executed.

- *State Transition 12: Quick Stop Active \Rightarrow Switch On Disabled*
Event: 'Quick Stop' is completed or 'Disable Voltage' command received from host.
This transition is possible, if the Quick-Stop-Option-Code is different 5 (Stay in Quick-Stop)
Action: The power section is switched off.
- *State Transition 13: All states \Rightarrow Fault Reaction Active*
A fatal fault has occurred in the drive.
Action: Execute appropriate fault reaction.
- *State Transition 14: Fault Reaction Active \Rightarrow Fault*
Event: The fault reaction is completed.
Action: The drive function is disabled. The power section may be switched off.
- *State Transition 15: Fault \Rightarrow Switch On Disabled*
Event: 'Fault Reset' command received from host.
Action: A reset of the fault condition is carried out if no fault exists currently on the drive.
After leaving the 'Fault' state the Bit 'Fault Reset' of the controlword has to be cleared by the host.
- *State Transition 16: Quick Stop Active \Rightarrow Operation Enable*
Event: 'Enable Operation' command received from host. This transition is possible if the Quick-Stop-Option-Code is 5, 6, 7 or 8 (\rightarrow Chapter 10.3.5).
Action: The drive function is enabled.

Notes:

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

'Drive function is disabled' implies no energy is supplied to the motor. This may be achieved by different manufacturers in different ways. Reference values are not processed.

'Drive function is enabled' implies that energy can be supplied to the motor. The reference values (Torque, Velocity, Position) are processed.

'Fault occurred' implies that a fault in the drive has occurred. In this case there is a transition to the state 'Fault Reaction Active'. In this state the device will execute a special fault reaction. After the execution of this fault reaction the device will switch to the state 'Fault'. This state can only be left by the command 'Fault reset', but only if the fault is not active any more.

10.2 Object Dictionary Entries

10.2.1 Objects defined in this Chapter

Index	Object	Name	Type	Attr.	M/O
6040 _h	VAR	controlword	Unsigned16	rw	M
6041 _h	VAR	statusword	Unsigned16	rw	M
605B _h	VAR	shutdown_option_code	Integer16	rw	O
605C _h	VAR	disable_operation_option_code	Integer16	rw	O
605A _h	VAR	quick_stop_option_code	Integer16	rw	O
605D _h	VAR	stop_option_code	Integer16	rw	O
605E _h	VAR	fault_reaction_option_code	Integer16	rw	O
6060 _h	VAR	modes_of_operation	Integer8	wo	M
6061 _h	VAR	modes_of_operation_display	Integer8	ro	M

10.3 Object Description

10.3.1 Object 6040_h: controlword

The logical addition of several bits in the *controlword* and the external signals (transitions) results in the device-control-command. The *controlword* is always mapped into the first two bytes of the drive's command Message. The bits of the *controlword* are defined as follows:

Index	6040_h
Name	controlword
Object Code	VAR
Data Type	Unsigned16

Value Description

Object Class	M: all	O: -I
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..65535	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

Data Description

MSB								LSB							
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
High-Byte								Low-Byte							

Bit	Name	Mandatory
0	Switch On	√
1	Disable Voltage	√
2	Quick Stop	√
3	Enable Operation	√
4	Operation Mode Specific	
5	Operation Mode Specific	
6	Operation Mode Specific	
7	Reset Fault	√
8	Halt	
9	Reserved	
10	Reserved	
11	Manufacturer Specific	
12	Manufacturer Specific	
13	Manufacturer Specific	
14	Manufacturer Specific	
15	Manufacturer Specific	

Table 5 **Bits in the *controlword***

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


command/ Bit of the <i>controlword</i>	Bit 7 Fault Reset	Bit 3 Enable Operation	Bit 2 Quick Stop	Bit 1 Disable Voltage	Bit 0 Switch On	Transitions
Shutdown	0	X	1	1	0	2,6,8
Switch On	0	X	1	1	1	3
Disable Voltage	0	X	X	0	X	7,9,10,12
Quick Stop	0	X	0	1	X	7,10,11
Disable Operation	0	0	1	1	1	5
Enable Operation	0	1	1	1	1	4,16
Fault Reset		X	X	X	X	15

Table 6: Device Control Commands

10.3.1.1 Description of the remaining Bits of the *controlword*

Bits 4, 5 and 6 are operation mode specific

Bit	Operation Mode					
	Velocity Mode	Profile Position Mode	Profile Velocity Mode	Profile Torque Mode	Homing Mode	Interpol. Position Mode
4	RFG disable	new_set- point	reserved	reserved	Homing Operation Start	enable_ip_ mode
5	RFG stop	change_set _immediatly	reserved	reserved	reserved	reserved
6	RFG zero	0: absolute 1: relative	reserved	reserved	reserved	reserved
8	Halt	Halt	Halt	Halt	Halt	Halt

Table 7: Mode specific Bits in the *controlword*

RFG: **R**unning up **F**requency **G**enerator (see chapter 18, Velocity Mode)

Halt: Interrupts the move of a drive, and wait for release to continue.

Bits 9, 10 are reserved

These bits are reserved for further use. They are inactive by setting to zero. If they have no special function, they must be set to zero.

Bits 11, 12, 13, 14 and 15 are manufacturer specific

10.3.2 Object 6041_h: statusword

The *statusword* indicates the current status of the drive and is always mapped into the first two bytes of the actual message. The following bits are defined in the statusword.

Index	6041_h
Name	statusword
Object Code	VAR
Data Type	Unsigned16

Value Description

Object Class	M: all O: -
Access	ro
PDO Mapping	Possible
Units	-
Value Range	0..65535
Mandatory Range	-
Default Value	-
Substitute Value	-

Data Description

MSB								LSB							
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
High-Byte								Low-Byte							

Bit	Name	Mandatory
0	Ready to Switch On	√
1	Switched On	√
2	Operation Enabled	√
3	Fault	√
4	Voltage Disabled	√
5	Quick Stop	√
6	Switch On Disabled	√
7	Warning	
8	Manufacturer Specific	
9	Remote	√
10	Target Reached	√
11	Internal Limit Active	√
12	Operation Mode Specific	
13	Operation Mode Specific	
14	Manufacturer Specific	
15	Manufacturer Specific	

Table 8: Bits in the *statusword*

Device Status Bit Meanings

The following bits indicate the status of the device:

State	Bit 6 Switch On Disable	Bit 5 Quick Stop	Bit 3 Fault	Bit 2 Operation Enable	Bit 1 Switched On	Bit 0 Ready to Switch On
Not Ready to Switch On	0	X	0	0	0	0
Switch On Disabled	1	X	0	0	0	0
Ready to Switch On	0	1	0	0	0	1
Switched On	0	1	0	0	1	1
Operation Enabled	0	1	0	1	1	1
Fault	0	X	1	1	1	1
Fault Reaction Active	0	X	1	1	1	1
Quick Stop Active	0	0	0	1	1	1

Table 9: Device State Bits

Bits marked X are irrelevant for that state. Other bit combinations are not allowed.

10.3.2.1 Description of the remaining Bits of the *statusword*

Bit 4: *voltage_disable*

The Disable Voltage request is active when the *voltage_disabled* bit is cleared to 0.

Bit 5: *quick_stop*

When reset, this bit indicates that the drive is reacting on a quick stop request. Bits 0, 1 and 2 of the *statusword* must be set to 1 to indicate that the drive is capable to regenerate. The setting of the other bits indicates the status of the drive (e.g. the drive is performing a quick stop as result of a reaction to a non-fatal fault. The fault bit is set as well as bits 0, 1 and 2).

Bit 7: *warning*

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

Bit 8 is manufacturer specific

This bit may be used by a drive manufacturer to implement any manufacturer specific functionality.

Bit 9: *remote*

If bit 9 is set, then parameters may be modified via the CAN-network, and the drive executes the content of a command message. If the bit *remote* is reset, then the drive is in local mode and will not execute the command message. The drive may transmit messages containing valid actual values like a *position_actual_value*, depending on the actual drive configuration. The drive will accept accesses via service data objects (SDOs) in local mode.

Bit 10: *target_reached*

If bit 10 is set by the drive, then a setpoint has been reached (torque, speed or position depending on the *modes_of_operation*). The change of a target value by software alters this bit.

If *quickstop_option_code* is 5, 6, 7 or 8, this bit must be set, when the quick stop operation is finished and the drive is halted.

If Halt occurred and the drive has halted then this bit is set too.

Bit 11: *internal_limit_active*

This bit set by the drive indicates, that an internal limitation is active (e.g. *position_range_limit*).

Bit 12, 13 are operation mode specific

Bit	Velocity mode	Profile Position Mode	Profile Velocity mode	Profile Torque mode	Homing mode	Interpol. Position mode
12	reserved	setpoint acknowledge	Speed = 0	reserved	Homing attained	lp-Mode active
13	reserved	following error	max_slippage error	reserved	Homing error	reserved

Table 10: Mode specific bits in the *statusword***Bit 14, 15 are manufacturer specific**

These bits may be used by a drive manufacturer to implement any manufacturer specific functionality.

Note:

All bits reflect the actual/current state of the drive. No bits are latched.

10.3.3 Object 605B_h: shutdown_option_code

The parameter *shutdown_option_code* determines what action should be taken if there is a transition 'OPERATION ENABLE' ⇒ 'READY TO SWITCH ON'

Index	605B_h
Name	shutdown_option_code
Object Code	VAR
Data Type	Integer16

Value Description

Object Class	M: - O: all
Access	rw
PDO Mapping	No
Units	-
Value Range	-32768..32767
Mandatory Range	Profile Specific Code
Default Value	0
Substitute Value	-

Data description

<i>reset_option_code</i>	Action
-32768 ... -1	Manufacturer Specific
0	Disable drive function
1	Slow down with slow down ramp disable of the drive function
2 ... 32767	reserved

10.3.4 Object 605C_h: disable_operation_option_code

The parameter *disable_operation_option_code* determines what action should be taken if there is a transition 'OPERATION ENABLE' ⇒ 'SWITCHED ON'.

Index	605C _h
Name	disable_operation_option_code
Object Code	VAR
Data Type	Integer16

Value Description

Object Class	M: - O: all
Access	rw
PDO Mapping	No
Units	-
Value Range	-32768..32767
Mandatory Range	-
Default Value	1
Substitute Value	-

Data description

<i>disable_operation_option_code</i>	Action
-32768 ... -1	Manufacturer Specific
0	disable drive function
1	Slow down with slow down ramp and then disabling of the drive function
2 ... 32767	reserved

10.3.5 Object 605A_h: quick_stop_option_code

The parameter *quick_stop_option_code* determines what action should be taken if the Quick Stop Function is executed.

Index	605A_h
Name	quick_stop_option_code
Object Code	VAR
Data Type	Integer16

Value Description

Object Class	M: - O: all
Access	rw
PDO Mapping	No
Units	-
Value Range	-32768..32767
Mandatory Range	-
Default Value	2
Substitute Value	-

Data description

<i>quick_stop_option_code</i>	Action
-32768 ... -1	Manufacturer Specific
0	Disable drive function
1	Slow down on slow down ramp
2	Slow down on quick stop ramp
3	Slow down on the current limit
4	Slow down on the voltage limit
5	Slow down on slow down ramp and stay in Quick-Stop
6	Slow down on quick stop ramp and stay in Quick-Stop
7	Slow down on the current limit and stay in Quick-Stop
8	Slow down on the voltage limit and stay in Quick-Stop
9 ... 32767	reserved

10.3.6 Object 605D_h: stop_option_code

The parameter *stop_option_code* determines what action should be taken if the Stop Function is active.

Index	605D_h
Name	stop_option_code
Object Code	VAR
Data Type	Integer16

Value Description

Object Class	M: - O: all
Access	rw
PDO Mapping	No
Units	-
Value Range	-32768..32767
Mandatory Range	-
Default Value	1
Substitute Value	-

Data description

<i>stop_option_code</i>	Action
-32768 ... -1	Manufacturer Specific
0	Disable drive, Motor is free to rotate
1	Slow down on slow down ramp
2	Slow down on quick stop ramp
3	Slow down on the current limit
4	Slow down on the voltage limit
5 ... 32767	reserved

10.3.7 Object 605E_h: fault_reaction_option_code

The parameter *fault_reaction_option_code* determines what action should be taken if a fault occurs in the drive.

Index	605E_h
Name	fault_reaction_option_code
Object Code	VAR
Data Type	Integer16

Value Description

Object Class	M: - O: all
Access	rw
PDO Mapping	No
Units	-
Value Range	-32768..32767
Mandatory Range	-
Default Value	2
Substitute Value	-

Data description

<i>fault_reaction_option_code</i>	Action
-32768 ... -1	Manufacturer specific
0	Disable drive, motor is free to rotate
1	Slow down on slow down ramp
2	Slow down on quick stop ramp
3	Slow down on the current limit
4	Slow down on the voltage limit
5 ... 32767	reserved

10.3.8 Object 6060_h: modes_of_operation

The parameter *modes_of_operation* switches the actually choosen operation-mode.

Index	6060_h
Name	modes_of_operation
Object Code	VAR
Data Type	Integer8

Value Description

Object Class	M: all	O: -
Access	wo	
PDO Mapping	Possible	
Units	-	
Value Range	-128..127	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

Data description

Modes of Operation	Action
-1 ... -128	Manufacturer specific modes of operation
0	reserved
1	Profile Position Mode
2	Velocity Mode
3	Profile Velocity Mode
4	Torque Profile Mode
5	reserved
6	Homing Mode
7	Interpolated Position Mode
8 ... 127	reserved

Note:

The actual mode is reflected in the *modes_of_operation_display* (index 6061_h), and not in the *modes_of_operation* (index 6060_h). It may be changed by writing to *modes_of_operation*

10.3.9 Object 6061_h: modes_of_operation_display

The *modes_of_operation_display* shows the current mode of operation. The meaning of the returned value corresponds to that of the *modes_of_operation* option code (index 6060_h)

Index	6061 _h
Name	modes_of_operation_display
Object Code	VAR
Data Type	Integer8

Value Description

Object Class	M: all	O: -
Access	ro	
PDO Mapping	Possible	
Units	-	
Value Range	-128..127	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

Data description

Same as for Object 6060_h *modes_of_operation*.

Note:

The actual mode is reflected in the *modes_of_operation_display* (index 6061_h), and not in the *modes_of_operation* (index 6060_h).

10.4 Functional Description

10.4.1 Modes of Operation Function

The device behaviour depends on the activated modes of operation.

It is possible to implement different device modes. Since it is not possible to operate the modes in parallel, the user is able to activate the required function by selecting a mode of operation. An example of exclusive functions are those for position and torque control, which can only control one variable at any one time. The variables can perform at most a limited function. Such hybrids are regarded as the particular characteristics of a mode of operation. Position control operation and encoder profile support can be active at the same time, for example. Consequently encoder profile support is not regarded as a mode of operation.

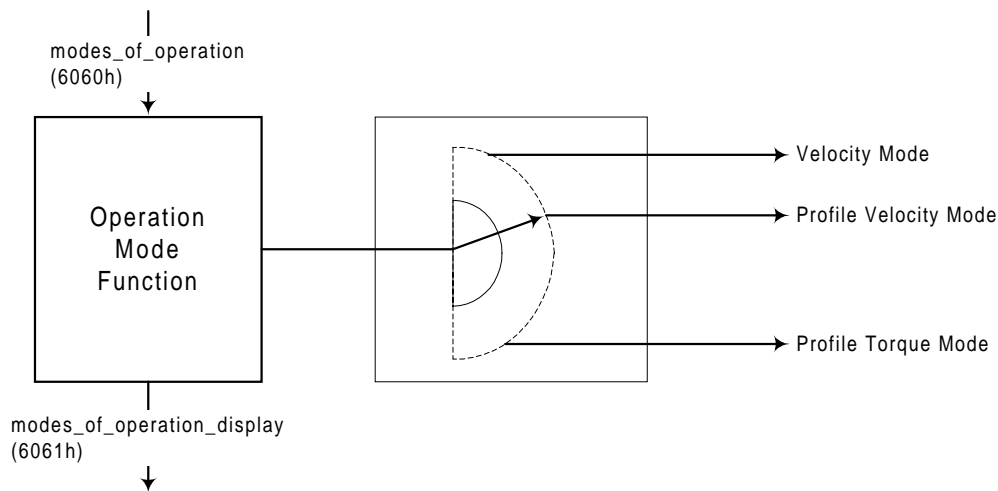


Figure 10: Operation Mode Function

It is possible for the user to switch between the various modes of operation as long as this is supported by the device. It is possible for the manufacturer to allow dynamic switching between different modes of operation at any time or to limit switching for example to the state 'SWITCH ON'. Switching can also be limited to the state 'local control'; i.e. not possible via the CAN-network. A device characteristic listed in the device function list can possible have several modes of operation.

The following modes of operation are listed:

- Velocity Mode (AC/DC drives, no feedback)
- Profile Velocity Mode (servo drives, feedback)
- Torque Profile Mode
- Homing Mode
- Profile Position Mode
- Interpolated Position Mode

With the exception of the 'Homing Mode', these listed modes of operation can all be put under the heading of 'setpoint setting'.

In parallel to this, manufacturer-specific modes of operation may also be available. These are not limited to setpoint settings.

The reference operation is regarded as a special form of a program function. The program function allows the user to run complex of time-critical sequence, e. g. tool change or special reference operations, directly in the device.

The switching between the modes of operation listed above should not incur any automatic reconfiguration of the process data channel. Problems which occur through switching of setpoint values during change of mode of operation must be monitored by the user. If necessary they can be rectified by prior reconfiguration of the process data channel.

Two objects are defined for management of the modes of operation.

- *modes_of_operation*
- *modes_of_operation_display*

The *statusword* contains bits, whose meaning is dependent on the mode of operation. When switching the mode of operation, the bits changing their meaning need to be monitored.

10.4.2 Drive Disabling Function

The drive disabling function defines the behaviour of the drive when transitioning from the 'OPERATION ENABLED' state to the 'READY TO SWITCH ON' (Shutdown command) or 'SWITCH ON' (Disable operation command) state.

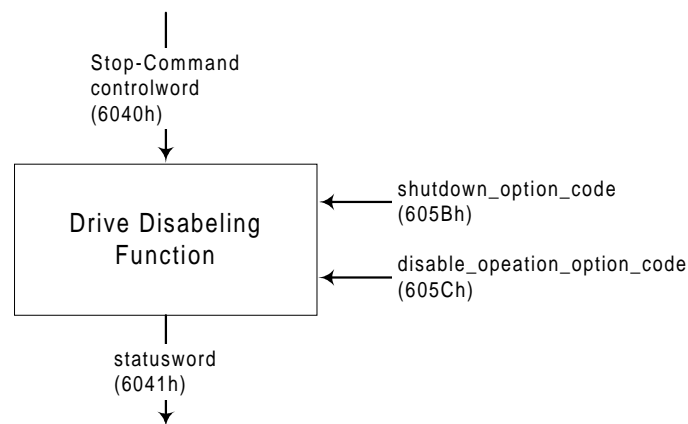


Figure 11: Modes of Operation Function

10.4.3 Quick Stop Function

The Quick Stop Function is triggered by the Quick Stop command.

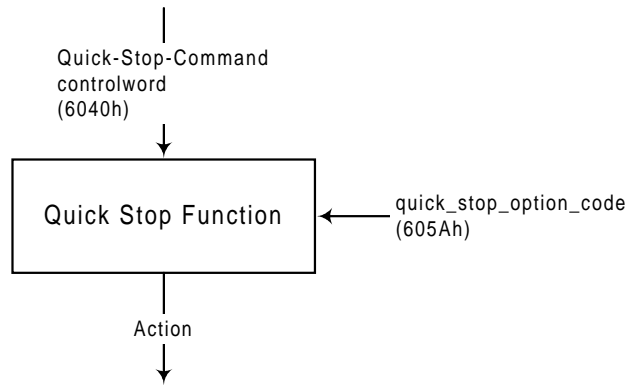


Figure 12: Quick-Stop-Function

10.4.4 Stop Function

The Stop Function may be triggered by resetting the bit 'RFG-disable' in the *controlword*.

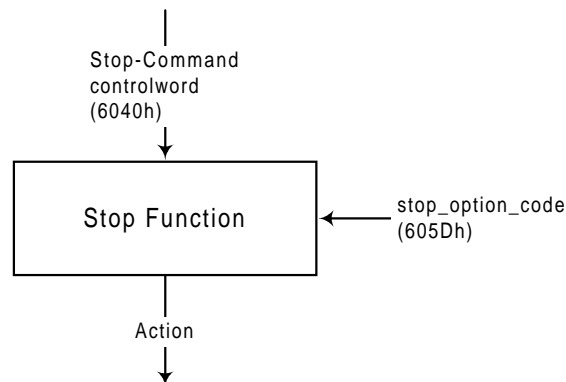


Figure 13: Stop-Function

10.4.5 Fault Reaction

Drive faults may be classified as fatal or non-fatal faults.

10.4.5.1 Fatal Faults

When a fatal fault occurs, the drive is no longer able to control the motor, so an immediate switch-off of the drive is necessary.

10.4.5.2 Non-Fatal Faults

When a non-fatal fault occurs, the drive can run the motor in a controlled fashion. The actions which are executed depend on the *fault_reaction_option_code*.

Once a fault occurs the drive will always enter the FAULT state, even if the fault clears before the drive enters the FAULT state. The FAULT state may only be left if the 'Fault Reset' command is received from a host, and no further fault is present in the drive.

11 Factor Group

11.1 General Information

11.1.1 Factors

There is a need to interchange physical dimensions and sizes into the device internal units. To implement the interchange, several factors are necessary. This chapter describes how these factors have an influence on the system, how they are calculated and which data is necessary to build them. Normalised parameters are denoted with an asterisk *.

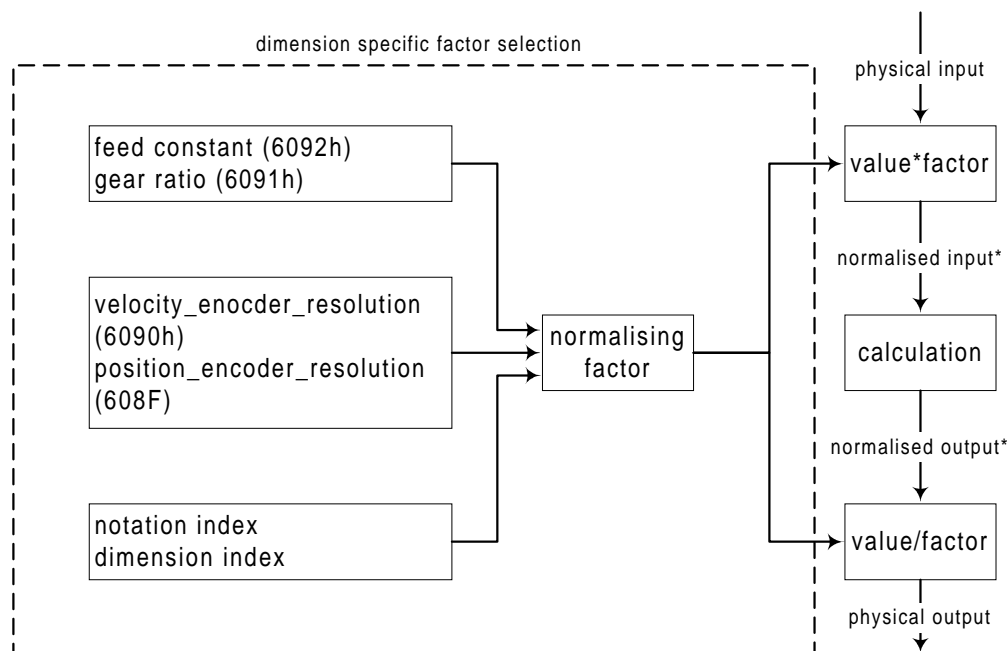


Figure 14: Influence of Factors

11.1.2 Relationship between Physical and Internal Units

The factors defined in the factor group set up a relationship between device internal units and physical units.

The factors are result of the calculation of two parameters called dimension index and notation index, which are defined in **Table D (see appendix Definition of Dimension Indices)**. One parameter indicates the physical dimension, the other the physical unit and a decimal exponent for the values. These factors are directly used to normalise the physical values.

The application specific parameters will be used in the corresponding mode of operation to build the described factors.

Parameters that are commonly used will be integrated in the object dictionary without defining their junctions. This guaranties a common parameter number for further use without the need for a predefinition.

parameter	typical values
<i>feed_constant</i>	0.1 1000 position units
<i>position_encoder_resolution</i>	50 10^6 inc
<i>velocity_encoder_resolution</i>	0 10^6 inc/s
<i>gear_ratio</i>	0.1 1000
<i>position_dimension_index</i>	mm, inch, grade
<i>position_notation_index</i>	-3
<i>velocity_dimension_index</i>	m/s, 1/min
<i>velocity_notation_index</i>	-3
<i>acceleration_dimension_index</i>	$1/s^2$
<i>acceleration_notation_index</i>	0
constants	value
torque resolution	0.001

The dimension index defines the physical dimension of the parameter. The notation index can be used in two ways:

- For a unit with decimal scaling and notation index < 64
the notation index defines the exponent/decimal place of the unit.
- or a unit with non-decimal scaling and notation index > 64
the notation index defines the sub-index of the physical dimension of the unit.

Examples for notation indices < 64:

For notation index <64 the value is used as an exponent. The unit is defined by the physical dimension and calculated by unit type and exponent, all declared in the dimension/notation index table **Table D (see appendix Definiton of Dimension Indices)**.

position unit

dimension index = 1: length

notation index = -6: micro meter

$$\begin{aligned}\text{position_units} &= 10^{\text{notation_index}} \times f(\text{dimension_index}) \\ &= 10^{-6} \text{ m}\end{aligned}$$

dimension index = 12: angle

notation index = 0: radian

$$\begin{aligned}\text{position_units} &= 10^{\text{notation_index}} \times f(\text{dimension_index}) \\ &= \text{radian}\end{aligned}$$

velocity unit

dimension index = 13: velocity

notation index = -3: milli metre per second

$$\begin{aligned}\text{velocity_units} &= 10^{\text{notation_index}} \times f(\text{dimension_index}) \\ &= 10^{-3} \text{ m/s}\end{aligned}$$

frequency units

dimension index = 28: frequency

notation index = 3: kilo hertz

$$\begin{aligned}\text{frequency_units} &= 10^{\text{notation_index}} \times f(\text{dimension_index}) \\ &= 10^3 \text{ Hz}\end{aligned}$$

Examples for notation indices > 64:

The unit is defined by the physical dimension and unit type, both declared in the dimension/notation index table **Table D (see appendix Definition of Dimension Indices)**

time units

dimension index = 4: time

notation index = 77: day

$$\begin{aligned}\text{time_units} &= f(\text{dimension_index}, \text{notation_index}) \\ &= \text{day}\end{aligned}$$

position unit

dimension index = 12: angle

notation index = 76: minute

$$\begin{aligned}\text{position_units} &= f(\text{dimension_index}, \text{notation_index}) \\ &= \text{minute}\end{aligned}$$

11.2 Object Dictionary Entries

11.2.1 Objects defined in this Chapter

Index	Object	Name	Type	Attr.	M/O
6089 _h	VAR	position_notation_index	Integer8	rw	O
608A _h	VAR	position_dimension_index	Unsigned8	rw	O
608B _h	VAR	velocity_notation_index	Integer8	rw	O
608C _h	VAR	velocity_dimension_index	Unsigned8	rw	O
608D _h	VAR	acceleration_notation_index	Integer8	rw	O
608E _h	VAR	acceleration_dimension_index	Unsigned8	rw	O
608F _h	ARRAY	position_encoder_resolution	Unsigned32	rw	O
6090 _h	ARRAY	velocity_encoder_resolution	Unsigned32	rw	O
6091 _h	ARRAY	gear_ratio	Unsigned32	rw	O
6092 _h	ARRAY	feed_constant	Unsigned32	rw	O
6093 _h	ARRAY	position_factor	Unsigned32	rw	O
6094 _h	ARRAY	velocity_encoder_factor	Unsigned32	rw	O
6095 _h	ARRAY	velocity_factor_1	Unsigned32	rw	O
6096 _h	ARRAY	velocity_factor_2	Unsigned32	rw	O
6097 _h	ARRAY	acceleration_factor	Unsigned32	rw	O
607E _h	VAR	polarity	Unsigned8	rw	O

11.3 Object Description

Objects in this group represent factors which are necessary to normalise the physical inputs and outputs. The user has to consider that the correct dimension and unit are used.

11.3.1 Object 6089_h: position_notation_index

The *position_notation_index* is used to scale the following objects:

position_actual_value
position_demand_value
target_position
position_window
following_error_window
home_offset
position_range_limit
software_position_limit
target_velocity

Index	6089 _h
Name	position_notation_index
Object Code	VAR
Data type	Integer8

Value Description

Object Class	M: -	O: pc pp ip hm pv tq
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	-128..127	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	

11.3.2 Object 608A_h: position_dimension_index

The *position_dimension_index* is necessary for:

position_actual_value
position_demand_value
target_position
position_window
following_error_window
home_offset
position_range_limit
software_position_limit
target_velocity

Index	608A _h
Name	position_dimension_index
Object Code	VAR
Data type	Unsigned8

Value Description

Object Class	M: -	O: pc pp ip hm pv tq
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..255	
Mandatory Range	0..32	
Default Value	-	
Substitute Value	-	

11.3.3 Object 608B_n: velocity_notation_index

The *velocity_notation_index* is necessary for:

velocity_actual_value
velocity_demand_value
end_velocity
profile_velocity
velocity_window
max_profile_velocity
velocity_threshold
homing_speeds

Index	608B _n
Name	velocity_notation_index
Object Code	VAR
Data Type	Integer8

Value Description

Object Class	M: -	O: pc pp ip hm pv tq
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	-128..127	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	

11.3.4 Object 608C_h: velocity_dimension_index

The *velocity_dimension_index* is necessary for:

velocity_actual_value
velocity_demand_value
end_velocity
profile_velocity
velocity_window
max_profile_velocity
velocity_threshold
homing_speeds

Index	608C _h
Name	velocity_dimension_index
Object Code	VAR
Data Type	Unsigned8

Value Description

Object Class	M: -	O: pc pp ip hm pv tq
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..255	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

11.3.5 Object 608D_h: acceleration_notation_index

The *acceleration_notation_index* is necessary for:

profile_acceleration
profile_deceleration
quick_stop_deceleration
homing_acceleration

Index	608D _h
Name	acceleration_notation_index
Object Code	VAR
Data type	Integer8

Value Description

Object Class	M: -	O: pc pp ip hm pv tq
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	-128..127	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

11.3.6 Object 608E_h: acceleration_dimension_index

The *acceleration_dimension_index* is necessary for:

profile_acceleration
profile_deceleration
quick_stop_deceleration
homing_acceleration

Index	608E _h
Name	acceleration_dimension_index
Object Code	VAR
Data type	Unsigned8

Value Description

Object Class	M: -	O: pc pp ip hm pv tq
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..255	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

11.3.7 Object 608F_h: position_encoder_resolution

The *position_encoder_resolution* defines the ratio of encoder increments per motor revolution.

$$\text{position_encoder_resolution} = \frac{\text{encoder_increments}}{\text{motor_revolutions}}$$

Index	608F_h
Name	position_encoder_resolution
Object Code	ARRAY
Number of Elements	2
Data Type	Unsigned32

Value Description

Sub-Index	01 _h	
Description	encoder_increments	
Object Class	M: -	O: pc pp ip pv tq
Access	rw	
PDO Mapping	Possible	
Units	inc	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	1	
Substitute Value	-	

Sub-Index	02 _h	
Description	motor_revolutions	
Object Class	M: -	O: pc pp ip pv tq
Access	rw	
PDO Mapping	Possible	
Units	inc	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	1	
Substitute Value	-	

11.3.8 Object 6090_h: velocity_encoder_resolution

The *velocity_encoder_resolution* defines the ratio of encoder increments/sec. per motor revolutions/sec.

$$\text{velocity_encoder_resolution} = \frac{\text{encoder_increments / sec}}{\text{motor_revolutions / sec.}}$$

Index	6090_h
Name	velocity_encoder_resolution
Object Code	ARRAY
Number of Elements	2
Data Type	Unsigned32

Value Description

Sub-Index	01 _h	
Description	encoder_increments_per_second	
Object Class	M: -	O: pc pp ip pv tq
Access	rw	
PDO Mapping	Possible	
Units	inc/sec	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	1	
Substitute Value	-	

Sub-Index	02 _h	
Description	motor_revolutions_per_second	
Object Class	M: -	O: pc pp ip pv tq
Access	rw	
PDO Mapping	Possible	
Units	inc/sec	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	1	
Substitute Value	-	

11.3.9 Object 6091_h: gear_ratio

The *gear_ratio* defines the ratio of feed in position units per driving shaft revolutions. This includes the gear if present.

$$\text{gear_ratio} = \frac{\text{motor_shaft_revolutions}}{\text{driving_shaft_revolutions}}$$

Index	6091_h
Name	gear_ratio
Object Code	ARRAY
Number of Elements	2
Data Type	Unsigned32

Value description

Sub-Index	01 _h	
Description	motor_revolutions	
Object Class	M: -	O: pc pp ip pv tq
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	1	
Substitute Value	-	

Sub-Index	02 _h	
Description	shaft_revolutions	
Object Class	M: -	O: pc pp ip pv tq
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	1	
Substitute Value	-	

11.3.10 Object 6092_h: feed_constant

The *feed_constant* defines the ratio of feed in position units per driving shaft revolutions. This includes the gear if present.

$$\text{feed_constant} = \frac{\text{feed [position_units]}}{\text{driving_shaft_revolutions}}$$

Index	6092 _h
Name	feed_constant
Object Code	ARRAY
Number of Elements	2
Data Type	Unsigned32

Value description

Sub-Index	01 _h	
Description	feed	
Object Class	M: -	O: pc pp ip pv tq
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	1	
Substitute Value	-	

Sub-Index	02 _h	
Description	shaft_revolutions	
Object Class	M: -	O: pc pp ip pv tq
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	1	
Substitute Value	-	

11.3.11 Object 6093_h: position_factor

The *position_factor* converts the desired position (in position units) into the internal format (in increments). This parameter may be calculated internally in the drive; nevertheless it is specified as read-writeable as the objects necessary for the calculation are defined as optional too and need not to be present in an implementation.

$$\text{position_factor} = \frac{\text{position_encoder_resolution} * \text{gear_ratio}}{\text{feed_constant}}$$

Index	6093 _h
Name	position_factor
Object Code	ARRAY
Number of Elements	2
Data Type	Unsigned32

Value description

Sub-Index	01 _h	
Description	numerator	
Object Class	M: -	O: pc pp ip
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	1	
Substitute Value	-	

Sub-Index	02 _h	
Description	feed_constant	
Object Class	M: -	O: pc pp ip
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	1	
Substitute Value	-	

11.3.12 Object 6094_h: velocity_encoder_factor

The *velocity_encoder_factor* converts the desired velocity (in velocity units) into the internal format (in increments).

$$\text{velocity_encoder_factor} = \frac{\text{velocity_encoder_resolution} * \text{gear_ratio} * \text{position_unit} * F_{\text{velocity}}(\text{notation_index})}{\text{feed_constant} * \text{velocity_unit} * \text{sec} * F_{\text{position}}(\text{notation_index})}$$

Index	6094_h
Name	velocity_encoder_factor
Object Code	ARRAY
Number of Elements	2
Data Type	Unsigned32

Value description

Sub-Index	01 _h	
Description	numerator	
Object Class	M: -	O: pv
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	1	
Substitute Value	-	

Sub-Index	02 _h	
Description	divisor	
Object Class	M: -	O: pv
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	1	
Substitute Value	-	

11.3.13 Object 6095_h: velocity_factor_1

The *velocity_factor_1* is used to convert motor data (e.g. maximum motor revolutions) into velocity data (e.g. maximum velocity), because both data items are based on different physical dimensions.

$$\text{velocity_factor_1} = \frac{\text{feed_constant} * \text{velocity_unit} * \text{sec} * F_{\text{position_unit}}(\text{notation_index})}{60 \text{ sec/min} * \text{gear_ratio} * \text{velocity_unit} * F_{\text{velocity_unit}}(\text{notation_index})}$$

Index	6095_h
Name	velocity_factor_1
Object Code	ARRAY
Number of Elements	2
Data Type	Unsigned32

Value description

Sub-Index	01 _h	
Description	numerator	
Object Class	M: -	O: pc pp ip pv tq
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	1	
Substitute Value	-	

Sub-Index	02 _h	
Description	divisor	
Object Class	M: -	O: pc pp ip pv tq
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	1	
Substitute Value	-	

11.3.14 Object 6096_h: velocity_factor_2

The *velocity_factor_2* is used to convert encoder data for positions into encoder data for velocity, because both data items are based on different physical dimensions. The velocity encoder system is transformed to the position encoder.

$$\text{velocity_factor_2} = \frac{\text{position_encoder_resolution}}{\text{velocity_encoder_resolution}}$$

Index	6096_h
Name	velocity_factor_2
Object Code	ARRAY
Number of Elements	2
Data Type	Unsigned32

Value description

Sub-Index	01 _h	
Description	numerator	
Object Class	M: -	O: pc pp ip pv tq
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	1	
Substitute Value	-	

Sub-Index	02 _h	
Description	divisor	
Object Class	M: -	O: pc pp ip pv tq
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	1	
Substitute Value	-	

11.3.15 Object 6097_h: acceleration_factor

The acceleration_factor converts the acceleration (in acceleration units/sec²) into the internal format (in increments/sec²).

$$\text{acceleration_factor} = \frac{\text{velocity_units} \cdot \text{velocity_encoder_factor}}{\text{acceleration_units} \cdot \text{sec}}$$

Index	6097_h
Name	acceleration_factor
Object Code	ARRAY
Number of Elements	2
Data Type	Unsigned32

Value description

Sub-Index	01 _h	
Description	numerator	
Object Class	M: -	O: pc pp ip pv tq
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	1	
Substitute Value	-	

Sub-Index	02 _h	
Description	divisor	
Object Class	M: -	O: pc pp ip pv tq
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	1	
Substitute Value	-	

11.3.16 Object 607E_h: polarity

Position_demand_value and *position_actual_value* are multiplied by 1 or -1 depending on the value of the polarity flag.

Index	607E _h
Name	polarity
Object Code	VAR
Data type	Unsigned8

Value Description

Object Class	M: -	O: pc pp pv tq ip
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..255	
Mandatory Range	0..192	
Default Value	0	
Substitute Value	-	

Bit	Meaning
7	Position polarity 0 => multiply by 1 (default) 1 => multiply by -1
6	Velocity polarity 0 => multiply by 1 (default) 1 => multiply by -1

12 Profile Position Mode

12.1 General Information

The overall structure for this mode is shown in Figure 15. A *target_position* is applied to the Trajectory Generator. It is generating a *position_demand_value* for the position control loop described in the Position Control Function (chapter 14). These two function blocks are optionally controlled by individual parameter sets.

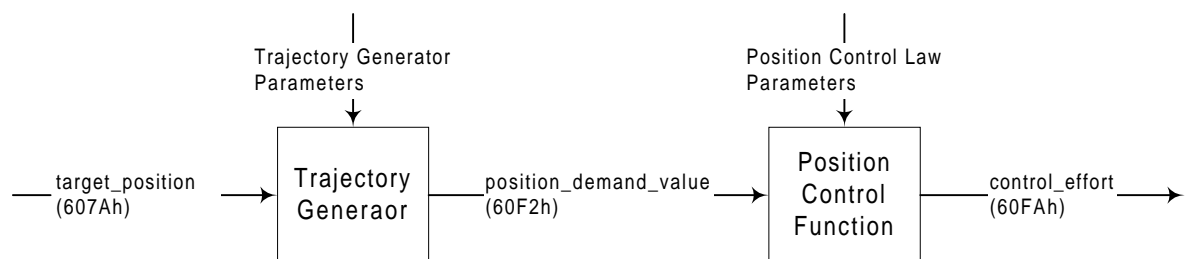
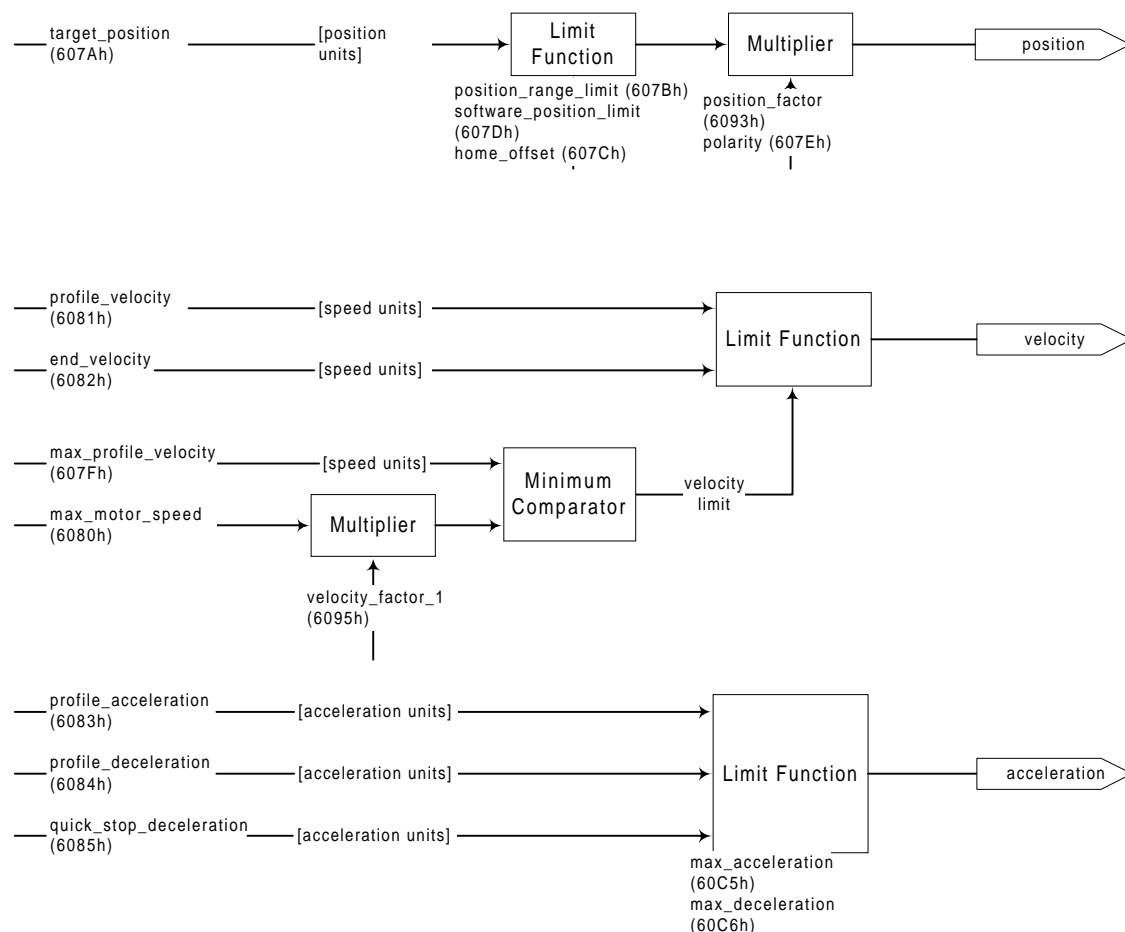


Figure 15: Overall Structure for the Profile Position Mode



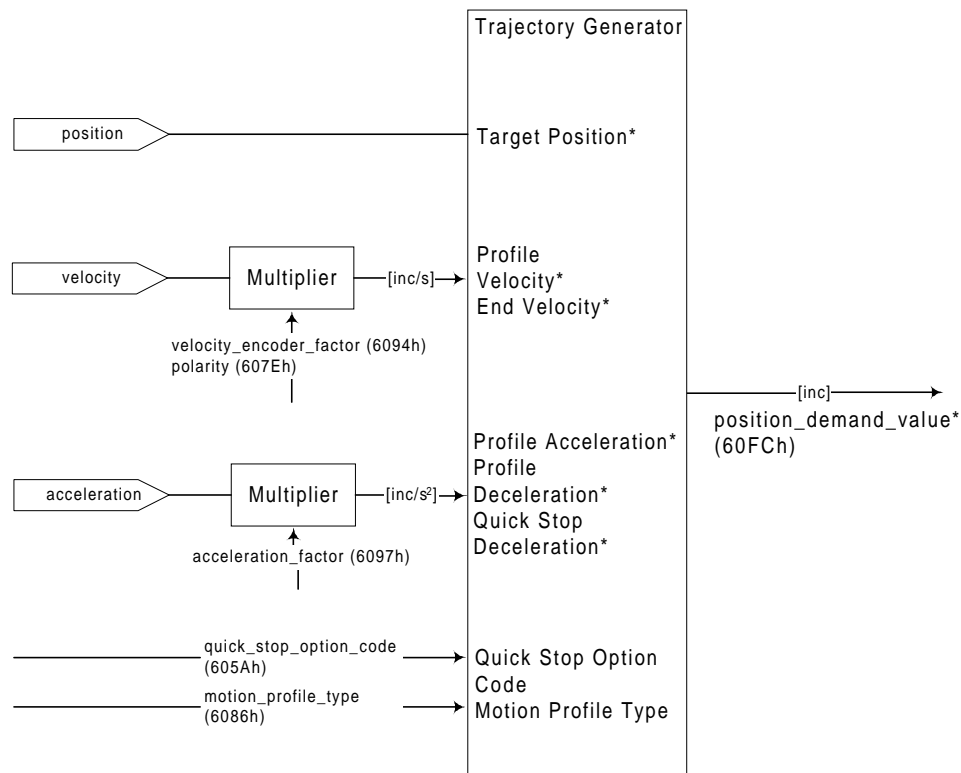


Figure 16: The Trajectory Generator

At the input to the Trajectory Generator, parameters may have optional limits applied before being normalised to internal units. Normalised parameters are denoted with an asterisk. The simplest form of a Trajectory Generator is just to pass through a *target_position* and to transform it to a *position_demand_value** with internal units (increments) only.

12.1.1 Input Data Description

Operating Mode	Input Parameters Used
Profile Position Mode	target_position, profile_velocity, end_velocity, profile_acceleration, profile_deceleration, quick_stop_deceleration, position_factor, quick_stop_option_code, polarity, velocity_encoder_factor, motion_profile_type, max_profile_velocity, max_motor_speed, position_range_limit, software_position_limit, acceleration_factor

12.1.2 Output Data Description

The output value provided by the Trajectory Generator is the input for Position Control Function. In that chapter the remotely accessible parameters of the device for a position control are described.

Operating Mode	Output Parameters Used
Profile Position Mode	position_demand_value*

12.2 Object Dictionary Entries

12.2.1 Objects defined in this Chapter

Index	Object	Name	Type	Attr.	M/O
607A _h	VAR	target_position	Integer32	rw	M
607B _h	ARRAY	position_range_limit	Integer32	rw	O
607D _h	ARRAY	software_position_limit	Integer32	rw	O
607F _h	VAR	max_profile_velocity	Unsigned32	rw	O
6080 _h	VAR	max_motor_speed	Unsigned16	rw	O
6081 _h	VAR	profile_velocity	Unsigned32	rw	M
6082 _h	VAR	end_velocity	Unsigned32	rw	O
6083 _h	VAR	profile_acceleration	Unsigned32	rw	M
6084 _h	VAR	profile_deceleration	Unsigned32	rw	M
6085 _h	VAR	quick_stop_deceleration	Unsigned32	rw	O
6086 _h	VAR	motion_profile_type	Integer16	rw	M
60C5 _h	VAR	max_acceleration	Unsigned32	rw	O
60C6 _h	VAR	max_deceleration	Unsigned32	rw	O

12.2.2 Objects defined in other Chapters

Index	Object	Name	Type	Chapter
6040 _h	VAR	controlword	Integer16	dc
6041 _h	VAR	statusword	Unsigned16	dc
605A _h	VAR	quick_stop_option_code	Integer16	dc
607E	VAR	polarity	Unsigned8	fg
6093 _h	ARRAY	position_factor	Unsigned32	fg
6094 _h	ARRAY	velocity_encoder_factor	Unsigned32	fg
6095 _h	ARRAY	velocity_factor_1	Unsigned32	fg
6097 _h	ARRAY	acceleration_factor	Unsigned32	fg

12.3 Object Description

12.3.1 Object 607A_n: target_position

The *target_position* is the position that the drive should move to in position profile mode using the current settings of motion control parameters such as velocity, acceleration, deceleration, *motion_profile_type* etc. The *target_position* is given in user defined position units. It is converted to position increments using the *position_factor* (see chapter Factor Group). The *target_position* will be interpreted as absolute or relative depending on the *absolute_relative* flag (bit 6) in the *controlword*.

Index	607A _n
Name	target_position
Object Code	VAR
Data type	Integer32

Value Description

Object Class	M: pp pc ip	O:
Access	rw	
PDO Mapping	Possible	
Units	position units	
Value Range	$(-2^{31})..(2^{31}-1)$	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

12.3.2 Object 607B_h: position_range_limit

Position_range_limit contains two sub-parameters, *min_position_range_limit* and *max_position_range_limit*. These limit the numerical range of the input value. On reaching or exceeding these limits, the input value automatically wraps to the other end of the range. Wrap-around of the input value can be prevented by setting software position limits.

Index	607B _h
Name	position_range_limit
Object Code	ARRAY
Number of Elements	2
Data Type	Integer32

Value description

Sub-Index	01 _h	
Description	min_position_range_limit	
Object Class	M: -	O: pp pc ip
Access	rw	
PDO Mapping	possible	
Units	position units	
Value Range	$(-2^{31})..(2^{31}-1)$	
Mandatory Range	-	
Default Value	-2^{31}	
Substitute Value	-	

Sub-Index	02 _h	
Description	max_position_range_limit	
Object Class	M: -	O: pp pc ip
Access	rw	
PDO Mapping	Possible	
Units	position units	
Value Range	$(-2^{31})..(2^{31}-1)$	
Mandatory Range	-	
Default Value	$2^{31}-1$	
Substitute Value	-	

12.3.3 Object 607D_h: software_position_limit

Software_position_limit contains the sub-parameters *min_position_limit* and *max_position_limit*. These parameters define the absolute position limits for the *position_demand_value* and the *position_actual_value*. Every new *target_position* must be checked against these limits. The limit positions are specified in position units (same as *target_position*) and are always relative to the machine home position. Before being compared with the *target_position* they must be corrected internally by the *home_offset* as follows:

$$\begin{aligned}\text{corrected_min_position_limit} &= \text{min_position_limit} - \text{home_offset} \\ \text{corrected_max_position_limit} &= \text{max_position_limit} - \text{home_offset}\end{aligned}$$

This calculation needs only be performed when *home_offset* or *software_position_limit* is changed.

Index	607D _h
Name	software_position_limit
Object Code	ARRAY
Number of Elements	2
Data Type	Integer32

Value description

Sub-Index	01 _h	
Description	min_position_limit	
Object Class	M: -	O: pp pc ip
Access	rw	
PDO Mapping	Possible	
Units	position units	
Value Range	(-2 ³¹)..(2 ³¹ -1)	
Mandatory Range	-	
Default Value	-2 ³¹	
Substitute Value	-	

Sub-Index	02 _h	
Description	max_position_limit	
Object Class	M: -	O: pp pc ip
Access	rw	
PDO Mapping	Possible	
Units	position units	
Value Range	$(-2^{31})..(2^{31}-1)$	
Mandatory Range	-	
Default Value	$2^{31}-1$	
Substitute Value	-	

12.3.4 Object 607F_h: max_profile_velocity

The *max_profile_velocity* is the maximum allowed speed in either direction during a profiled move. It is given in the same units as *profile_velocity*.

Index	607F_h
Name	max_profile_velocity
Object Code	VAR
Data type	Unsigned32

Value Description

Object Class	M: -	O: pp ip pv tq
Access	rw	
PDO Mapping	Possible	
Units	speed units	
Value Range	$0..(2^{31}-1)$	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

12.3.5 Object 6080_h: max_motor_speed

The *max_motor_speed* is the maximum allowable speed for the motor in either direction and is given in rpm. This is used to protect the motor and can be taken from the motor data sheet.

Index	6080_h
Name	max_motor_speed
Object Code	VAR
Data type	Unsigned16

Value description

Object Class	M: -	O: all
Access	rw	
PDO Mapping	Possible	
Units	rpm	
Value Range	0..65535	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

12.3.6 Object 6081_h: profile_velocity

The *profile_velocity* is the velocity normally attained at the end of the acceleration ramp during a profiled move and is valid for both directions of motion. The *profile_velocity* is given in user defined speed units. It is converted to position increments per second using the *velocity_encoder_factor* (see chapter Factor Group).

Index	6081_h
Name	profile_velocity
Object Code	VAR
Data type	Unsigned32

Value description

Object Class	M: pp pv	O:
Access	rw	
PDO Mapping	Possible	
Units	speed units	
Value Range	0..($2^{31}-1$)	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

12.3.7 Object 6082_h: end_velocity

The *end_velocity* defines the velocity which the drive must have on reaching the *target_position*. Normally, the drive stops at the *target_position*, i.e. the *end_velocity* = 0. The *end_velocity* is given in the same units as *profile_velocity*.

Index	6082_h
Name	end_velocity
Object Code	VAR
Data type	Unsigned32

Value description

Object Class	M: -	O: pp
Access	rw	
PDO Mapping	Possible	
Units	speed units	
Value Range	0..($2^{31}-1$)	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	

12.3.8 Object 6083_h: profile_acceleration

The *profile_acceleration* is given in user defined acceleration units. It is converted to position increments per second² using the normalizing factors (see chapter Factor Group).

Index	6083_h
Name	profile_acceleration
Object Code	VAR
Data type	Unsigned32

Value description

Object Class	M: pp pv	O:
Access	rw	
PDO Mapping	Possible	
Units	acceleration units	
Value Range	0..($2^{32}-1$)	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

12.3.9 Object 6084_h: profile_deceleration

The *profile_deceleration* is given in the same units as *profile_acceleration*. If this parameter is not supported, then the *profile_acceleration* value is also used for deceleration.

Index	6084_h
Name	profile_deceleration
Object Code	VAR
Data type	Unsigned32

Value description

Object Class	M: pp pv	O:
Access	rw	
PDO Mapping	Possible	
Units	acceleration units	
Value Range	0..($2^{32}-1$)	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

12.3.10 Object 6085_h: quick_stop_deceleration

The *quick_stop_deceleration* is the deceleration used to stop the motor if the quick stop command (bit 2 of the *controlword*) is given and the *quick_stop_option_code* (605Ah) is set to 2. The *quick_stop_deceleration* is given in the same units as the *profile_acceleration*.

Index	6085_h
Name	quick_stop_deceleration
Object Code	VAR
Data type	Unsigned32

Value description

Object Class	M: -	O: pp ip pv vo tq hc
Access	rw	
PDO Mapping	Possible	
Units	acceleration units	
Value Range	0..($2^{32}-1$)	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

12.3.11 Object 6086_h: motion_profile_type

The *motion_profile_type* is used to select the type of motion profile used to perform a profiled move.

Index	6086_h
Name	motion_profile_type
Object Code	VAR
Data type	Integer16

Value description

Object Class	M: pp pv	O:
Access	rw	
PDO Mapping	Possible	
Units	none	
Value Range	-32768..32767	
Mandatory Range	-	
Default Value	0	
Substitute Value	0	

Profile Code	Profile Type
-32768 ... -1	Manufacturer specific
0	Linear ramp (trapezoidal profile)
1	Sin ² ramp
2	Jerk-free ramp
3	Jerk-limited ramp
4 .. 32767	Reserved for future profile types

12.3.12 Object 60C5_h: max_acceleration

To prevent the motor and the application from being destroyed, the *max_acceleration* can be used to limit the acceleration to an acceptable value.

The max_acceleration is given in user defined *acceleration_units* (608D_h , 608E_h). It is converted to position increments per second² using the *acceleration_factor* (6097_h).

Index	60C5 _h
Name	max_acceleration
Object Code	VAR
Data type	Unsigned32

Value description

Object Class	M: -	O: ip pp pc
Access	rw	
PDO Mapping	Possible	
Units	acceleration units	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	-	
Substitute Value	0 (no acceleration limitation)	

12.3.13 Object 60C6_h: max_deceleration

To prevent the motor and the application from being destroyed, the *max_deceleration* can be used to limit the deceleration to an acceptable value.

The *max_deceleration* is given in the same units as the *max_acceleration* (60C5_h). If this parameter is not supported, then the *max_acceleration* value is also used for deceleration.

Index	60C6 _h
Name	max_deceleration
Object Code	VAR
Data type	Unsigned32

Value description

Object Class	M: -	O: ip pp pc
Access	rw	
PDO Mapping	Possible	
Units	acceleration units	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	-	
Substitute Value	max_acceleration	

12.4 Functional Description

Two different ways to apply *target_positions* to a drive, are supported by this device profile.

Set of setpoints:

After reaching the *target_position* the drive unit immediatly processes the next *target_position* which results in a move where the velocity of the drive normally is not reduced to zero after achieving a setpoint.

Single setpoints:

After reaching the *target_position* the drive unit signals this status to a host computer and then receives a new setpoint. After reaching a *target_position* the velocity normally is reduced to zero before starting a move to the next setpoint.

The two modes are controlled by the timing of the bits *new_setpoint* and *change_set_immediatly* in the *controlword* and *setpoint_acknowledge* in the *statusword*. These bits allow to set up a request-response mechanism in order to prepare a set of setpoints while another set still is processed in the drive unit. This minimizes reaction times within a control program on a host computer.

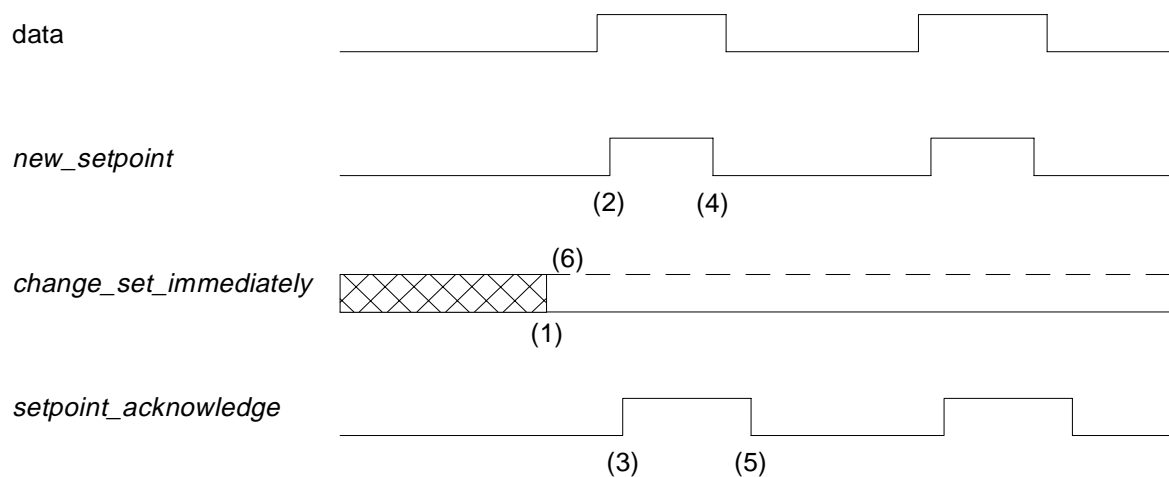


Figure 17: Setpoint transmission from a host computer

Figure 17, Figure 18 and Figure 19 show the difference between the "set of setpoints" mode and the "single setpoint" mode. The initial status of the bit *change_set_immediately* in the *controlword* determines which mode is used. To keep simple these examples, only trapezoidal moves are used.

If the bit *change_set_immediately* is "0" (continuously drawn line in Figure 17) a single setpoint is expected by the drive (1). After data is applied to the drive, a host signals that the data is valid by changing the bit *new_setpoint* to "1" in the *controlword* (2). The drive responds with *setpoint_acknowledge* set to "1" in the *statusword* (3) after it recognized and buffered the new valid data. Now the host may release *new_setpoint* (4) and afterwards the drive signals with *setpoint_acknowledge* equal "0" its ability to accept new data again (5). In Figure 18 this mechanism results in a velocity of zero after ramping down in order to reach a *target_position* x_1 at t_1 . After signalling to the host, that the setpoint is reached like described above, the next *target_position* x_2 is processed at t_2 and reached at t_3 .

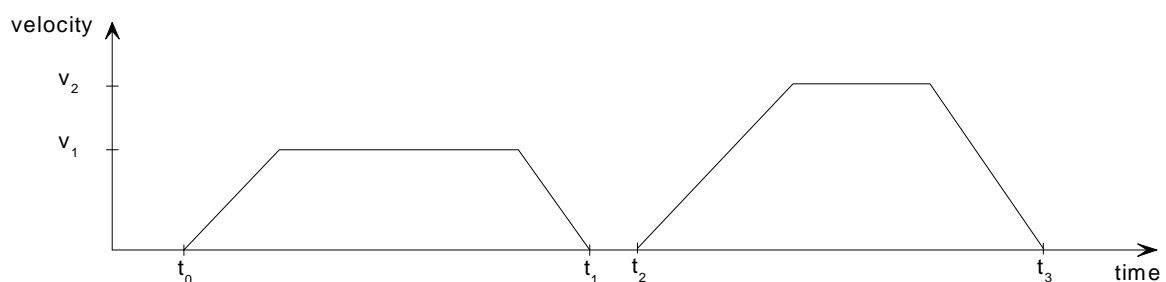


Figure 18: Single setpoint

With *change_set_immediately* set to "1" (6), symbolized by the dashed line in Figure 17, the host advises the drive to apply a new setpoint immediately after reaching the last one. The relative timing of the other signals is unchanged. This behaviour causes the drive to already process the next setpoint x_2 and to keep its velocity when it reaches the *target_position* x_1 at t_1 . Then drive moves immediately to the already calculated next *target_position* x_2 .

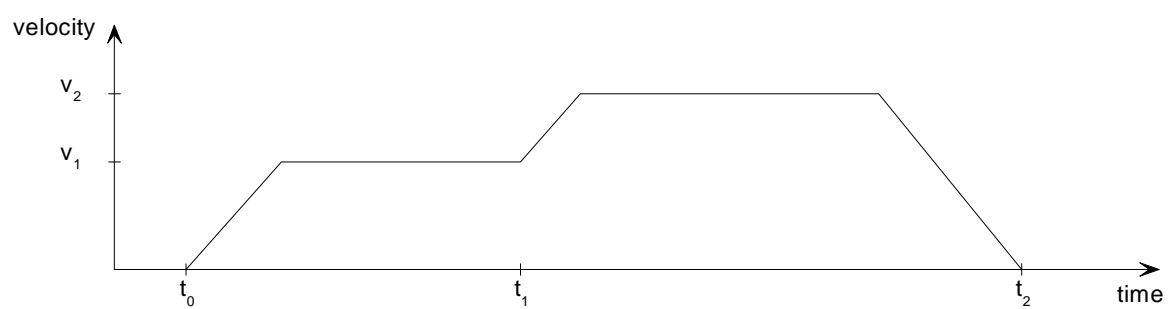


Figure 19: Change set immediatly

13 Homing Mode

13.1 General Information

This chapter describes the method by which a drive seeks the home position (also called, the datum, reference point or zero point). There are various methods of achieving this using limit switches at the ends of travel or a home switch (zero point switch) in mid-travel, most of the methods also use the index (zero) pulse train from an incremental encoder.

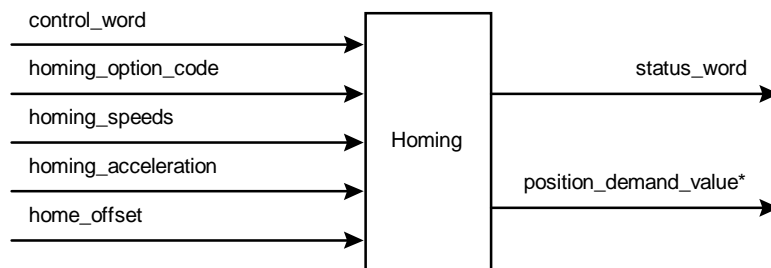


Figure 20: The Homing Function

13.1.1 Input Data Description

The user can specify the speeds, acceleration and the method of homing. There is a further object *home_offset* which allows the user to displace zero in the user's coordinate system from the home position.

There are two *homing_speeds*; in a typical cycle the faster speed is used to find the home switch and the slower speed is used to find the index pulse. The manufacturer is allowed some discretion in the use of these speeds as the response to the signals may be Dependent upon the hardware used.

13.1.2 Output Data Description

There is no output data except for those bits in the *statusword* which return the status or result of the homing process and the demand to the position control loops.

13.1.3 Internal States

There is only one internal state called homing which is reflected in the bits of the *statusword*.

13.2 Object Dictionary Entries

13.2.1 Objects defined in this Chapter

Index	Object	Name	Type	Attr.	M/O
607C _h	VAR	home_offset	Integer32	rw	O
6098 _h	VAR	homing_method	Integer8	rw	M
6099 _h	ARRAY	homing_speeds	Unsigned32	rw	M
609A _h	VAR	homing_acceleration	Unsigned32	rw	O

13.2.2 Objects defined in other Chapters

Index	Object	Name	Type	Chapter
6040 _h	VAR	controlword	Integer16	dc
6041 _h	VAR	statusword	Unsigned16	dc

13.3 Object Description

13.3.1 Object 607C_h: home_offset

The *home_offset* object is the difference between the zero position for the application and the machine home position (found during homing). During homing the home position is found and once the homing is completed the zero position is offset from the home position by adding the *home_offset* to the home position. All subsequent absolute moves shall be taken relative to this new zero position. This is illustrated in the following diagram.

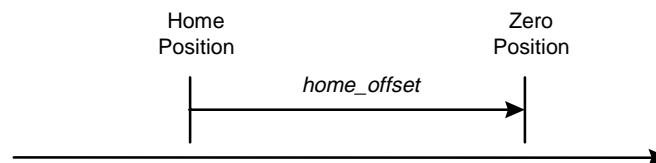


Figure 21: Home Offset

If the *home_offset* is not implemented then it shall be zero.

Index	607C_h
Name	home_offset
Object Code	VAR
Data type	Integer32

Value Description

Object Class	M: -	O: hm
Access	rw	
PDO Mapping	Possible	
Units	position units	
Value Range	$(-2^{31})..(2^{31}-1)$	
Mandatory Range	-	
Default Value	0	
Substitute Value	0	

13.3.2 Object 6098_h: homing_method

The homing_method object determines the method that will be used during homing.

Index	6098_h
Name	homing_method
Object Code	VAR
Data Type	Integer8

Value Description

Object Class	M: hm	O:
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	-128..127	
Mandatory Range	0	
Default Value	0	
Substitute Value	0	

Data Description

<i>homing_method</i>	Meaning
-128 .. -1	Manufacturer specific
0	No homing operation required
1..35	Methods 1 to 35 (see the functional description)
36 .. 127	Reserved

13.3.3 Object 6099_h: homing_speeds

This entry in the object dictionary defines the speeds used during homing.

Index	6099_h
Name	homing_speeds
Object Code	ARRAY
Number of Elements	2
Data Type	Unsigned32

Value Description

Sub-Index	01 _h	
Description	speed_during_search_for_switch	
Object Class	M: hm	O: -
Access	rw	
PDO Mapping	Possible	
Units	velocity Units	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	0	
Substitute Value	0	

Sub-Index	02 _h	
Description	speed_during_search_for_zero	
Object Class	M: hm	O: -
Access	rw	
PDO Mapping	Possible	
Units	velocity units	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	0	
Substitute Value	0	

13.3.4 Object 609A_h: homing_acceleration

The homing_acceleration establishes the acceleration to be used for all accelerations and decelerations with the standard homing modes.

Index	609A_h
Name	homing_acceleration
Object Code	VAR
Data Type	Unsigned32

Value Description

Object Class	M: -	O: hm
Access	rw	
PDO Mapping	Possible	
Units	acceleration units	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

13.4 Functional Description

By choosing a method of homing by writing a value to homing_method will clearly establish

- the homing signal (positive limit switch, negative limit switch, home switch)
- the direction of actuation and where appropriate
- the position of the index pulse.

The home position and the zero position are offset by the home_offset, see the definition of home_offset for how this offset is used.

Various homing positions are illustrated in the following diagrams. An encircled number indicates the code for selection of this homing position. The direction of movement is also indicated. Further homing methods may be defined by the manufacturer using the negative values of homing_method.

There are four sources of homing signal available, these are the negative and positive limit switches, the home switch and the index pulse from an encoder:

In the diagrams of homing sequences shown below, the encoder count increases as the axle's position moves to the right, in other words the left is the minimum position and the right is the maximum position.

For the operation of positioning drives, an exact knowledge of the absolute position is normally required. Since for cost reasons, drives often do not have an absolute encoder, a

homing operation is necessary. There are several, application-specific methods. The `homing_method` is used for selection.

The exact sequence of the homing operation is clearly described by the method. In some circumstances, a device has several methods to choose from, using the `homing_method`.

13.4.1 Homing Methods

The following sub-sections describe the details of how each of the homing modes shall function.

13.4.1.1 Method 1: Homing on the Negative Limit Switch

Using this method the initial direction of movement is leftward if the negative limit switch is inactive (here shown as low). The home position is at the first index pulse to the right of the position where the negative limit switch becomes inactive.

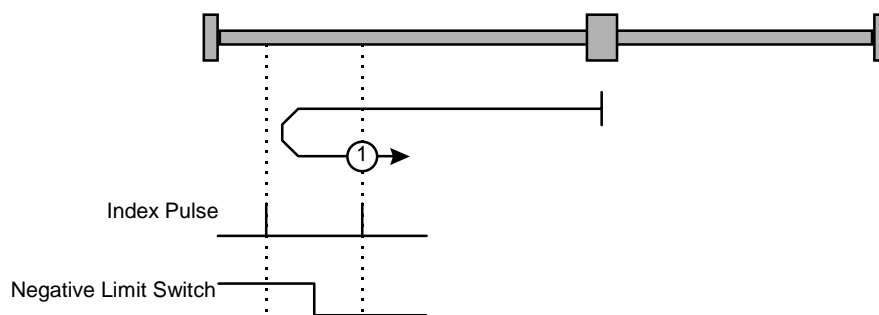


Figure 22: Homing on the Negative Limit Switch

13.4.1.2 Method 2: Homing on the Positive Limit Switch

Using this method the initial direction of movement is rightward if the positive limit switch is inactive (here shown as low). The position of home is at the first index pulse to the left of the position where the positive limit switch becomes inactive.

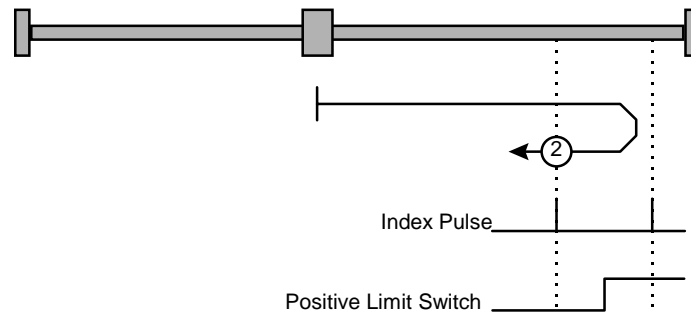


Figure 23: Homing on the Positive Limit Switch

13.4.1.3 Methods 3 and 4: Homing on the Positive Home Switch and Index Pulse

Using methods 3 or 4 the initial direction of movement is dependent on the state of the home switch. The home position is at the index pulse to either to the left or the right of the point where the home switch changes state. If the initial position is sited so that the direction of movement must reverse during homing, the point at which the reversal takes place is anywhere after a change of state of the home switch.

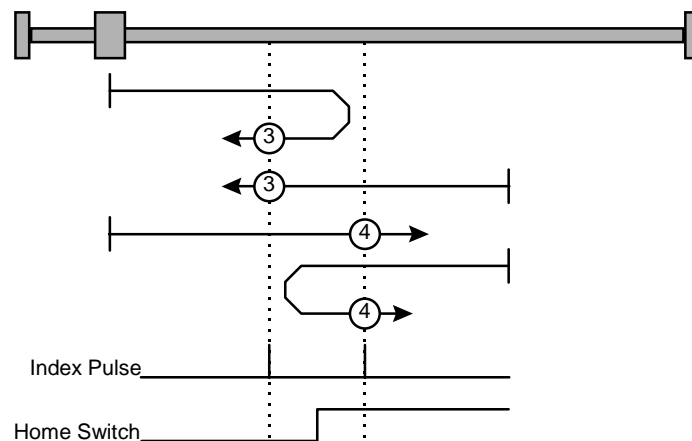


Figure 24: Homing on the Positive Home Switch and Index Pulse

13.4.1.4 Methods 5 and 6: Homing on the Negative Home Switch and Index Pulse

Using methods 5 or 6 the initial direction of movement is dependent on the state of the home switch. The home position is at the index pulse to either to the left or the right of the point where the home switch changes state. If the initial position is sited so that the direction of movement must reverse during homing, the point at which the reversal takes place is anywhere after a change of state of the home switch.

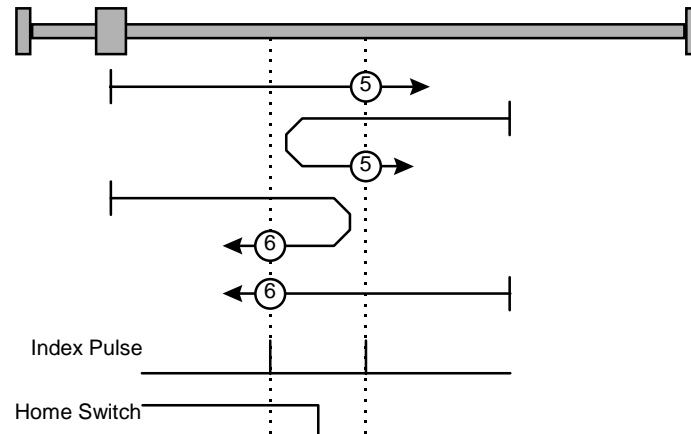


Figure 25: Homing on the Negative Home Switch and Index Pulse

13.4.1.5 Methods 7 to 14: Homing on the Home Switch and Index Pulse

These methods use a home switch which is active over only portion of the travel, in effect the switch has a 'momentary' action as the axle's position sweeps past the switch.

Using methods 7 to 10 the initial direction of movement is to the right, and using methods 11 to 14 the initial direction of movement is to the left except if the home switch is active at the start of the motion. In this case the initial direction of motion is Dependent on the edge being sought. The home position is at the index pulse on either side of the rising or falling edges of the home switch, as shown in the following two diagrams. If the initial direction of movement leads away from the home switch, the drive must reverse on encountering the relevant limit switch.

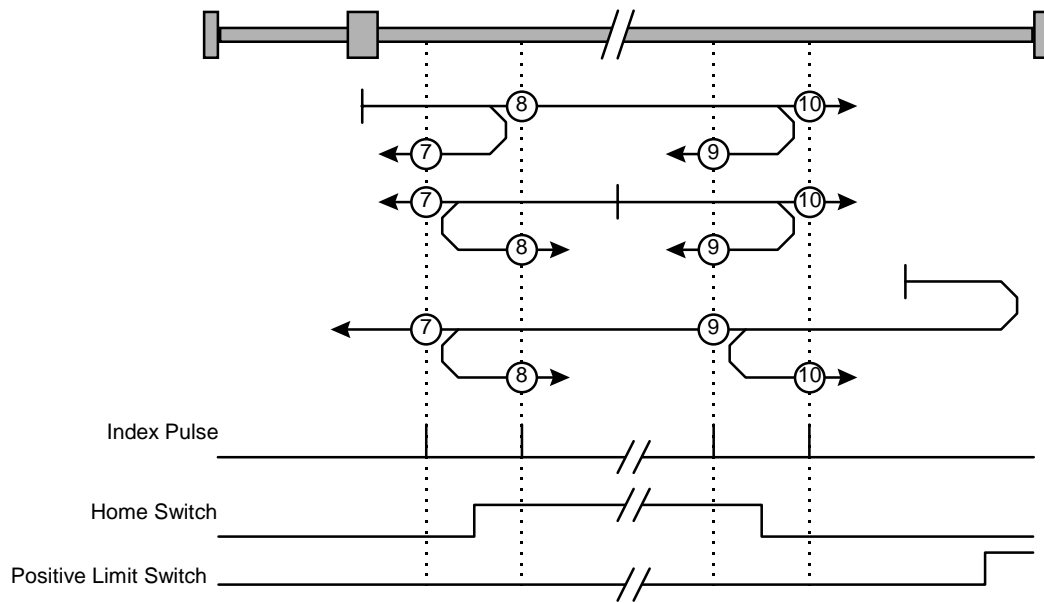


Figure 26: Homing on the Home Switch and Index Pulse - Positive Initial Move

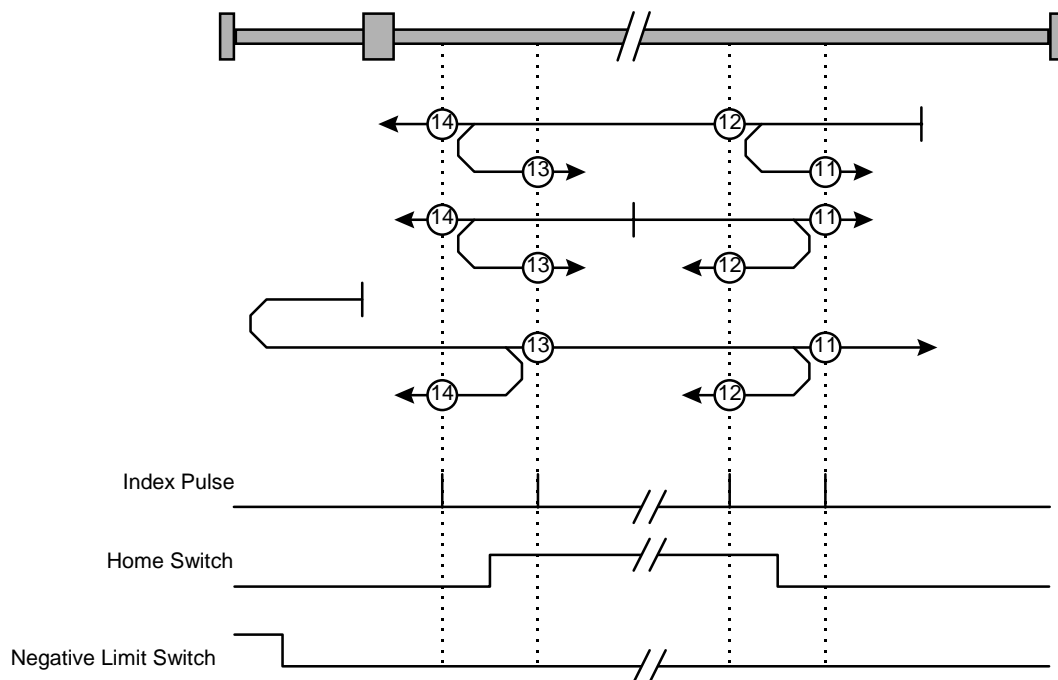


Figure 27: Homing on the Home Switch and Index Pulse - Negative Initial Move

13.4.1.6 Methods 15 and 16: Reserved

These methods are reserved for future expansion of the homing mode.

13.4.1.7 Methods 17 to 30: Homing without an Index Pulse

These methods are similar to methods 1 to 14 except that the home position is not dependent on the index pulse but only dependent on the relevant home or limit switch transitions. For example methods 19 and 20 are similar to methods 3 and 4 as shown in the following diagram.

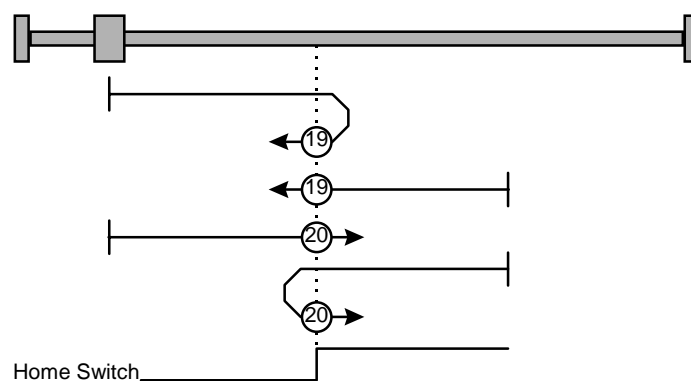


Figure 28: Homing on the Positive Home Switch

13.4.1.8 Methods 31 and 32: Reserved

These methods are reserved for future expansion of the homing mode.

13.4.1.9 Methods 33 to 34: Homing on the Index Pulse

Using methods 33 or 34 the direction of homing is negative or positive respectively. The home position is at the index pulse found in the selected direction.

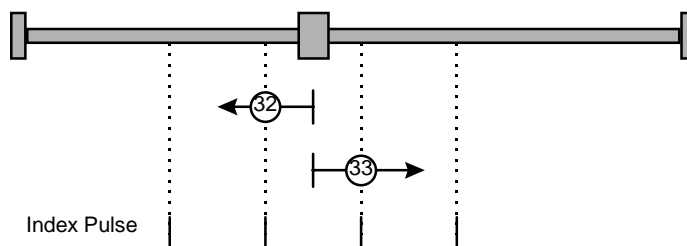


Figure 29: Homing on the Index Pulse

13.4.1.10 Method 35: Homing on the Current Position

In method 35 the current position is taken to be the home position.

13.4.2 Homing Mode Sequence

The homing operation is started by setting bit 4 in the device controlword. The successful completion is indicated by a one in bit 12 of the device statusword. A one in bit 13 of the statusword indicates an error was detected during the homing operation. The cause is found by reading the error codes.

Bit 4	Meaning
0	homing operation inactive
0→1	Start homing operation
1	homing operation active
1→0	Interrupt homing operation

Table 11: Extended Description of the Bits in the controlword

Bit 13	Bit 12	Meaning
0	0	homing operation not yet completed
0	1	homing operation carried out successfully
1	0	homing operation not successfully carried out
1	1	prohibited condition

Table 12: Extended Description of the Bits in the statusword

14 Position Control Function

14.1 General information

In this chapter, all parameters are described which are necessary for a closed loop position control. The control loop is fed with the *position_demand_value* as one of the outputs of the Trajectory Generator and with the output of the position detection unit (*position_actual_value*) like a resolver or encoder as input parameters. The behaviour of the control may be influenced by control parameters which are externally applicable. To keep stable the loop, a relative limitation of the output using the previous *control_effort* is possible. In order not to exceed physical limits of a drive, an absolute limit function is implemented for the *control_effort*. The *control_effort* may be a *velocity_demand_value*, a *position_demand_value* or any other output value, depending on the modes of operation implemented by a manufacturer. Especially in cascaded control structures, where a position control is followed by a torque control e.g., the *control_effort* of the position control loop is used as an input for a further calculation.

All values are transformed - if necessary - from user defined units to normalised units like increments with the functions of the chapter Factor Group.

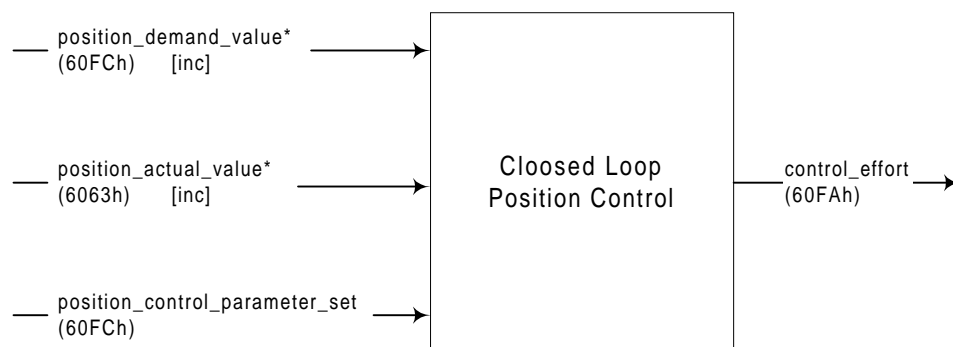


Figure 31: Position Control Function

Within this chapter, the following sub-functions are defined:

1. Following Error

A *position_actual_value* outside the allowed range of the *following_error_window* around a *position_demand_value* for longer than the *following_error_time_out* results in setting bit 13 *following_error* in the *statusword*.

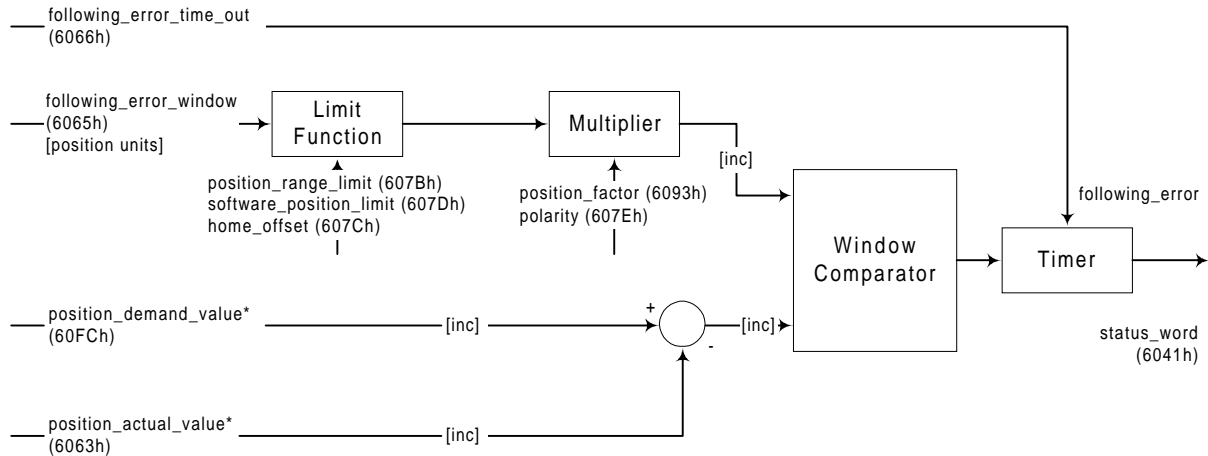


Figure 31: Following Error - Functional Overview

2. Position Reached

This function offers the possibility to define a position range around a *position_demand_value* to be regarded as valid. If a drive's position is within this area for a specified time - the *position_window_time* - the related control bit 10 *target_reached* in the *statusword* is set.

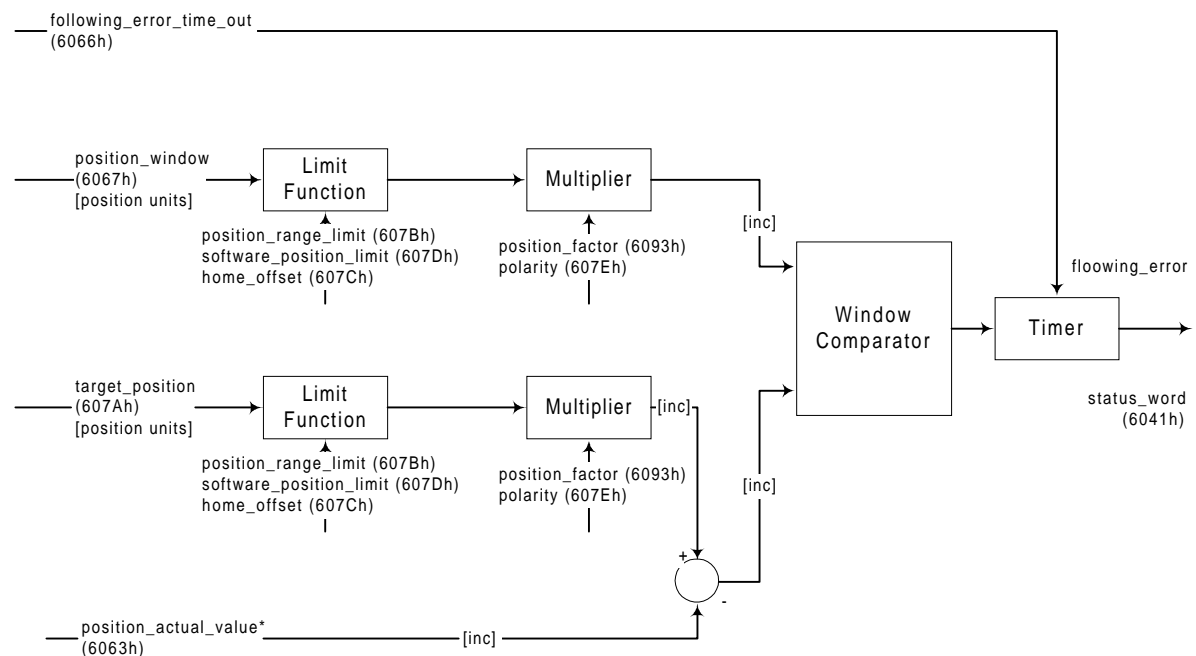


Figure 32: Position Reached - Functional Overview

The control functions following error and position reached have direct access to the *statusword* and give immediate notification to the user if their results change.

14.1.1 Input Data Description

Depending on the supported modes of operation and on the capabilities of different categories of drives, only some of the mentioned input parameters may be necessary.

Operating Mode	Input Parameters used
position mode, homing mode, interpolated position mode	position_demand_value*, position_window_time, position_window, following_error_time_out, following_error_window, position_actual_value, digital_inputs, target_position, position_factor, position_range_limit, polarity

14.1.2 Output Data Description

Operating Mode	Output Parameters used
position mode, homing mode, interpolated position mode	statusword, control_effort, digital_outputs

14.2 Object Dictionary Entries

14.2.1 Objects defined in this Chapter

Index	Object	Name	Type	Attr.	M/O
6062 _h	VAR	position_demand_value	Integer32	ro	O
6063 _h	VAR	position_actual_value*	Integer32	ro	O
6064 _h	VAR	position_actual_value	Integer32	ro	M
6065 _h	VAR	following_error_window	Unsigned32	rw	O
6066 _h	VAR	following_error_time_out	Unsigned16	rw	O
6067 _h	VAR	position_window	Unsigned32	rw	O
6068 _h	VAR	position_window_time	Unsigned16	rw	O
60F4 _h	VAR	following_error_actual_value	Integer32	ro	O
60FA _h	VAR	control_effort	Integer32	ro	O
60FB _h	RECORD	position_control_parameter_set	-	rw	O
60FC _h	VAR	position_demand_value*	Integer32	ro	O

14.2.2 Objects defined in other Chapters

Index	Object	Name	Type	Chapter
607A _h	VAR	target_position	Integer32	pp
607B _h	VAR	position_range_limit	Integer32	pp
607C _h	VAR	home_offset	Integer32	hm
607D _h	VAR	software_position_limit	Integer32	pp
607E _h	VAR	polarity	Unsigned8	fg
6093 _h	VAR	position_factor	Unsigned32	fg
6094 _h	ARRAY	velocity_encoder_factor	Unsigned32	fg
6095 _h	ARRAY	velocity_factor_1	Unsigned32	fg
6096 _h	ARRAY	acceleration_factor	Unsigned32	fg
6041 _h	VAR	controlword	Integer16	dc
6041 _h	VAR	statusword	Unsigned16	dc

14.3 Object Description

14.3.1 Object 6062_h: position_demand_value

Index	6062_h
Name	position_demand_value
Object Code	VAR
Data type	Integer32

Value Description

Object Class	M: -	O: pc
Access	ro	
PDO Mapping	Possible	
Units	position units	
Value Range	$(-2^{31})..(2^{31}-1)$	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

14.3.2 Object 6063_h: position_actual_value*

The actual value of the position measurement device is one of the two input values of the closed loop position control. The data unit is defined as increments. If necessary, the data unit must be transformed with the *position_factor* defined in the Factor Group from user defined units to increments.

Index	6063_h
Name	position_actual_value*
Object Code	VAR
Data type	Integer32

Value Description

Object Class	M: -	O: pc, pp, ip, hm, tq
Access	ro	
PDO Mapping	Possible	
Units	increments	
Value Range	$(-2^{31})..(2^{31}-1)$	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

14.3.3 Object 6064_h: position_actual_value

This object represents the actual value of the position measurement device in user defined units.

Index	6064_h
Name	position_actual_value
Object Code	VAR
Data type	Integer32

Value Description

Object Class	M: pc	O: pc, pp, ip, hm, tq
Access	ro	
PDO Mapping	Possible	
Units	position units	
Value Range	$(-2^{31})..(2^{31}-1)$	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

14.3.4 Object 6065_h: following_error_window

The *following_error_window* defines a range of tolerated position values symmetrically to the *position_demand_value*. As it is in most cases used with user defined units, a transformation into increments with the *position_factor* is necessary. If the *position_actual_value* is out of the *following_error_window*, a following error occurs

A following error might occur when:

- a drive is blocked
- unreachable profile velocity
- wrong closed loop coefficients

If the value of the following error window is $2^{32}-1$, the following control is switched off.

Index	6065_h
Name	following_error_window
Object Code	VAR
Data type	Unsigned32

Value Description

Object Class	M: -	O: pc, pp
Access	rw	
PDO Mapping	Possible	
Units	position units	
Value Range	0..($2^{32}-1$)	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

14.3.5 Object 6066_h: following_error_time_out

When a following error occurs longer than the defined value of the time-out, the corresponding bit 13 *following_error* in the status message will be set to one. The reaction of the drive when a following error occurs, is manufacturer specific.

Index	6066_h
Name	following_error_time_out
Object Code	VAR
Data type	Unsigned16

Value Description

Object Class	M: -	O: pc, pp
Access	rw	
PDO Mapping	Possible	
Units	milliseconds	
Value Range	0..65535	
Mandatory Range	-	
Default Value	-	
Substitute Value	0	

14.3.6 Object 6067_h: position_window

The *position_window* defines a symmetrical range of accepted positions relatively to the *target_position*. If the actual value of the position encoder is within the *position_window*, this *target_position* is regarded as reached. As the user mostly prefers to specify the *position_window* in his application in user defined units, the *position_factor* of the Factor Group must be used to transform this value into increments. The *target_position* has to be handled in the same manner as in the Trajectory Generator concerning limiting functions and transformation into internal machine units before it can be used with this function.

If the value of the position window is $2^{32}-1$, the position window control is switched of.

Index	6067_h
Name	position_window
Object Code	VAR
Data type	Unsigned32

Value Description

Object Class	M: -	O: pc
Access	rw	
PDO Mapping	Possible	
Units	position units	
Value Range	0..($2^{32}-1$)	
Mandatory Range	-	
Default Value	-	
Substitute Value	0	

14.3.7 Object 6068_h: position_window_time

When the actual position is within the *position_window* during the defined *position_window_time*, the corresponding bit 10 *target_reached* in the *statusword* will be set to one.

Index	6068_h
Name	position_window_time
Object Code	VAR
Data type	Unsigned16

Value Description

Object Class	M: -	O: pc
Access	rw	
PDO Mapping	Possible	
Units	milliseconds	
Value Range	0..65535	
Mandatory Range	-	
Default Value	-	
Substitute Value	0	

14.3.8 Object 60F4_h: following_error_actual_value

This object represents the actual value of the following error.

Index	60F4_h
Name	following_error_actual_value
Object Code	VAR
Data type	Integer32

Value description

Object Class	M: -	O: pc, pp, ip, hm, tq
Access	ro	
PDO Mapping	Possible	
Units	position units	
Value Range	$(-2^{31})..(2^{31}-1)$	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

14.3.9 Object 60FA_h: control_effort

The output of the position control loop is the *control_effort*. It is particular to the Position Control Function that the notation of the *control_effort* is mode dependent and therefore not specified in the object description.

Index	60FA_h
Name	control_effort
Object Code	VAR
Data type	Integer32

Value Description

Object Class	M: -	O: pc, pp, ip, hm, tq
Access	ro	
PDO Mapping	Possible	
Units	-	
Value Range	$(-2^{31})..(2^{31}-1)$	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

14.3.10 Object 60FB_h: position_control_parameter_set

In order to control the behaviour of the position control loop, one or more parameters are necessary. This object is a means to define control parameters which are highly manufacturer specific. For this reason, these parameters shall not be described in this document at all.

Index	60FB_h
Name	position_control_parameter_set
Object Code	RECORD
Number of Elements	0..255

Value Description

Sub-Index	01 _h ..255 _h	
Description	Manufacturer specific	
Object Class	M: -	O: pc, pp, ip, hm, tq
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	-	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	
Data Type	user defined	

14.3.11 Object 60FC_h: position_demand_value*

This output of the *trajectory_generator* in *position_mode* is an internal value using increments as unit what is expressed with an *. To save calculation time for some applications, this object is additionally introduced to the *position_demand_value* (6062_h).

Index	60FC _h
Name	position_demand_value*
Object Code	VAR
Data type	Integer32

Value Description

Object Class	M: -	O: pc, pp, ip, hm; tq
Access	ro	
PDO Mapping	Possible	
Units	increments	
Value Range	$(-2^{31})..(2^{31}-1)$	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

14.4 Functional Description

Figure 32 shows the meaning of the subfunctions position reached and following error. Symmetrically around the target_position x_i a window is defined for the accepted position range between x_i-x_0 and x_i+x_0 . The positions x_{t0} and x_{t1} e.g. are situated within this *position_window*. If a drive is sensed in the accepted range a virtual timer can be imagined to be started. This timer is available as communication object called *position_window_time*. To change the status bit (bit 10) *target_reached* it is necessary that the *position_window_time* expires while the drive position is still in this range between x_i-x_0 and x_i+x_0 .

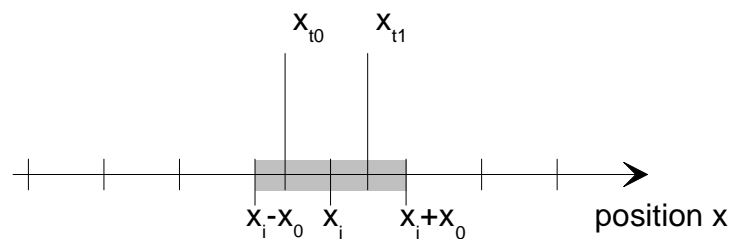


Figure 33: Position reached

Another position area - the *following_error_window* - may be defined to mark a *following_error*. In this case the *following_error_time_out* is used to specify the time the *actual_position* may be outside the *position_window* without a *following_error* to be stated in the *statusword* at bit 13.

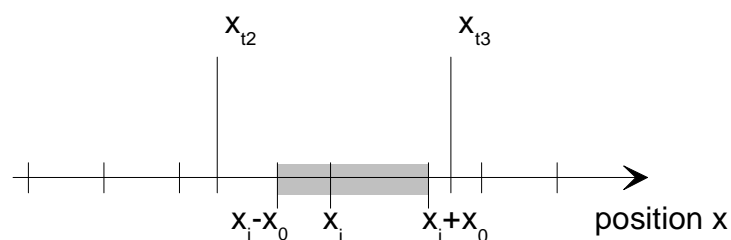


Figure 34: Following error

15 Interpolated Position Mode

15.1 General Information

The Interpolated Position Mode is used to control multiple coordinated axes or a single axle with the need for time-interpolation of setpoint data. The Interpolated Position Mode normally uses time synchronization mechanisms like the sync object defined in /16/ for a time coordination of the related drive units.

The *interpolation_data_record* contains the interpolation data; the data type of the members of this structure is not detailed by the CANopen Device Profile for Drives and Motion Control. Only the record size is fixed in the *size_of_data_record* as sub-index of the *interpolation_data_configuration*

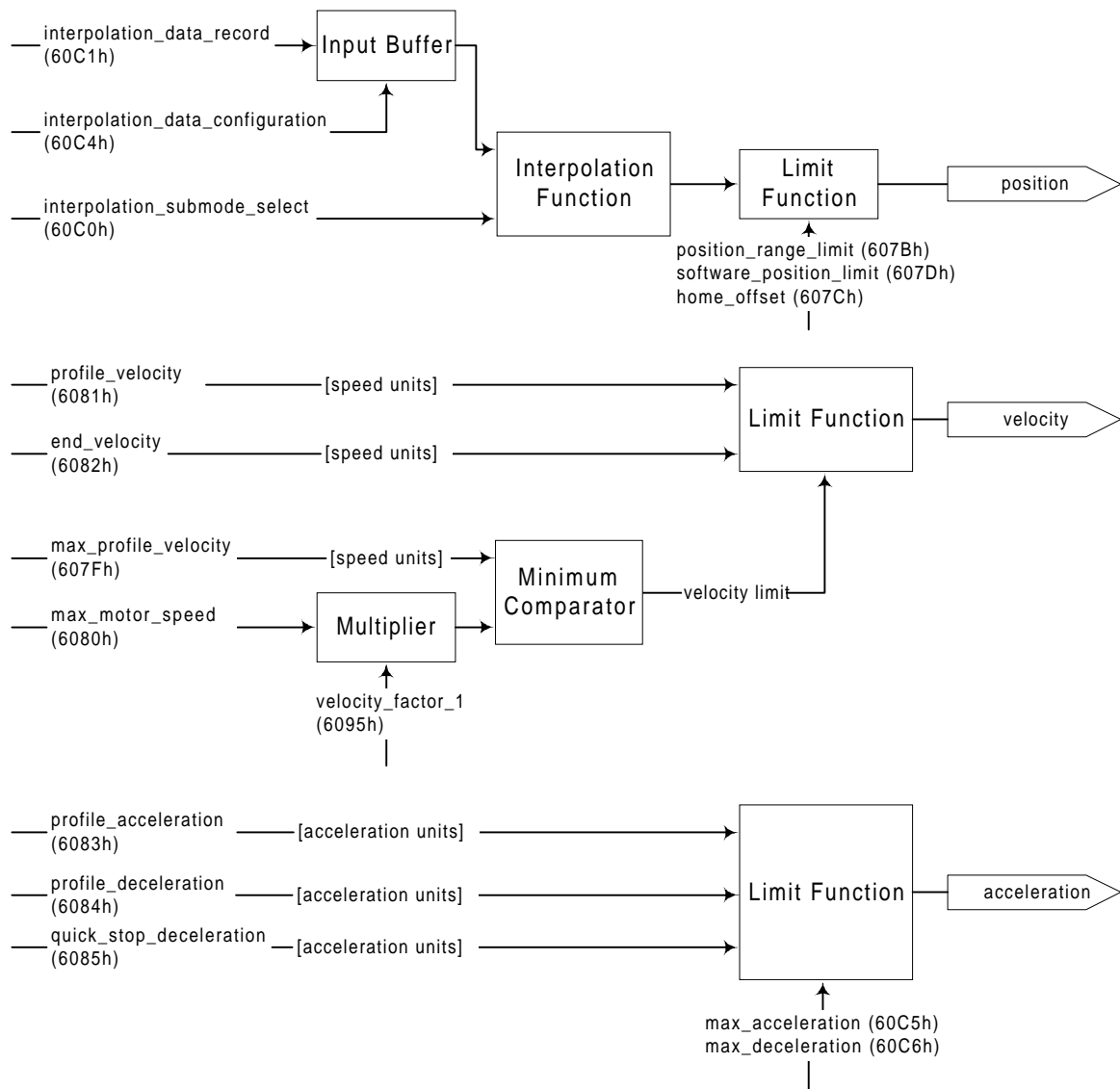
For synchronous operation the interpolation cycle time is defined by the object *interpolation_time_period*. For asynchronous operation, the *interpolation_time_period* for each time slice must be included in the *interpolation_data_record*.

Time synchronization can be done by the CANopen sync message (see /16/), a specific group sync signal (broadcast) or in specified time slices which are activated with the start signal.

The Interpolated Position Mode allows a host controller to transmit a stream of interpolation data with either an implicit or explicit time reference to a drive unit. If the drive supports an input buffer, the interpolation data may be sent in bursts rather than continuously in real time. The actually available and the maximum size of the input buffer can be requested by a host using the *interpolation_data_configuration*. The buffer size is the number of *interpolation_data_records* which may be sent to a drive to fill the input buffer and it is not the size in bytes. Devices without input buffer capabilities can only but must accept at least one interpolation data item.

The interpolation algorithm is defined in the *interpolation_submode_select*. Linear interpolation is the default interpolation method. This requires only one interpolation data item to be buffered for the calculation of the next demand value. For each interpolation cycle, the drive will calculate a *position_demand_value* by interpolating positions over a period of time.

Optionally the common limit functions for speed, acceleration and deceleration may be applied to the interpolation data.



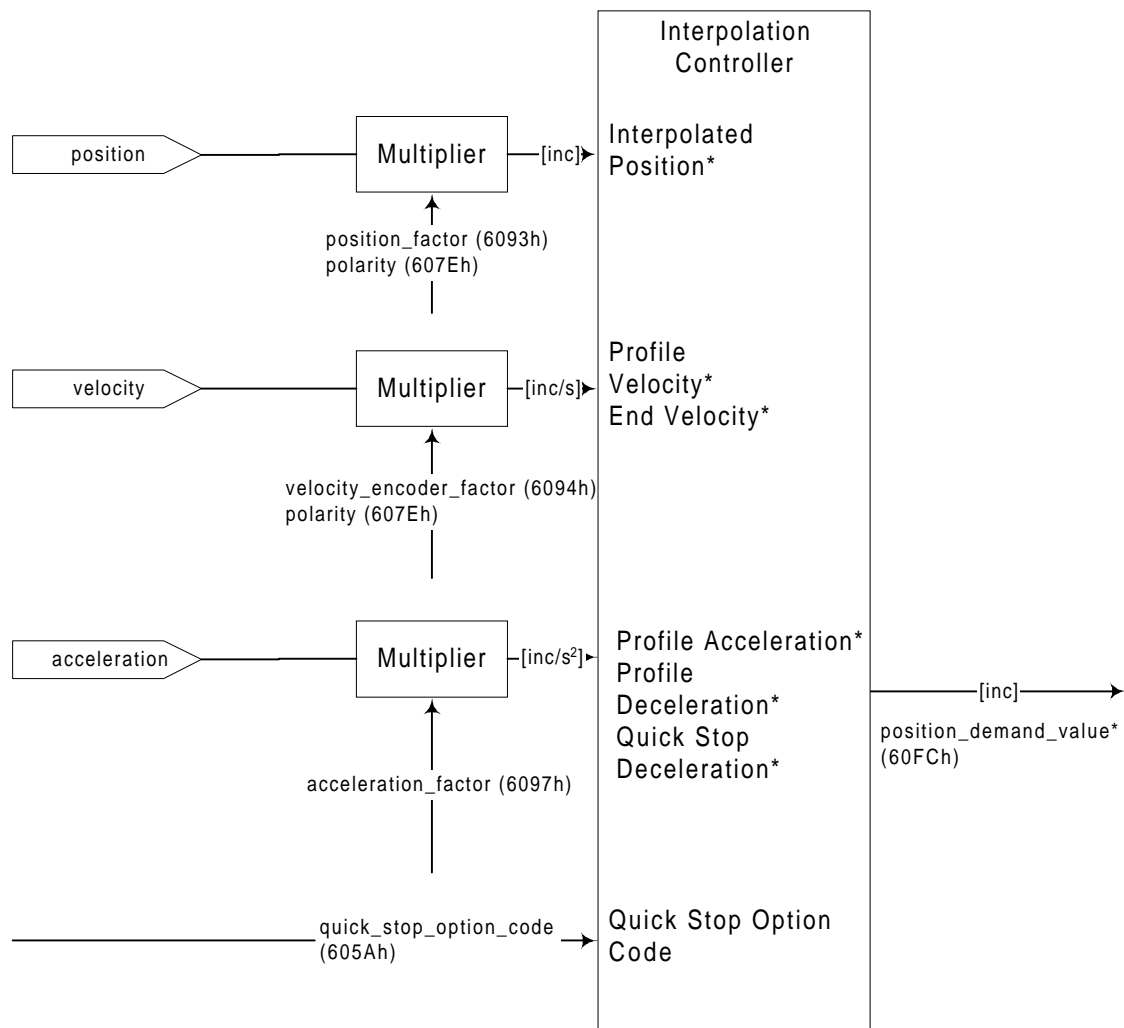


Figure 35: Interpolation Controller

15.1.1 Input Data Description

Operating Mode	Input Parameters Used
Interpolated Position Mode	interpolation_submode_select, max_profile_velocity, profile_acceleration, profile_deceleration, quick_stop_deceleration*, quick_stop_mode

15.1.2 Output Data Description

The output values provided by the Interpolated Position Mode depend on the number and type of interpolation functions implemented by a manufacturer. For the predefined linear time interpolation the output is a *position_demand_value**.

Operation Mode	Output Parameter used
Interpolated Position Mode	position_demand_value*

15.1.3 Internal States

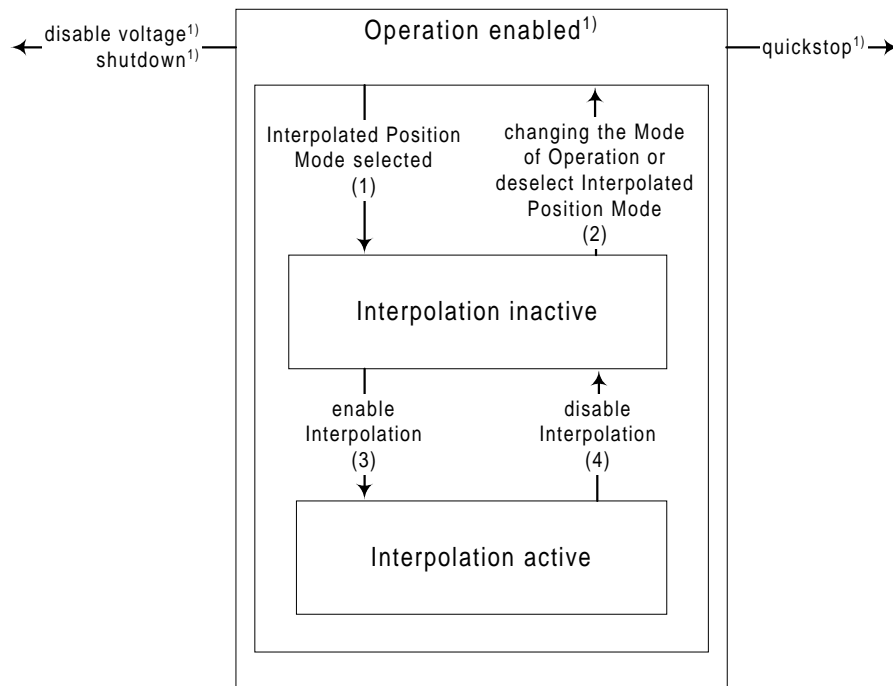


Figure 36: Internal States for the Interpolated Position Mode

¹⁾see statemachine

- Interpolation inactive
This state is entered when the device is in state Operation enabled and the Interpolated Position Mode is selected. The drive unit will accept input data and will buffer it for interpolation calculations, but it does not move the axes.
- Interpolation active
This state is entered when the device is in state Operation enabled, the Interpolated Position Mode is selected and enabled. The drive unit will accept input data and it moves the axes.

15.1.4 State Transitions of the Internal States

State Transition 1: NO IP-MODE SELECTED => IP-MODE INACTIVE
Event: Select ip-mode with *controlword* while operation enabled

State Transition 2: IP-MODE INACTIVE => NO IP-MODE SELECTED
Event: Select any other mode *controlword* while operation enabled

State Transition 3: IP-MODE INACTIVE => IP-MODE ACTIVE
Event: Set bit enable_ip_mode (bit4) of the *controlword* while in ip-mode and operation enabled

State Transition 4: IP-MODE ACTIVE => IP-MODE INACTIVE

Event: Reset bit enable_ip_mode (bit4) of the *controlword* while in ip-mode and operation enabled

15.2 Object Dictionary Entries

15.2.1 Objects defined in this Chapter

Index	Object	Name	Type	Attr.	M/O
60C0 _h	VAR	interpolation_submode_select	Integer16	rw	O
60C1 _h	RECORD	interpolation_data_record	Integer32	rw	O
60C2 _h	RECORD	interpolation_time_period	Unsigned8	rw	O
60C3 _h	ARRAY	interpolation_sync_definition	Unsigned8	rw	O
60C4 _h	RECORD	interpolation_data_configuration	Unsigned16	rw	O

15.2.2 Objects defined in other Chapters

Index	Object	Name	Type	Chapter
6040 _h	VAR	controlword	Integer16	dc
6041 _h	VAR	statusword	Unsigned16	dc
605A _h	VAR	quick_stop_mode	Integer16	tg
6060 _h	VAR	modes_of_operation	Unsigned8	dc
6061 _h	VAR	modes_of_operation_display	Unsigned8	dc
6062 _h	VAR	position_reference_variable	Integer32	pp
6063 _h	VAR	position_encoder_actual_value	Integer32	pp
606A _h	VAR	sensor_selection_code	Unsigned8	pv
607F _h	VAR	max_profile_speed	Unsigned32	pp
6089 _h	VAR	position_notation_index	Integer8	fg
608A _h	VAR	position_dimension_index	Unsigned8	fg
608B _h	VAR	velocity_notation_index	Integer8	fg
608C _h	VAR	velocity_dimension_index	Unsigned8	fg
608D _h	VAR	acceleration_notation_index	Integer8	fg
608E _h	VAR	acceleration_dimension_index	Unsigned8	fg
608F _h	ARRAY	position_encoder_solution	Unsigned32	fg
6090 _h	ARRAY	velocity_encoder_solution	Unsigned32	fg
6091 _h	ARRAY	gear_ratio	Unsigned32	fg
6092 _h	ARRAY	feed_constant	Unsigned32	fg
6093 _h	ARRAY	position_factor	Unsigned32	fg
6094 _h	ARRAY	velocity_encoder_factor	Unsigned32	fg
6095 _h	ARRAY	velocity_factor_1	Unsigned32	fg
6098 _h	ARRAY	velocity_factor_2	Unsigned32	fg
6097 _h	ARRAY	acceleration_factor	Unsigned32	fg
60C5 _h	VAR	max_acceleration	Integer32	pp
60C6 _h	VAR	max_deceleration	Integer32	pp

15.3 Object Descriptions

15.3.1 Object 60C0_h: interpolation_submode_select

For the Interpolated Position Mode a manufacturer may offer different interpolation algorithms. This object reflects or changes the actually chosen interpolation mode.

Index	60C0 _h
Name	interpolation_submode_select
Object Code	VAR
Data type	Integer16

Value description

Object Class	M: -	O: ip
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	-32768..32767	
Mandatory Range	0	
Default Value	0	
Substitute Value	-	

Data Description

Interpolation Submode	Description
-32768..-1	manufacturer specific
0	linear interpolation
+1..+32767	reserved

15.3.2 Object 60C1_h: interpolation_data_record

The *interpolation_data_record* are the data words which are necessary to perform the interpolation algorithm. The number of data words in the record is defined by *interpolation_data_configuration*. The interpretation of the data words in *interpolation_data_record* may vary with the different possible interpolation modes as set by the *interpolation_submode_select*.

For the linear interpolation mode each interpolation data record simply can be regarded as a new position setpoint. To describe a cubic spline interpolation e.g., four or more data words are needed for the spline coefficients, and further interpolation parameters.

After the last item of an *interpolation_data_record* is written to the device's input buffer, the pointer of the buffer is automatically incremented to the next buffer position.

Index	60C1_h
Name	interpolation_data_record
Object Code	RECORD
Number of Elements	defined by <i>interpolation_data_configuration</i> (60C4 _h)

Value description

Sub-Index	01 _h	
Description	x ₁ : the first parameter of ip function f _{ip} (x ₁ , .. x _N)	
Object Class	M: -	O: ip
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	-	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	
Data Type	Interpolated Position Mode dependent	

Sub-Index (N > 1)	02 _h .. N	
Description	x _i : the i-th parameter of ip function f _{ip} (x ₁ , .. x _N)	
Object Class	M: -	O: ip
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	-	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	
Data Type	Interpolated Position Mode dependent	

15.3.3 Object 60C2_h: interpolation_time_period

The *interpolation_time_period* is used for time synchronized Interpolated Position Modes.

Index	60C2 _h
Name	interpolation_time_period
Object Code	RECORD
Number of Elements	2

Value description

Sub-Index	01 _h	
Description	ip_time_units	
Object Class	M: -	O: ip
Access	rw	
PDO Mapping	Possible	
Units	10 ^{ip_time_index} • seconds	
Value Range	1..255	
Mandatory Range	1	
Default Value	1	
Substitute Value	-	
Data Type	Unsigned8	

Sub-Index	02 _h	
Description	ip_time_index	
Object Class	M: -	O: ip
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	63..-128	
Mandatory Range	-3 (= milli second)	
Default Value	-3	
Substitute Value	-	
Data Type	Integer8	

15.3.4 Object 60C3_h: interpolation_sync_definition

Devices in the Interpolated Position Mode often interact with other devices. Therefore it is necessary to define a communication object which is used to synchronize these interactions.

This can be done by the general sync-signal as described in /16/, or a specific group-sync-signal. Each reception of this trigger-signal or a specified number of occurrences of the trigger-signal can synchronize the devices; a second opportunity is to use fixed time slices for synchronisation.

Index	60C3_h
Name	interpolation_sync_definition
Object Code	ARRAY
Number of Elements	2
Data Type	Unsigned8

Value description

Sub-Index	01 _h	
Description	synchronize on group	
Object Class	M: -	O: ip
Access	rw	
PDO Mapping	Possible	
Units	number	
Value Range	0..255	
Mandatory Range	-	
Default Value	0: general sync is used	
Substitute Value	-	

Sub-Index	02 _h	
Description	ip_sync every n event	
Object Class	M: -	O: ip
Access	rw	
PDO Mapping	Possible	
Units	counts	
Value Range	0..255	
Mandatory Range	-	
Default Value	1 (= each)	
Substitute Value	-	

15.3.5 Object 60C4_h: interpolation_data_configuration

It is possible to offer different algorithms of interpolation. Most of them need a larger number of position to calculate the actual position the axes should reach. To enable the device to receive the needed data in advance a data space is used to store the positions and further data sent by the host.

Index	60C4 _h
Name	interpolation_data_configuration
Object Code	RECORD
Number of Elements	6

Value description

Sub-Index	01 _h	
Description	max_buffer_size	
Object Class	M: -	O: ip
Access	ro	
PDO Mapping	Possible	
Units	number	
Value Range	0..(2 ³² -1)	
Mandatory Range	1	
Default Value	1	
Substitute Value	-	
Data Type	Unsigned32	

Sub-Index	02 _h	
Description	actual_size	
Object Class	M: -	O: ip
Access	rw	
PDO Mapping	Possible	
Units	number	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	
Data Type	Unsigned32	

Sub-Index	03 _h	
Description	buffer_organisation	
Object Class	M: -	O: ip
Access	rw	
PDO Mapping	Possible	
Units	number	
Value Range	0..255	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	
Data Type	Unsigned8	

Data Description

buffer_organisation	description
0	FIFO-buffer
1	ring-buffer
all others	reserved

Sub-Index	04 _h	
Description	buffer_position	
Object Class	M: -	O: ip
Access	rw	
PDO Mapping	Possible	
Units	number	
Value Range	0..65535	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	
Data Type	Unsigned16	

Sub-Index	05 _h	
Description	size_of_data_record	
Object Class	M: -	O: ip
Access	wo	
PDO Mapping	Possible	
Units	number	
Value Range	1..255	
Mandatory Range	-	
Default Value	1	
Substitute Value	-	
Data Type	Unsigned8	

Sub-Index	06 _h	
Description	buffer_clear	
Object Class	M: -	O: ip
Access	wo	
PDO Mapping	Possible	
Units	-	
Value Range	0, 1	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	
Data Type	Unsigned8	

Data Description

buffer_clear	description
0	clear input buffer/access disabled clear all ip_data_records
1	enable access to the input buffer for the drive functions
all others	reserved

15.4 Functional Description

15.4.1 Interpolated Position Mode

A drive can be controlled and supervised by the *controlword* and the *statusword* respectively. To choose the operation mode, the *modes_of_operation* is used. The activated operation mode is monitored by *modes_of_operation_display*.

A drive manufacturer has to specify the way the device handles a just received interpolation data record. This can be in a way corresponding to the standard position mode, or might be a more complex algorithm. The standard method is to apply new data immediatly, respectively after the next sync-signal.

An input buffer for interpolation data records is not mandatory, although it eases the data exchange between a host and a drive unit. The real-time requirements to the CAN-bus as well as to the drive unit decrease in this case, because an input buffer decouples the data processing in the drive from the data transmission via the bus line.

15.4.2 Linear Interpolated Position Mode with several Axles

In order to follow a two- or more-dimensional curve through the space with a defined speed, a host (an interpolation controller or a PLC) calculates the different positions P_i for each set of coordinates which have to be reached at specified times t_i .

To use the interpolation mode with several axles the host calculates the next or more positions and timestamps, and transmits them to the different axles. For each setpoint P_i the interpolation controller has to calculate x_i , y_i ... and t_i . Each axle gets a set of *interpolation_data_records* which each axle has to process internally independent from the other axles according to the choosen interpolation mode.

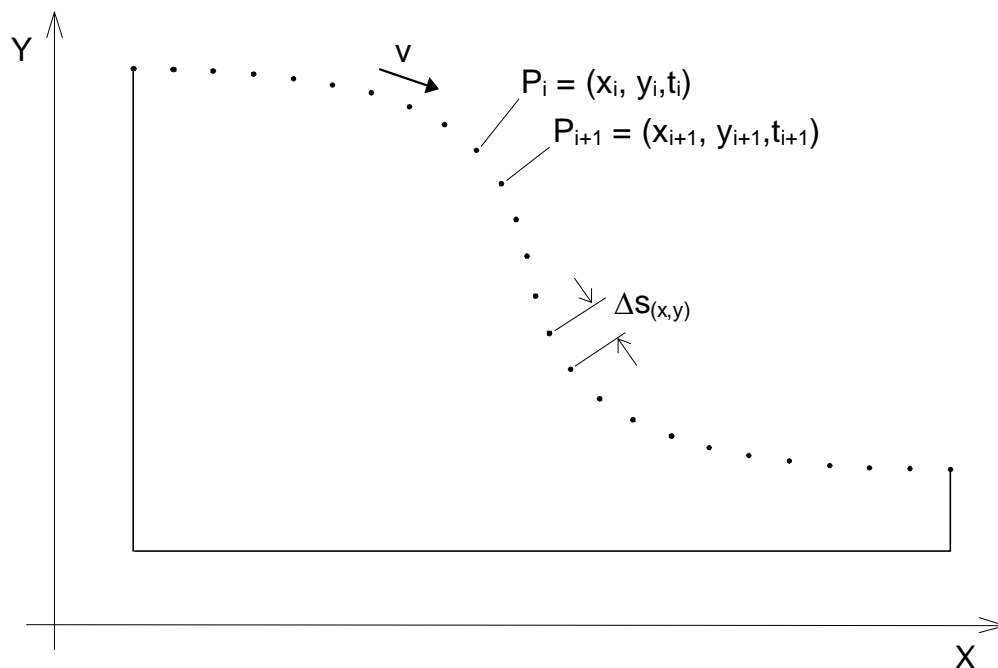


Figure 37: Interpolation for two Axes

In a centralized drive system with a remote motion device doing the interpolation calculation, a central clocking scheme for synchronization of the different axes based on any kind of sync-signal is used. This results in a movement depending on the calculation cycle time of the interpolation controller. The velocity becomes more or less a fixed value for each axle.

calculated positions	ip_data_records for		
	x-axle	y-axle	z-axle
P_i	x_i, t_i	y_i, t_i	z_i, t_i
P_{i+1}	x_{i+1}, t_{i+1}	y_{i+1}, t_{i+1}	z_{i+1}, t_{i+1}
P_{i+2}	x_{i+2}, t_{i+2}	y_{i+2}, t_{i+2}	z_{i+2}, t_{i+2}
P_{i+3}	x_{i+3}, t_{i+3}	y_{i+3}, t_{i+3}	z_{i+3}, t_{i+3}
•	•	•	•
•	•	•	•
•	•	•	•
P_{i+n}	x_{i+n}, t_{i+n}	y_{i+n}, t_{i+n}	z_{i+n}, t_{i+n}

Table 13: Position Calculation in Interpolated Position Mode for several Axes

In decentralized motion systems a host starts all relevant axes by changing the mode-internal state to Interpolation active after preparing and sending one or more *interpolation_data_records* to all axes and synchronizes them by a (group) sync-signal. Each axle calculates internally and independently the necessary speed and acceleration needed to move from one position to the next. This can be done by calculating a linear or any other move between two given position setpoints. Along this track every axle controls the movement between the setpoints independently from the other axes. The axes may

continue their move, as long as there is enough data to continue the calculations. Therefore it is easy to use the input buffer to give data records ahead.

With this information each axle can act like it is shown in Figure 37.

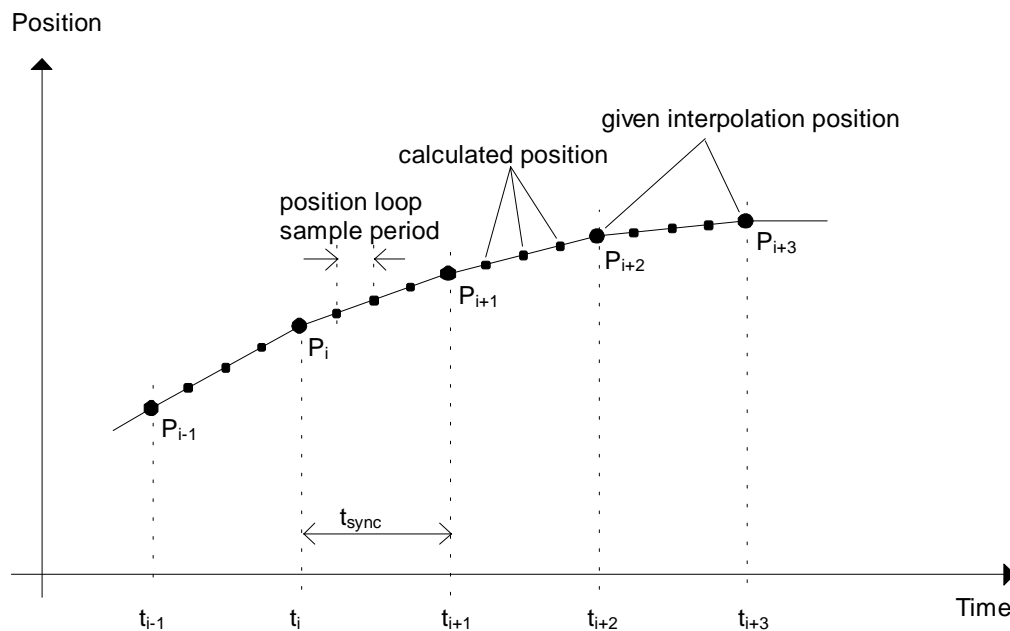


Figure 38: Linear Interpolation for one Axle

15.4.3 Buffer Strategies for the Interpolated Position Mode

If a device provides an input buffer for *interpolation_data_records* its size can be organized by a host using the *interpolation_data_configuration*. The host splits the available buffer capacity into pages which have the size of one *interpolation_data_record* each. This is done by *size_of_data_record*. If one page remains, which can not keep one complete data record, it can not be used. After the reorganisation of the input buffer all previous stored data will be lost. All devices supporting the Interpolated Position Mode need to implement an input buffer, which at least can keep one *interpolation_data_record*.

The contents of the buffer items can only be accessed via the *interpolation_data_record*.

Commonly first-in-first-out (FIFO) structures or ring buffers are used as input buffers.

○ FIFO:

If the buffer is organized as FIFO, every new received *interpolation_data_record* is placed at the end of the queue, and the device takes the next data record from the top of the queue. When the last item of a data record is stored, the buffer pointer is incremented in order to point to the next buffer position. For this buffer principle the object *buffer_position* does not have any influence.

○ Ring Buffer:

If the buffer is structured as a ring, the host can place an *interpolation_data_record* into

any valid position in the ring by changing the pointer defined in *buffer_position*. Without changing the *buffer_position* all data records will be written at the same location. The drive reads the next entry out of the buffer by an internal ring pointer. It is set to the first data record with *buffer_clear*, and after the reorganisation of the input buffer.

↑ data_ record_ size ↓	parameter 1 parameter 2 • • parameter n	<i>ip_data_record 1</i>
↑ data_ record_ size ↓	parameter 1 parameter 2 • • parameter n	<i>ip_data_record 2</i>
		more data records
		•
		•
		•
		•
↑ data_ record_ size ↓	parameter 1 parameter 2 • • parameter n	<i>ip_data_record i</i>
		not accessible

Figure 39: Input Buffer Organisation

16 Profile Velocity Mode

16.1 General Information

The Profile Velocity Mode includes the following subfunctions:

- demand value input via Trajectory Generator
- velocity capture using position sensor or velocity sensor
- velocity control function with appropriate input and output signals
- limitation of *torque_demand_value*
- monitoring of the *profile_velocity* using a window-function
- monitoring of *velocity_actual_value* using a threshold

The operation of the reference value generator and its input parameters:

- *profile_velocity* ;
- *profile_acceleration* ;
- *profile_deceleration*;
- *emergency_stop* and
- *motion_profile_type*

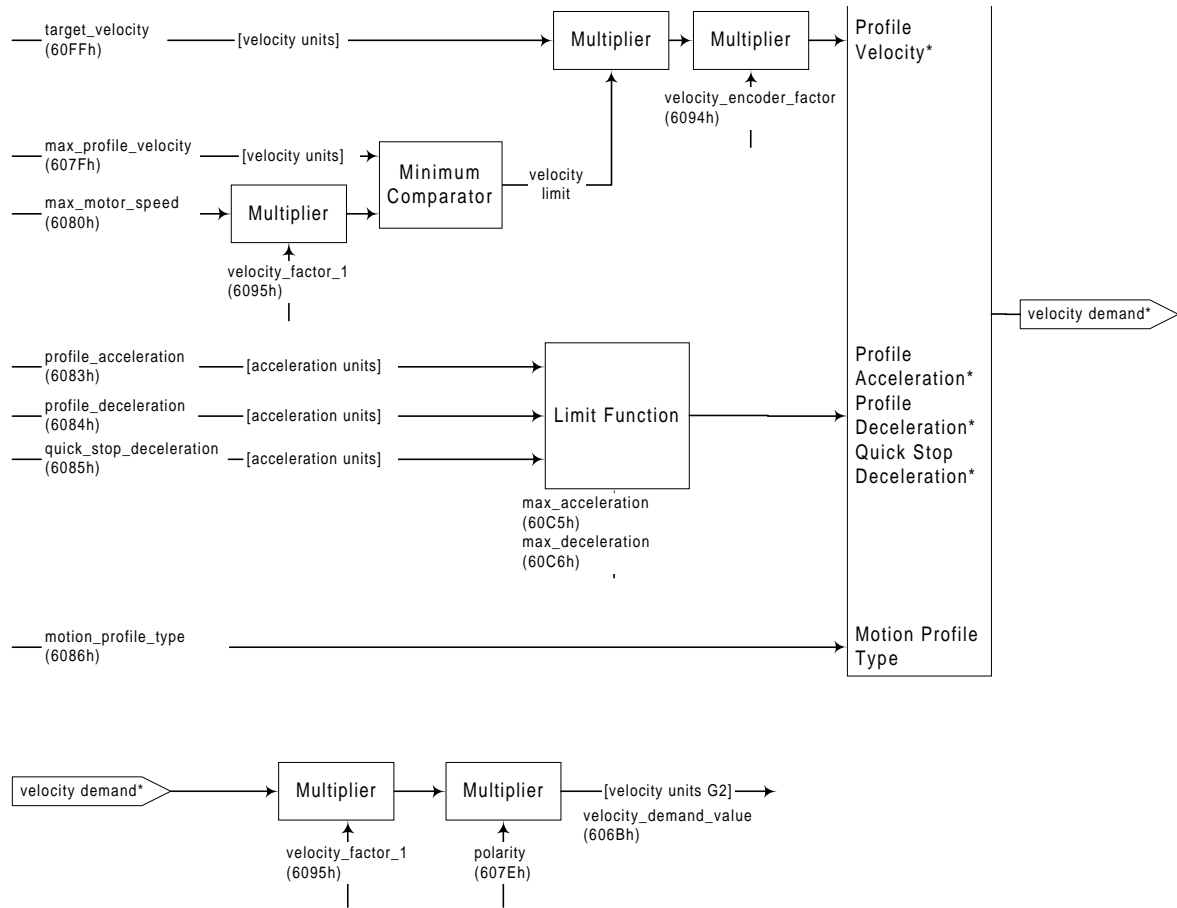
are described in the Profile Position Mode.

Various sensors can be used for velocity capture. In particular the aim is that costs should be reduced and the system should be simplified by evaluating position and velocity using a common sensor, such as is possible using a resolver or an encoder.

The velocity control function is not specified more precisely at this point as it is highly manufacturer specific, but the format and maximum number of control coefficients are established.

The velocity controller calculates a torque variable. This is added to a torque pre control calculated by the Trajectory Generator and limited to a *torque_max_value*. The limited total is used as input to the torque controller as a *torque_demand_value*.

Monitoring functions for the *velocity_actual_value* provide status information for super-ordinated systems.



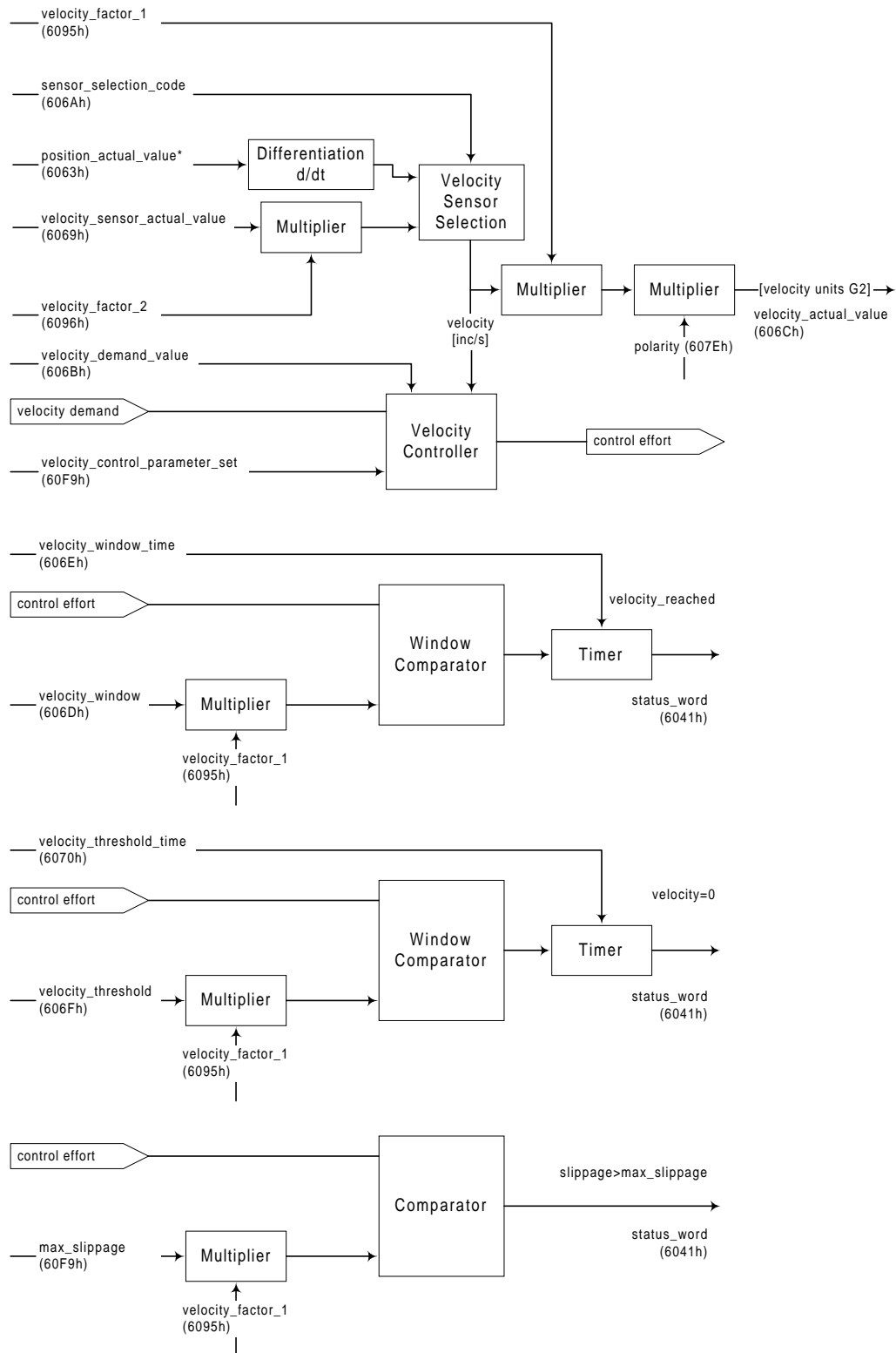


Figure 40: Structure of the Profile Velocity Mode

16.1.1 Input Data Description

Operating Mode	Input Parameters Used
Profile Velocity Mode	target_velocity, velocity_factor_1, velocity_factor_2, velocity_window, velocity_window_time, velocity_threshold, velocity_threshold_time, max_slippage, profile_acceleration, profile_deceleration, quick_stop_deceleration, max_acceleration, max_deceleration, polarity, quick_stop_option_code, motion_profile_type, max_profile_velocity, max_motor_speed

16.1.2 Output Data Description

Operation Mode	Output Parameter used
Profile Velocity Mode	velocity_actual_value, velocity_demand_value, statusword

16.2 Object Dictionary Entries

16.2.1 Objects defined in this Chapter

Index	Object	Name	Type	Attr.	M/O
6069 _h	VAR	velocity_sensor_actual_value	Integer32	ro	M
606A _h	VAR	sensor_selection_code	Integer16	rw	M
606B _h	VAR	velocity_demand_value	Integer32	ro	M
606C _h	VAR	velocity_actual_value	Integer32	ro	M
606D _h	VAR	velocity_window	Unsigned16	rw	O
606E _h	VAR	velocity_window_time	Unsigned16	rw	O
606F _h	VAR	velocity_threshold	Unsigned16	rw	O
6070 _h	VAR	velocity_threshold_time	Unsigned16	rw	O
60FF _h	VAR	target_velocity	Integer32	rw	M
60F8 _h	VAR	max_slippage	Integer32	rw	O
60F9 _h	RECORD	velocity_control_parameter_set		rw	O

16.2.2 Objects defined in other Chapters

Index	Object	Name	Type	Chapter
6040 _h	VAR	controlword	Integer16	dc
6041 _h	VAR	statusword	Unsigned16	dc
6063 _h	VAR	position_actual_value*	Integer32	pc
6069 _h	VAR	velocity_sensor_actual_value	Integer32	pc
6071 _h	VAR	torque_demand_extern	Integer16	tq
6072 _h	VAR	max_torque_value	Unsigned16	tq
607E _h	VAR	polarity	Unsigned8	fg
607F _h	VAR	max_profile_velocity	Unsigned32	pp
6080 _h	VAR	max_motor_speed	Unsigned32	pp
6083 _h	VAR	profile_acceleration	Unsigned32	pp
6084 _h	VAR	profile_deceleration	Unsigned32	pp
6085 _h	VAR	quick_stop_deceleration	Unsigned32	pp
6086 _h	VAR	motion_profile_type	Integer16	pp
6094 _h	ARRAY	velocity_encoder_factor	Unsigned32	fg
6095 _h	ARRAY	velocity_factor_1	Unsigned32	fg
6096 _h	ARRAY	velocity_factor_2	Unsigned32	fg

16.3 Object Description

The factors necessary for scaling

velocity_reference_factor
velocity_factor_1
velocity_factor_2

all have a linear relationship and are therefore described in the Factor Group.

The *polarity* is described in the Factor Group as well.

16.3.1 position_encoder

The actual velocity can be obtained through differentiation from the position encoder and is represented in position encoder increments.

It is described in greater detail in the position function.

16.3.2 Object 6069_h: velocity_sensor_actual_value

The *velocity_sensor_actual_value* describes the value read from a velocity encoder (if present) in increments (in the case of encoders) and in increments per second (in the case of tachometers and AD converters). This value is scaled to the format of the position encoder using the scaling factor *velocity_factor_2*.

Index	6069 _h
Name	velocity_sensor_actual_value
Object Code	VAR
Data Type	Integer32

Value Description

Object Class	M: pv	O: -
Access	ro	
PDO Mapping	Possible	
Units	increments/sec	
Value Range	$(-2^{31})..(2^{31}-1)$	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

16.3.3 Object 606A_h: sensor_selection_code

The source of the *velocity_sensor_actual_value* can be determined using the *sensor_selection_code*. This determines whether a differentiated position signal or the signal from a separate velocity sensor is to be evaluated.

Index	606A _h
Name	sensor_selection_code
Object Code	VAR
Data Type	Integer16

Value Description

Object Class	M: pv	O: -
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	-32768..32767	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

Data Description

Selection Code	Meaning of the Selection Function
-32768..-1	manufacturer specific
0	velocity actual value from position encoder
1	velocity actual value from velocity encoder
2..32767	reserved for other profiles

16.3.4 Object 606B_n: velocity_demand_value

The output value of the Trajectory Generator may be corrected by the output value of the Position Control Function. It is then provided as a demand value for the velocity controller.

Index	606B _n
Name	velocity_demand_value
Object Code	VAR
Data Type	Integer32

Value Description

Object Class	M: pv	O:
Access	ro	
PDO Mapping	Possible	
Units	velocity units G2	
Value Range	$(-2^{31})..(2^{31}-1)$	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

16.3.5 Object 606C_n: velocity_actual_value

The *velocity_actual_value* is also represented in velocity units and is coupled to the velocity used as input to the velocity controller.

Index	606C _n
Name	velocity_actual_value
Object Code	VAR
Data Type	Integer32

Value Description

Object Class	M: pv	O: -
Access	ro	
PDO Mapping	Possible	
Units	velocity units G2	
Value Range	$(-2^{31})..(2^{31}-1)$	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

16.3.6 Object 606D_n: velocity_window

The *velocity_window* monitors whether the required process velocity has been achieved after an eventual acceleration or braking phase.

Index	606D _n
Name	velocity_window
Object Code	VAR
Data Type	Unsigned16

Value Description

Object Class	M: -	O: pv
Access	rw	
PDO Mapping	Possible	
Units	velocity units G2	
Value Range	0..65535	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

16.3.7 Object 606E_h: velocity_window_time

The corresponding bit 10 *target_reached* is set in the *statusword* when the difference between the *target_velocity* and the *velocity_actual* is within the *velocity_window* longer than the *velocity_window_time*.

Index	606E _h
Name	velocity_window_time
Object Code	VAR
Data Type	Unsigned16

Value Description

Object Class	M: -	O: pv
Access	rw	
PDO Mapping	Possible	
Units	millisecond	
Value Range	0..65535	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	

16.3.8 Object 606F_h: velocity_threshold

As soon as the *velocity_actual_value* exceeds the *velocity_threshold* longer than the *velocity_threshold_time* bit 12 *velocity* = 0 is reset in the status word. Below this threshold the bit is set and indicates that the axle is stationary.

Index	606F _h
Name	velocity_threshold
Object Code	VAR
Data Type	Unsigned16

Value Description

Object Class	M: -	O: pv
Access	rw	
PDO Mapping	Possible	
Units	velocity units G2	
Value Range	0..65535	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

16.3.9 Object 6070_h: velocity_threshold_time

Index	6070_h
Name	velocity_threshold_time
Object Code	VAR
Data Type	Unsigned16

Value Description

Object Class	M: -	O: pv
Access	r/w	
PDO Mapping	Possible	
Units	millisecond	
Value Range	0..65535	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

16.3.10 Object 60FF_h: target_velocity

The *target_velocity* is the input for the Trajectory Generator.

Index	60FF_h
Name	target_velocity
Object Code	VAR
Data Type	Integer32

Value Description

Object Class	M: pv	O:
Access	rw	
PDO Mapping	Possible	
Units	velocity units G2	
Value Range	$(-2^{31})..(2^{31}-1)$	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

16.3.11 Object 60F8_h: max_slippage

The *max_slippage* monitors whether the maximal slippage has actually been reached.

This value is scaled to the format of the position encoder using the scaling factor *velocity_factor_2*.

When the *max_slippage* has been reached, the corresponding bit 13 *max_slippage_error* in the status message will be set to one. The reaction of the drive when the *max_slippage* error occurs, is manufacturer specific.

Index	60F8_h
Name	max_slippage
Object Code	VAR
Data Type	Integer32

Value Description

Object Class	M: pv	O: -
Access	ro	
PDO Mapping	Possible	
Units	velocity units G2	
Value Range	$(-2^{31})..(2^{31}-1)$	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	

16.3.12 Object 60F9_h: velocity_control_parameter_set

In order to control the behaviour of the velocity control loop, one or more parameters are necessary. This object defines a rudimentary set of three parameters for a PID-control which may be enlarged by the manufacturer up to 255 parameters.

Index	60F9_h
Name	velocity_control_parameter_set
Object Code	RECORD
Number of Elements	3..255

Value Description

Sub-Index	01 _h	
Description	V: gain	
Object Class	M: -	O: pv
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..65535	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	
Data Type	unsigned16	

Sub-Index	02 _h	
Description	T _i : integration time constant	
Object Class	M: -	O: pv
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..65535	
Mandatory Range	-	
Default Value	-	
Substitute Value	-	
Data Type	unsigned16	

17 Profile Torque Mode

17.1 General Information

This chapter describes the Profile Torque Mode. The Profile Torque Mode allows a host (external) control system (i.e. closed-loop speed controller, open-loop transmission force controller) to transmit the *target_torque* value, which is processed via the Trajectory Generator. The *torque_slope* and *torque_profile_type* parameters are required.

Should the host control system switch the *controlword* bit 8 (hold) from 0 to 1 or from 1 to 0, then the Trajectory Generator ramps its *control_effort* output down to zero, respectively up to the *target_torque*. In both cases the Trajectory Generator takes the *torque_slope* and *torque_profile_type* into consideration.

All definitions within this document refer to rotating motors. Using linear motors instead requires that all "torque" objects refer to a "force" instead. For the sake of simplicity, the objects are not duplicated and their names should not be modified. As an example, the linear motor target force must be transmitted using the *target_torque* object. Refer to the object descriptions for additional information.

The manufacturer-specific torque control and power-stage functions are not described as they fall beyond the scope of this standard. They are only mentioned for showing how some parameters affect them. As an example the closed-loop torque control coefficients (if any) are to be defined and described by the manufacturer.

The *torque_control_parameters*, *power_stage_parameters* and *motor_parameters* are defined as objects in order that they can be handled (i.e. downloaded) in a standard way. Their detailed data content is manufacturer-specific.

The *torque_demand*, *torque_actual_value*, *current_actual_value* and *DC_link_voltage* may be available to the user as parameters, if they are monitored.

Depending on the drive and motor technologies the manufacturer-specific torque control function has to be active when another mode is selected (hc, pv, pc, ip). In such a case, selecting one of these modes implicitly activates the torque control and power-stage function, using the *control_effort* as input.

17.1.1 Structure of the Profile Torque Mode

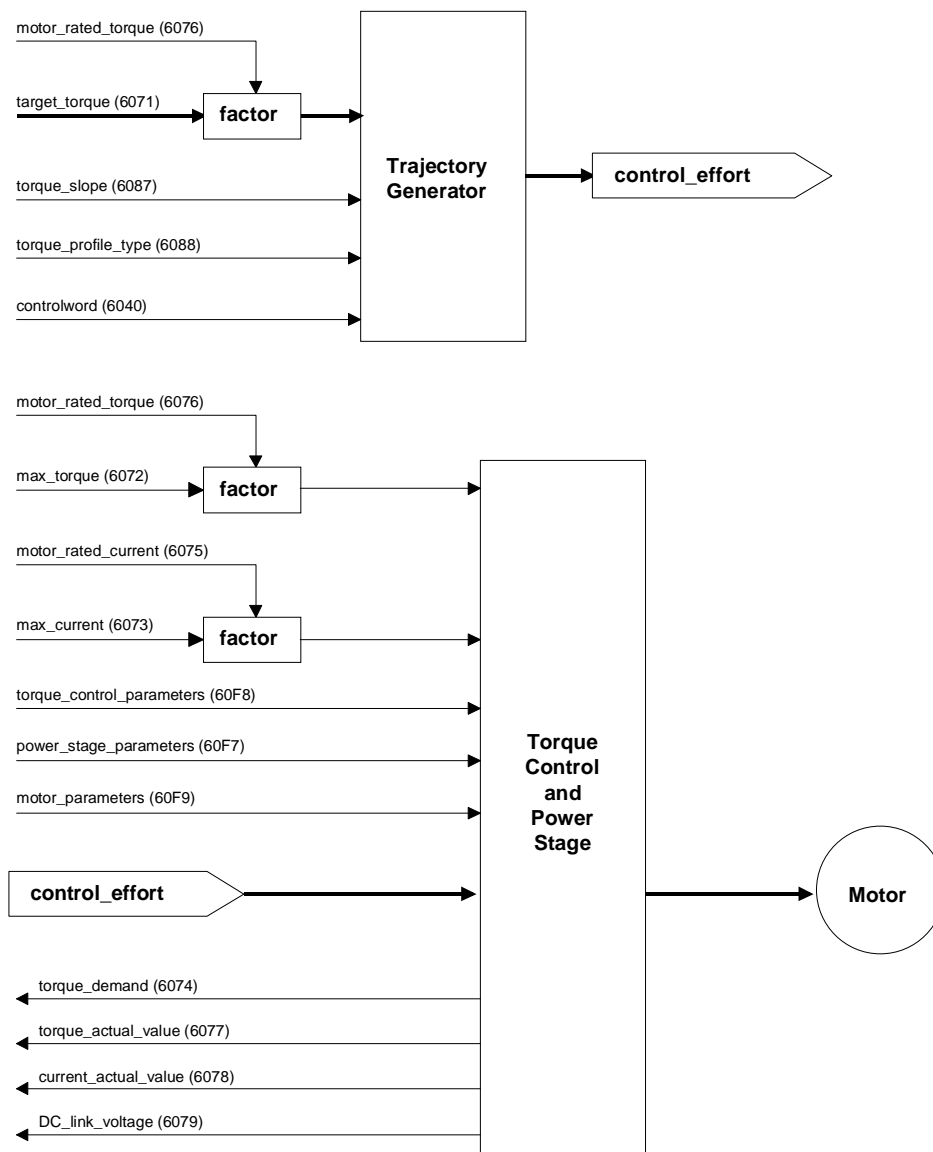


Figure 41: Structure of the Profile Torque Mode

17.2 Object Dictionary Entries

17.2.1 Objects defined in this Chapter

Index	Object	Name	Type	Attr.	M/O
6071 _h	VAR	target_torque	Integer16	rw	M
6072 _h	VAR	max_torque	Unsigned16	rw	O
6073 _h	VAR	max_current	Unsigned16	rw	O
6074 _h	VAR	torque_demand_value	Integer16	ro	O
6075 _h	VAR	motorRatedCurrent	Unsigned32	rw	O
6076 _h	VAR	motorRatedTorque	Unsigned32	rw	O
6077 _h	VAR	torqueActualValue	Integer16	ro	O
6078 _h	VAR	currentActualValue	Integer16	ro	O
6079 _h	VAR	DCLinkCircuitVoltage	Unsigned32	ro	O
6087 _h	VAR	torqueSlope	Unsigned32	rw	M
6088 _h	VAR	torqueProfileType	Integer16	rw	M
60F7 _h	RECORD	powerStageParameters		rw	O
60F6 _h	RECORD	torqueControlParameters		rw	O

17.2.2 Objects defined in other chapters

Index	Object	Name	Type	Chapter
6040 _h	VAR	controlword	Integer16	dc
60F9 _h	RECORD	motorParameters		go

17.3 Object Description

17.3.1 Object 6071_h: target_torque

This parameter is the input value for the torque controller in Profile Torque Mode.

Index	6071 _h
Name	target_torque
Object Code	VAR
Data type	Integer16

Value description

Object Class	M: tq	O: -
Access	rw	
PDO Mapping	optional	
Units	per thousand of rated torque	
Value Range	-32768..32767	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	

17.3.2 Object 6072_h: max_torque

This value represents the maximum permissible torque in the motor.

Index	6072 _h
Name	max_torque
Object Code	VAR
Data Type	Unsigned16

Value description

Object Class	M: -	O: tq, hc, pv, pp, ip
Access	rw	
PDO Mapping	optional	
Units	per thousand of rated torque	
Value Range	0..65535	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	

17.3.3 Object 6073_h: max_current

This value represents the maximum permissible torque creating current in the motor.

Index	6073_h
Name	max_current
Object Code	VAR
Data Type	Unsigned16

Value description

Object Class	M: -	O: tq, hc, pv, pp, ip
Access	rw	
PDO Mapping	optional	
Units	per thousand of rated current	
Value Range	0..65535	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	

17.3.4 Object 6074_h: torque_demand_value

This parameter is the output value of the torque limit function (if available within the torque control and power-stage function).

Index	6074_h
Name	torque_demand_value
Object Code	VAR
Data Type	Integer16

Value description

Object Class	M: -	O: tq, hc, pv, pp, ip
Access	ro	
PDO Mapping	optional	
Units	per thousand of rated torque	
Value Range	-32768..32767	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	

17.3.5 Object 6075_h: motorRatedCurrent

This value is taken from the motor name plate and is entered as units of 1 milliamp (or 0.001 amp). Depending on the motor and drive technology this current may be either DC, peak or rms (root-mean-square) current. All relative current data refers to this value.

Index	6075 _h
Name	motorRatedCurrent
Object Code	VAR
Data Type	Unsigned32

Value description

Object Class	M: -	O: tq, hc, pv, pp, ip
Access	rw	
PDO Mapping	optional	
Units	1 mA	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	

17.3.6 Object 6076_h: motorRatedTorque

This value is taken from the motor name plate and is entered as units of 0.001 Nm. All relative torque data refer to this value.

For linear motors, the object name is not changed, but the motor rated force value must be entered as units of 0.001 N.

Index	6076 _h
Name	motorRatedTorque
Object Code	VAR
Data Type	Unsigned32

Value description

Object Class	M: -	O: tq, hc, pv, pp, ip
Access	rw	
PDO Mapping	optional	
Units	0.001 Nm	
Value Range	0..(2 ³² -1)	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	

17.3.7 Object 6077_h: torque_actual_value

The torque actual value corresponds to the instantaneous torque in the drive motor.

Index	6077_h
Name	torque_actual_value
Object Code	VAR
Data Type	Integer16

Value description

Object Class	M: -	O: tq, hc, pv, pp, ip
Access	ro	
PDO Mapping	optional	
Units	per thousand of rated torque	
Value Range	-32768..32767	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	

17.3.8 Object 6078_h: current_actual_value

The current actual value refers to the instantaneous current in the drive motor.

Index	6078_h
Name	current_actual_value
Object Code	VAR
Data Type	Integer16

Value description

Object Class	M: -	O: tq, hc, pv, pp, ip
Access	ro	
PDO Mapping	optional	
Units	per thousand of rated current	
Value Range	-32768..32767	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	

17.3.9 Object 6079_h: DC_link_circuit_voltage

This parameter describes the instantaneous DC link current voltage at the drive controller.

Index	6079_h
Name	DC_link_circuit_voltage
Object Code	VAR
Data Type	Unsigned32

Value description

Object Class	M: -	O: tq, hc, pv, pc, ip
Access	ro	
PDO Mapping	optional	
Units	milli volts	
Value Range	0..($2^{32}-1$)	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	

17.3.10 Object 6087_h: torque_slope

This parameter describes the rate of change of torque in units of per thousand of rated torque per second.

Index	6087_h
Name	torque_slope
Object Code	VAR
Data Type	Unsigned32

Value description

Object Class	M: tq	O:
Access	rw	
PDO Mapping	optional	
Units	per thousand of rated torque per second	
Value Range	0..($2^{32}-1$)	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	

17.3.11 Object 6088_h: torque_profile_type

The torque_profile_type is used to select the type of torque profile used to perform a torque change.

Index	6088_h
Name	torque_profile_type
Object Code	VAR
Data Type	Integer16

Value description

Object Class	M: tq	O:
Access	rw	
PDO Mapping	optional	
Units	none	
Value Range	-32768..32767	
Mandatory Range	-	
Default Value	0	
Substitute Value	-	

Data Description

Profile Code	Profile Type
-32768 ... -1	Vendor specific
0	Linear ramp (trapezoidal profile)
1	Sin ² ramp
2 .. 32767	Reserved for further profile types

17.3.12 Object 60F7_h: power_stage_parameters

The *power_stage_parameters* object is used to handle (i.e. download) all manufacturer-specific power-stage parameters as a whole, in a standard way.

Index	60F7_h
Name	power_stage_parameters
Object Code	RECORD
Number of Elements	1..255

Value description

Sub-Index	01 _h	
Description		
Object Class	M: -	O: tq, hc, pv, pp, ip
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range		
Mandatory Range	-	
Default Value	-	
Substitute Value	-	
Data Type		

17.3.13 Object 60F6_h: torque_control_parameters

The *torque_control_parameters* object is used to handle (i.e. download) all manufacturer-specific torque control parameters as a whole, in a standard way.

Index	60F6_h
Name	torque_control_parameters
Object Code	RECORD
Number of Elements	1..255

Value description

Sub-Index	01 _h	
Description		
Object Class	M: -	O: tq, hc, pv, pp, ip
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range		
Mandatory Range	-	
Default Value	-	
Substitute Value	-	
Data Type		

18 Velocity Mode

18.1 General Description

The Velocity Mode is based on /18/ and /19/ and refers to the Speed Function Group 1 of /19/.

The most frequently used devices with this mode are low-cost frequency inverters. But this profile could be used with all types of drives and other devices where it fits. Therefore data objects are almost 16bit wide. The calculation of variables at the drive is possible by usual 8 bit microprocessors.

Most applications use a velocity setpoint and a control word for switching the drive on and off.

Example for a minimal implementation of the Velocity Mode.

See chapter 10 for Device Control.

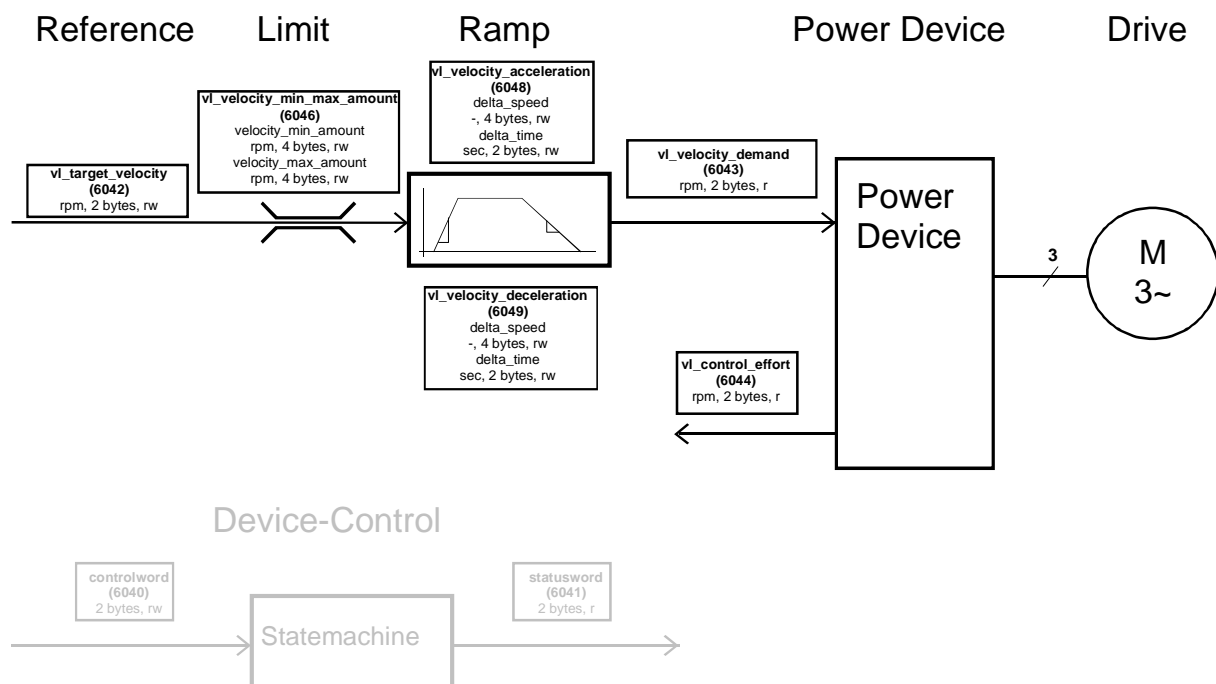


Figure 42: Example of a Velocity Mode application

18.1.1 Input Data Description

The Velocity Mode has the following input parameter:

Operating Mode	Input Parameters Used
Velocity Mode	vl_target_velocity, vl_nominal_percentage, vl_pole_number, vl_dimension_factor; vl_velocity_min_max_amount, vl_velocity_min_max, vl_velocity_motor_min_max_amount - vl_velocity_motor_min_max, vl_frequency_motor_min_max_amount, vl_frequency_motor_min_max, vl_velocity_acceleration, vl_velocity_deceleration, vl_velocity_quick_stop, vl_ramp_function_time, vl_slow_down_time, vl_quick_stop_time, vl_velocity_reference, vl_setpoint_factor

These objects are only used for the Velocity Mode.

18.1.2 Output Data Description

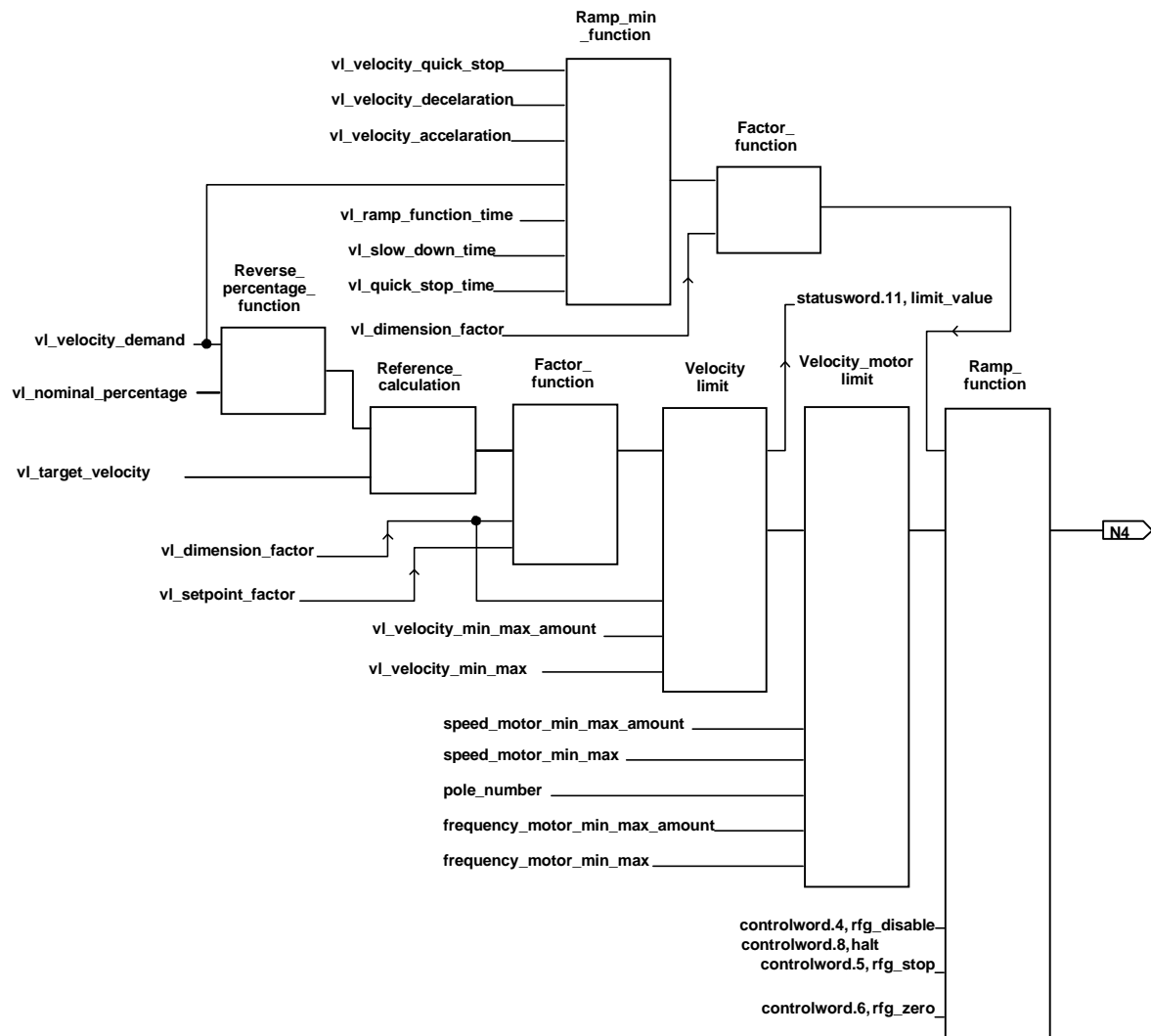
The Velocity Mode provides the following output parameters:

Operation Mode	Output Parameter used
Velocity Mode	vl_control_effort, vl_manipulated_velocity, vl_percentage_demand, vl_actual_percentage, vl_velocity_demand, vl_manipulated_percentage

These objects are only used for the Velocity Mode.

18.1.3 Structure of the Velocity Mode

The diagram below shows the overall structure of the Velocity Mode. All mandatory and optional objects are used. It is not intended with it to specify implementations, but to describe the scope of functions. In these structures, the unit in which the velocity values in the speed functions are calculated is rpm. The descriptions of the drive functions refer to this structure. The Device Control is of course used in the Velocity Mode, but it is described in an extra chapter.



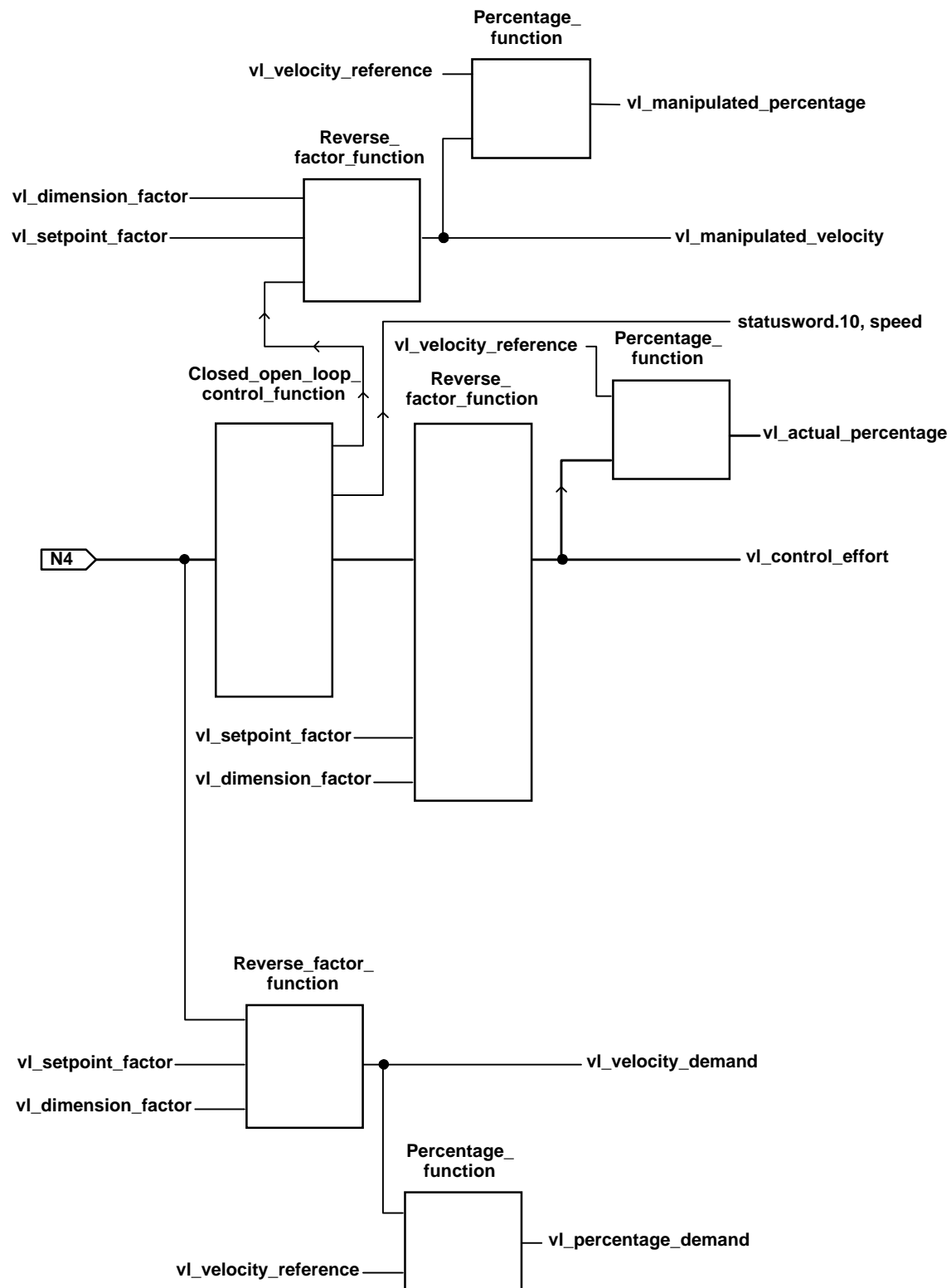


Figure 43: Velocity Mode with all Objects

All device using this profile and supporting the Velocity Mode have to implement the mandatory objects and there functionality. The diagram below shows the structure all devices will have at minimum.

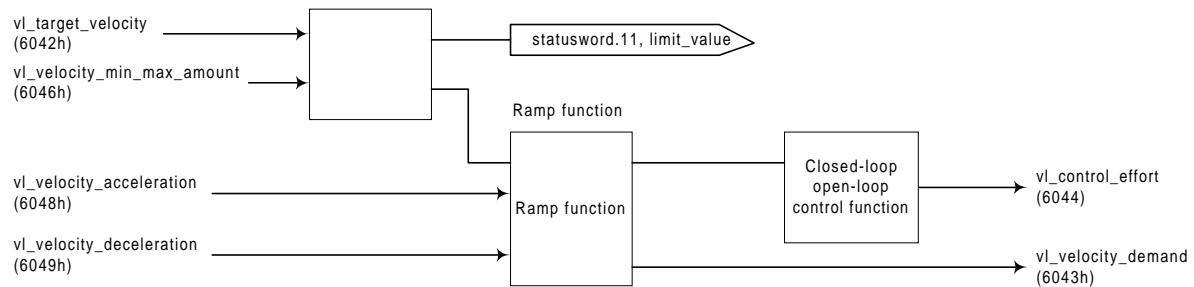


Figure 46: Velocity Mode with mandatory Objects only

18.1.4 Subfunction Description

The Velocity Mode is composed of the following subfunctions:

- Reference Calculation
- Factor Function, Reverse Factor Function
- Percentage Function, Reverse Percentage Function
- Pole Number Function, Reverse Pole Number Function
- Velocity Limit Function
- Velocity Motor Limit Function
- Ramp Function
- Ramp Min Function
- Closed Open Loop Control Function

These subfunctions are only used for the Velocity Mode.

18.2 Object Dictionary Entries

18.2.1 Objects defined in this Chapter

Index	Object	Name	Type	Attr.	M/O
6042 _h	VAR	vl_target_velocity	Integer16	rw	M
6043 _h	VAR	vl_velocity_demand	Integer16	ro	M
6044 _h	VAR	vl_control_effort	Integer16	ro	M
6045 _h	VAR	vl_manipulated_velocity	Integer16	ro	O
6046 _h	ARRAY	vl_velocity_min_max_amount	Unsigned32	rw	M
6047 _h	ARRAY	vl_velocity_min_max	Unsigned32	rw	O
6048 _h	RECORD	vl_velocity_acceleration	Ramp	rw	M
6049 _h	RECORD	vl_velocity_deceleration	Ramp	rw	M
604A _h	RECORD	vl_velocity_quick_stop	Ramp	rw	O
604B _h	ARRAY	vl_setpoint_factor	Integer16	rw	O
604C _h	ARRAY	vl_dimension_factor	Integer32	rw	O
604D _h	VAR	vl_pole_number	Unsigned8	rw	O
604E _h	VAR	vl_velocity_reference	Unsigned32	rw	O
604F _h	VAR	vl_ramp_function_time	Unsigned32	rw	O
6050 _h	VAR	vl_slow_down_time	Unsigned32	rw	O
6051 _h	VAR	vl_quick_stop_time	Unsigned32	rw	O
6052 _h	VAR	vl_nominal_percentage	Integer16	rw	O
6053 _h	VAR	vl_percentage_demand	Integer16	ro	O
6054 _h	VAR	vl_actual_percentage	Integer16	ro	O

18.2.2 Objects defined in other Chapters

Index	Object	Name	Type	Chapter
603F _h	VAR	error_code	Integer16	ce
6040 _h	VAR	controlword	Unsigned16	dc
6041 _h	VAR	statusword	Integer16	dc
605A _h	VAR	quick_stop_option_code	Integer16	dc
605B _h	VAR	shut_down_option_code	Integer16	dc
605C _h	VAR	disable_operation_option_code	Integer16	dc
605D _h	VAR	stop_option_code	Integer16	dc
6060 _h	VAR	modes_of_operation	Integer8	dc
6061 _h	VAR	modes_of_operation_display	Integer8	dc

18.3 Object Description

18.3.1 Object 6042_h: vl_target_velocity

The *vl_target_velocity* is the required velocity of the system. It is multiplied by the *vl_dimension_factor* and the *vl_setpoint_factor*, if these are implemented. The *vl_target_velocity* is converted to the unit [rpm] by multiplying the *vl_target_velocity* by the *vl_dimension_factor*. The unit of the *vl_target_velocity* is interpreted as [rpm] if the *vl_dimension_factor* is not implemented or has the value 1.

Index	6042 _h
Name	vl_target_velocity
Object Code	VAR
Data Type	Integer16

Value description

Object Class	M: vl	O:
Access	rw	
PDO Mapping	Possible	
Units	Speed Units G1a	
Value Range	-32768..32767	
Mandatory Range	-32768..0..+32767	
Default Value	0	
Substitute Value	-	

18.3.2 Object 6043_h: vl_velocity_demand

The *vl_velocity_demand* is the instantaneous velocity provided by the Ramp_function, scaled to the unit of the *vl_target_velocity*. The value ranges from -32768 to 32767 (Integer16). The parameter could only be read, because it is changed only by the drive.

Index	6043 _h
Name	vl_velocity_demand
Object Code	VAR
Data Type	Integer16

Value description

Object Class	M: vl	O:
Access	ro	
PDO Mapping	Possible	
Units	Speed Units G1a	
Value Range	-32768..32767	
Mandatory Range	-32768..0..+32767	
Default Value	- (drive output variable)	
Substitute Value	-	

18.3.3 Object 6053_h: vl_percentage_demand

The *vl_percentage_demand* is calculated on the basis of the *vl_velocity_demand* by using the Percentage Function. It is the velocity provided by the Ramp_function in percent. Accordingly, the *vl_percentage_demand* is within the same value range as the *vl_nominal_percentage*. The value ranges from -32768 to 32767 (Integer16). The value 16383 corresponds to 100% of the *vl_velocity_reference*. Accordingly, an indication range of +/- 200% is possible. The parameter is read-only.

Index	6053_h
Name	vl_percentage_demand
Object Code	VAR
Data Type	Integer16

Value description

Object Class	M:	O: vl
Access	ro	
PDO Mapping	Possible	
Units	(100/16383) %	
Value Range	-32768..32767	
Mandatory Range	-32768..0..+32767	
Default Value	- (drive output variable)	
Substitute Value	-	

18.3.4 Object 6054_h: vl_actual_percentage

The *vl_actual_percentage* is calculated on the basis of the *vl_control_effort* by using the Percentage Function. In this way, the *vl_actual_percentage* has the same value range as the *vl_nominal_percentage*. The value ranges from -32768 to 32767 (Integer16). The value 16383 corresponds to 100% of the *vl_velocity_reference*. Therefore, an indication range of +/- 200% is possible. The parameter is read-only.

Index	6054_h
Name	vl_actual_percentage
Object Code	VAR
Data Type	Integer16

Value description

Object Class	M:	O: vl
Access	ro	
PDO Mapping	Possible	
Units	(100/16383) %	
Value Range	-32768..32767	
Mandatory Range	-32768..0..+32767	
Default Value	- (drive output variable)	
Substitute Value	-	

18.3.5 Object 6055_h: vl_manipulated_percentage

The *vl_manipulated_percentage* is calculated on the basis of the *vl_manipulated_velocity*. In this way, the *vl_manipulated_percentage* is shown in the same value range as the *vl_nominal_percentage*. The value ranges from -32768 to 32767 (Integer16). The value 16383 corresponds to 100% of the *vl_velocity_reference*. Therefore, an indication range of +/- 200% is possible. The parameter is read-only.

Index	6055_h
Name	vl_manipulated_percentage
Object Code	VAR
Data Type	Integer16

Value description

Object Class	M:	O: vl
Access	ro	
PDO Mapping	Possible	
Units	(100/16383) %	
Value Range	-32768..32767	
Mandatory Range	-32768..0..+32767	
Default Value	- (drive output variable)	
Substitute Value	-	

18.3.6 Object 604E_h: vl_velocity_reference

This parameter serves to represent *velocity values* (setpoints, actual values and ramps) as relative values. If the *vl_velocity_reference* is modified, the ramps slopes, if objects *vl_ramp_function_time*, *vl_slow_down_time* or *vl_quick_stop_time* are implemented, are changed relative to the change in the *vl_velocity_reference*.

This parameter has the same unit as the *vl_target_velocity* and the following value range: 0....4 294 967 295 (Unsigned32).

Index	604E_h
Name	vl_velocity_reference
Object Code	VAR
Data Type	Unsigned32

Value description

Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	Speed Units G1a	
Value Range	0..(2 ³² -1)	
Mandatory Range	0 ...+ 4294967295	
Default Value	- (drive output variable)	
Substitute Value	-	

Converting percentages to velocity values

$$\text{Velocity value} = \frac{\text{Percentage} * \text{vl_velocity_reference}}{3FFF_h}$$

Converting velocity values to percentages

$$\text{Percentage} = \frac{\text{Velocity value} * 3FFF_h}{\text{vl_velocity_reference}}$$

18.3.7 Object 604C_n: vl_dimension_factor

The *vl_dimension_factor* is generated by division using a numerator subparameter and a denominator subparameter. These parameter have a value ranging from -2 147 483 648 to 2 147 483 647 (Integer32), but except the value 0!

The *vl_dimension_factor* serves to include gearing in calculation or serves to scale the frequencies or specific units of the user. It influences the specified setpoint, the velocity limit and the Ramp_function as well as the output variables of the Speed Function .

Index	604C _n
Name	vl_dimension_factor
Object Code	ARRAY
Number of Elements	2
Data Type	Integer32

Value description

Sub-Index	01 _h	
Description	vl_dimension_factor_numerator	
Object Class	M: -	O: vl
Access	rw	
PDO Mapping	Possible	
Units	-(no units)	
Value Range	$(-2^{31})..(2^{31}-1)$	
Mandatory Range	1	
Default Value	manufacturer defined	
Substitute Value	1	

Sub-Index	02 _h	
Description	vl_dimension_factor_denominator	
Object Class	M: -	O: vl
Access	rw	
PDO Mapping	Possible	
Units	-(no units)	
Value Range	$(-2^{31})..(2^{31}-1)$	
Mandatory Range	1	
Default Value	manufacturer defined	
Substitute Value	1	

Calculating the vl_dimension_factor

Every user's specific speed consists of a specific unit referred to a specific unit of time (e.g. 1/sec, bottles/min, m/sec,...).

The purpose of the dimension factor is to convert this specific unit to the revolutions/minute unit.

Specific unit * DF = 1 revolution (motor shaft)

$$I \cdot DF = O$$

I = vl_target_velocity expressed as the user's specific speed

Input value of the Factor Function

Unit of I: [I] = Specific unit

O = Speed value in [rpm]

Output value of the Factor Function

$$\text{Unit of O: } [O] = \frac{1}{\text{min}} = \frac{\text{Revolution}}{\text{min}}$$

DF = Dimension factor

$$\text{Unit of DF: } [DF] = \frac{1}{\text{Specific unit}} * \frac{1}{\text{min}}$$

Refer to the application note for an examples.

18.3.8 Object 604B_n: vl_setpoint_factor

The *vl_setpoint_factor* is generated by division, using a numerator subparameter and a denominator subparameter. These subparameter have no unit and have values within a range from -32768 to 32767 (Integer16), but excluding the value 0!

The *vl_setpoint_factor* serves to modify the resolution or directing range of the specified setpoint. It is included in calculation of the specified setpoint and the output variables of the Speed Function only.

Index	604B _n
Name	vl_setpoint_factor
Object Code	ARRAY
Number of Elements	2
Data Type	Integer16

Value description

Sub-Index	01 _n	
Description	vl_setpoint_factor_numerator	
Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	-(no units)	
Value Range	-32768..32767	
Mandatory Range	1	
Default Value	manufacturer defined	
Substitute Value	1	

Sub-Index	02 _h	
Description	vl_setpoint_factor_denominator	
Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	-(no units)	
Value Range	-32768..32767	
Mandatory Range	1	
Default Value	manufacturer defined	
Substitute Value	1	

18.3.9 Object 604D_h: vl_pole_number

The user must describe the *vl_pole_number* parameter with a value corresponding to the number of poles belonging to the connected motor. This parameter has no unit. The value range depends on the manufacturer-specific need and is represented as unsigned 8. If the Object *vl_pole_number* does fit for the desired type of motor, this object could left out or set to value 2.

Index	604D_h
Name	vl_pole_number
Object Code	VAR
Data Type	Unsigned8

Value description

Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	- (number of poles)	
Value Range	0..255	
Mandatory Range	2	
Default Value	manufacturer defined	
Substitute Value	2	

If the number of pole pairs is known, the number of poles is: $vl_pole_number = 2 * \text{pole pairs}$

Converting velocity values to frequency values

$$\text{Frequency} = \frac{\text{Velocity} * vl_pole_number}{60 * 2}$$

Converting frequency values to velocity values

$$\text{Velocity} = \frac{\text{Frequency} * 60 * 2}{vl_pole_number}$$

18.3.10 Object 6046_h: vl_velocity_min_max_amount

The *vl_velocity_min_max_amount* parameter is composed of the *vl_velocity_min_amount* and *vl_velocity_max_amount* subparameter. These subparameters don't have units and have values within a range from 0 to 4 294 967 295 (unsigned 32):

The *vl_velocity_max_amount* subparameter is mapped internally to the *vl_velocity_max_pos* and *vl_velocity_max_neg* values. The *vl_velocity_min_amount* subparameter is mapped internally to the *vl_velocity_min_pos* and *vl_velocity_min_neg* values.

Only the positive values are returned if the *vl_velocity_min_max_amount* parameter is read out.

Index	6046_h
Name	vl_velocity_min_max_amount
Object Code	ARRAY
Number of Elements	2
Data Type	Unsigned32

Value description

Sub-Index	01 _h	
Description	vl_velocity_min_amount	
Object Class	M: vl	O:
Access	rw	
PDO Mapping	Possible	
Units	G1b	
Value Range	0..(2 ³² -1)	
Mandatory Range	0... manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

Sub-Index	02 _h	
Description	vl_velocity_max_amount	
Object Class	M: vl	O:
Access	rw	
PDO Mapping	Possible	
Units	G1b	
Value Range	0..($2^{32}-1$)	
Mandatory Range	0... manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

This transfer characteristic results from *vl_velocity_min_max_amount*

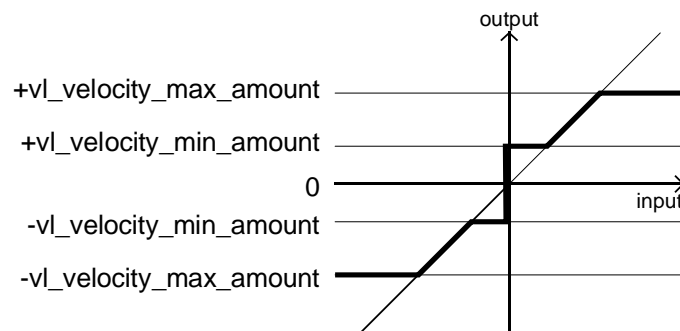


Figure 45: vl_velocity_min_max_amount transfer characteristic

18.3.11 Object 6047_h: vl_velocity_min_max

The *vl_velocity_min_max* parameter is composed of the *vl_velocity_min_pos*, *vl_velocity_max_pos*, *vl_velocity_min_neg* and *vl_velocity_max_neg* subparameter. These subparameter have no units and have values within a range from 0 to 4 294 967 295 (unsigned 32).

The subparameter are mapped internally to the corresponding values.

Index	6047_h
Name	vl_velocity_min_max
Object Code	ARRAY
Number of Elements	4
Data Type	Unsigned32

Value description

Sub-Index	01 _h	
Description	vl_velocity_min_pos	
Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	G1b	
Value Range	0..($2^{32}-1$)	
Mandatory Range	manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

Sub-Index	02 _h	
Description	vl_velocity_max_pos	
Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	G1b	
Value Range	0..($2^{32}-1$)	
Mandatory Range	manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

Sub-Index	03 _h	
Description	vl_velocity_min_neg	
Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	G1b	
Value Range	0..($2^{32}-1$)	
Mandatory Range	manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

Sub-Index	04 _h	
Description	vl_velocity_max_neg	
Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	G1b	
Value Range	0..($2^{32}-1$)	
Mandatory Range	manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

This transfer characteristic results from *vl_velocity_min_max*

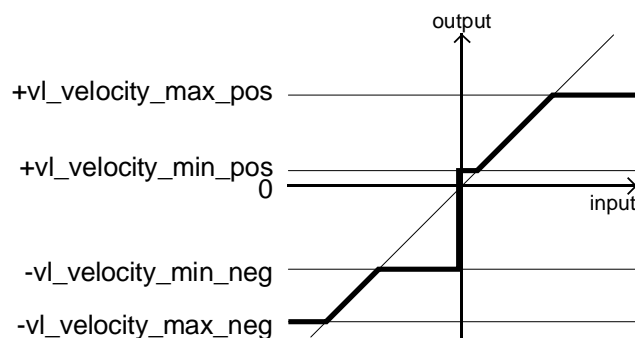


Figure 46: vl_velocity_min_max transfer characteristic

18.3.12 Object 6058_h: vl_frequency_motor_min_max_amount

The frequency parameter of the *vl_frequency_motor_min_max_amount* objects are mapped internally to the parameter of the corresponding speed objects.

Index	6058 _h
Name	vl_frequency_motor_min_max_amount
Object Code	ARRAY
Number of Elements	2
Data Type	Unsigned32

Value description

Sub-Index	01 _h	
Description	vl_frequency_motor_min_amount	
Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..($2^{32}-1$)	
Mandatory Range	0... manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

Sub-Index	02 _h	
Description	vl_frequency_motor_max_amount	
Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..($2^{32}-1$)	
Mandatory Range	0... manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

This transfer characteristic results from vl_frequency_motor_min_max_amount

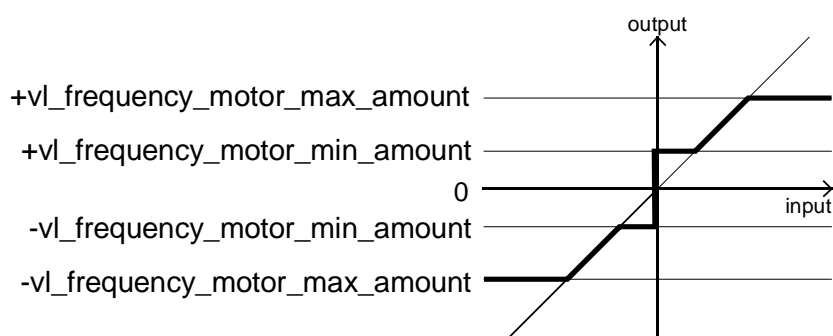


Figure 47: vl_frequency_motor_min_max_amount transfer characteristic

18.3.13 Object 6059_h: vl_frequency_motor_min_max

The frequency parameter of the *vl_frequency_motor_min_max* objects are mapped internally to the parameter of the corresponding speed objects.

Index	6059_h
Name	vl_frequency_motor_min_max
Object Code	ARRAY
Number of Elements	4
Data Type	Unsigned32

Value description

Sub-Index	01 _h	
Description	vl_frequency_motor_min_pos	
Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..(2 ³² -1)	
Mandatory Range	manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

Sub-Index	02 _h	
Description	vl_frequency_motor_max_pos	
Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..(2 ³² -1)	
Mandatory Range	manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

Sub-Index	03 _h	
Description	vl_frequency_motor_min_neg	
Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	- (1/1000 rpm)	
Value Range	0..($2^{32}-1$)	
Mandatory Range	manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

Sub-Index	04 _h	
Description	vl_frequency_motor_max_neg	
Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	-	
Value Range	0..($2^{32}-1$)	
Mandatory Range	manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

This transfer characteristic results from vl_velocity_min_max

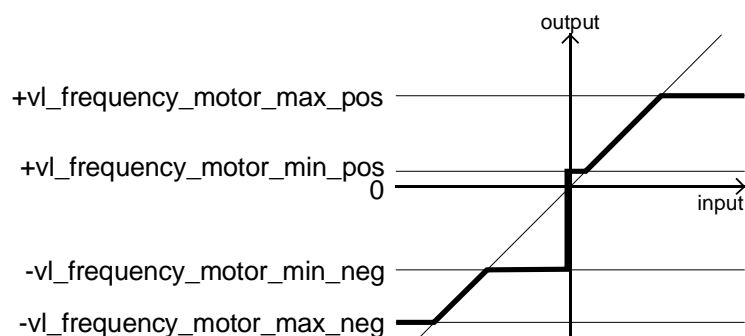


Figure 48: vl_velocity_min_max transfer characteristic

18.3.14 Object 6056_h: vl_velocity_motor_min_max_amount

The *vl_velocity_motor_min_max_amount* parameter is composed of the *vl_velocity_motor_min_amount* and *vl_velocity_motor_max_amount* parameter. These subparameters have the unit [(1/1000) rpm] and values within a range from 0 to 4294967295 [(1/1000) rpm] (Unsigned32). This results in a limiting range from 0..4294967295[rpm].

The *vl_velocity_motor_max_amount* subparameter is mapped internally to the *vl_velocity_motor_max_pos* and *vl_velocity_motor_max_neg* values. The *vl_velocity_motor_min_amount* subparameter is mapped internally to the *vl_velocity_motor_min_pos* and *vl_velocity_motor_min_neg* values.

Only the positive values are returned if the *vl_velocity_motor_min_max_amount* parameter is read.

Index	6056 _h
Name	vl_velocity_motor_min_max_amount
Object Code	ARRAY
Number of Elements	2
Data Type	Unsigned32

Value description

Sub-Index	01 _h	
Description	vl_velocity_motor_min_amount	
Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	(1/1000 rpm)	
Value Range	0..(2 ³² -1)	
Mandatory Range	0... manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

Sub-Index	02 _h	
Description	vl_velocity_motor_max_amount	
Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	(1/1000 rpm)	
Value Range	0..(2 ³² -1)	
Mandatory Range	0... manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

This transfer characteristic results from *vl_velocity_motor_min_max_amount*

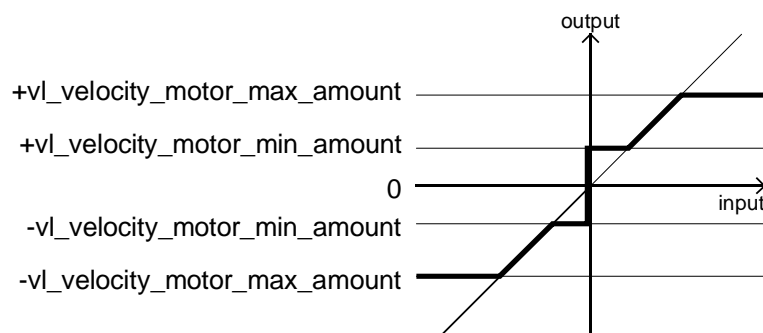


Figure 49: vl_velocity_motor_min_max_amount transfer characteristic

18.3.15 Object 6057_h: vl_velocity_motor_min_max

The *vl_velocity_motor_min_max* parameter is composed of the *vl_velocity_motor_min_pos*, *vl_velocity_motor_max_pos*, *vl_velocity_motor_min_neg* and *vl_velocity_motor_max_neg* subparameter. These subparameters have the unit [1/(1000 min)] and values within a range from 0 ... 4 294 967 295 [1/(1000 min)] (Unsigned32). This results in a limiting range from 0..4 294 967 [rpm].

The subparameter are mapped internally to the corresponding values.

Index	6057_h
Name	vl_velocity_motor_min_max
Object Code	ARRAY
Number of Elements	4
Data Type	Unsigned32

Value description

Sub-Index	01 _h	
Description	vl_velocity_motor_min_pos	
Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	- (1/1000 rpm)	
Value Range	0..($2^{32}-1$)	
Mandatory Range	manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

Sub-Index	02 _h	
Description	vl_velocity_motor_max_pos	
Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	- (1/1000 rpm)	
Value Range	0..($2^{32}-1$)	
Mandatory Range	manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

Sub-Index	03 _h	
Description	vl_velocity_motor_min_neg	
Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	- (1/1000 rpm)	
Value Range	0..($2^{32}-1$)	
Mandatory Range	manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

Sub-Index	04 _h	
Description	vl_velocity_motor_max_neg	
Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	- (1/1000 rpm)	
Value Range	0..($2^{32}-1$)	
Mandatory Range	manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

This transfer characteristic results from vl_velocity_min_max

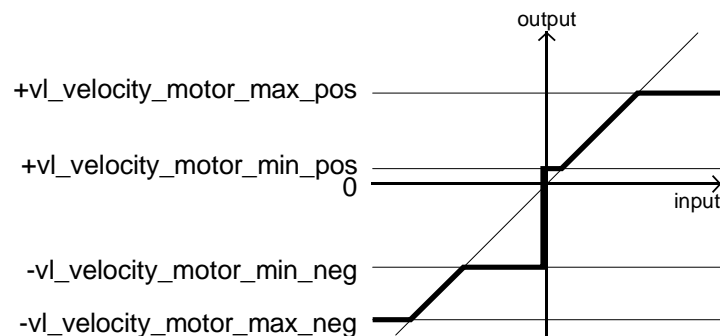


Figure 50: *vl_velocity_min_max* transfer characteristic

Limit-Value

The Limit-Value message is generated if the input value of the *Velocity_motor_limit_function* results in a value outside of the operating range of the *Velocity_motor_limit_function*. The Limit-Value message is mapped as one bit in the status word.

18.3.16 Object 6048_h: vl_velocity_acceleration

The *vl_velocity_acceleration* parameter specifies the slope of the acceleration ramp. It is generated as the quotient of the delta_speed and delta_time subparameter.

Index	6048 _h
Name	vl_velocity_acceleration
Object Code	RECORD
Number of Elements	2

Value description

Sub-Index	01 _h	
Description	delta_speed	
Object Class	M: vl	O:
Access	rw	
PDO Mapping	Possible	
Units	G1b	
Value Range	0..(2 ³² -1)	
Mandatory Range	0..manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	
Data Type	Unsigned32	

Sub-Index	02 _h	
Description	delta_time	
Object Class	M: vl	O:
Access	rw	
PDO Mapping	Possible	
Units	Seconds (s)	
Value Range	0..65535	
Mandatory Range	0..manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	
Data Type	Unsigned16	

delta_speed

The delta_speed has the same unit as the *vl_target_velocity*. This subparameter has the following value range: 0 ... 4294967295 (unsigned 32).

delta_time

This subparameter is specified in sec and has the following value range: 0 ... 65 535 [sec] (unsigned 16).

This function directly follows the setpoint if the parameter 0 is defined for the delta_time value.

$$vl_velocity_acceleration = \frac{delta_speed}{delta_time} = a_B$$

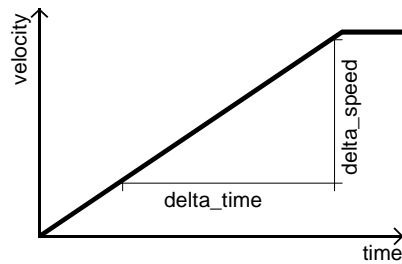


Figure 51: vl_velocity_acceleration transfer characteristic

18.3.17 Object 6049_h: vl_velocity_deceleration

The *vl_velocity_deceleration* parameter specifies the slope of the deceleration ramp. It is generated as the quotient of the delta_speed and delta_time subparameter.

index	6049_h
name	vl_velocity_deceleration
object code	RECORD
number of elements	2

Value description

sub-index	01 _h	
description	delta_speed	
object class	M: vl	O:
access	rw	
PDO mapping	Possible	
units	G1b	
value range	0...+4294967295	
mandatory range	0... manufacturer defined	
default value	manufacturer defined	
substitute value	-	
Data Type	Unsigned32	

sub-index	02 _h	
description	delta_time	
object class	M: vl	O:
access	rw	
PDO mapping	Possible	
units	Seconds (s)	
value range	0 ...+ 65535	
mandatory range	0... manufacturer defined	
default value	manufacturer defined	
substitute value	-	
Data Type	Unsigned16	

delta_speed

The delta_speed has the same unit as the *vl_target_velocity*. This subparameter has the following value range:

$$0..(2^{32}-1) \text{ (Unsigned32).}$$

delta_time

This subparameter is specified in sec and has the following value range:

$$0..65535 \text{ [sec] (Unsigned16).}$$

This function directly follows the setpoint if the value 0 is defined for the delta_time parameter.

$$vl_velocity_deceleration = \frac{delta_speed}{delta_time} = a_v$$

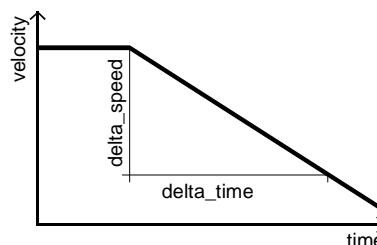


Figure 52: vl_velocity_deceleration transfer characteristic

18.3.18 Object 604A_h: vl_velocity_quick_stop

The *vl_velocity_quick_stop* parameter specifies the slope of the quick stop ramp. It is generated as the quotient of the *delta_speed* and *delta_time* subparameter.

Index	604A_h
Name	vl_velocity_quick_stop
Object Code	RECORD
Number of Elements	2

Value description

Sub-Index	01 _h	
Description	delta_speed	
Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	G1b	
Value Range	0...+4294967295	
Mandatory Range	0... manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	
Data Type	Unsigned32	

Sub-Index	02 _h	
Description	delta_time	
Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	Seconds (s)	
Value Range	0..65535	
Mandatory Range	0..manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	
Data Type	Unsigned16	

delta_speed

The velocity has the same unit as the *vl_target_velocity*. This subparameter has the following value range:

$$0..(2^{32}-1) \text{ (unsigned 32).}$$

delta_time

This subparameter is specified in sec and has the following value range:

0..65535 [sec] (unsigned 16).

This function directly follows the setpoint if the parameter 0 is defined for the delta_time value.

$$velocity_quick_stop = \frac{delta_speed}{delta_time} = a_s$$

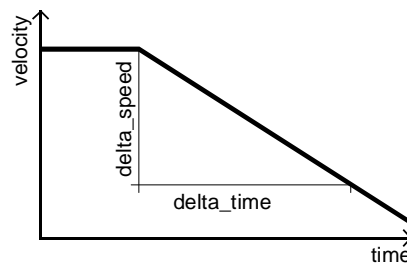


Figure 53: vl_velocity_quick_stop transfer characteristic

18.3.19 Object 604F_h: vl_ramp_function_time

The *vl_ramp_function_time* specifies the time during which the drive starts up from zero to the *vl_velocity_reference*.

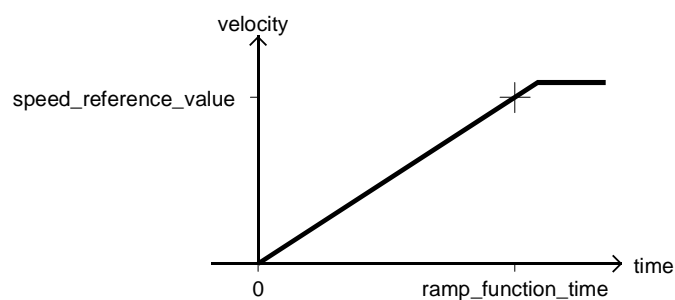


Figure 54: vl_ramp_function_time transfer characteristic

This parameter is specified in ms and has the following value range:

0..(2³²-1) [msec] (unsigned 32).

By setting the parameter 0 for the *vl_ramp_function_time*, the ramp becomes infinite and the reference variable directly follows the setpoint.

Index	604F_h
Name	vl_ramp_function_time
Object Code	VAR
Data Type	Unsigned32

Value description

Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	Milliseconds (ms)	
Value Range	0..($2^{32}-1$)	
Mandatory Range	manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

18.3.20 Object 6050_h: vl_slow_down_time

The *vl_slow_down_time* specifies the time during which the drive slows down from *vl_velocity_reference* to zero.

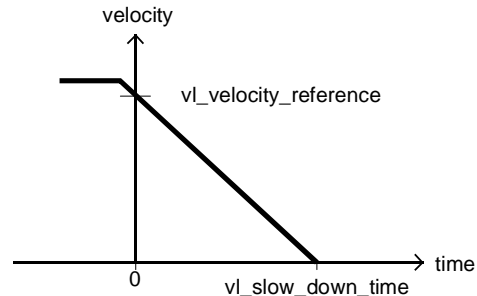


Figure 55: vl_slow_down_time transfer characteristic

This parameter is specified in ms and has the following value range:

0..($2^{32}-1$) [msec] (unsigned 32).

By defining the parameter 0 for the *vl_slow_down_time*, the ramp becomes infinite and the reference variable directly follows the setpoint.

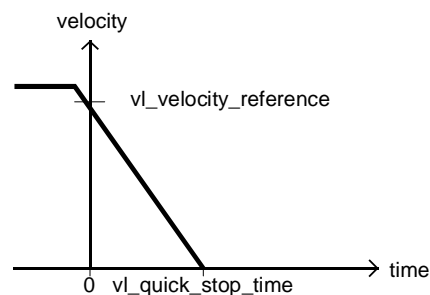
Index	6050_h
Name	vl_slow_down_time
Object Code	VAR
Data Type	Unsigned32

Value description

Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	Milliseconds (ms)	
Value Range	$0..(2^{32}-1)$	
Mandatory Range	manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

18.3.21 Object 6051_h: vl_quick_stop_time

The *vl_quick_stop_time* specifies the time during which the drive slows down from *vl_velocity_reference* to zero in the QUICK STOP ACTIVE state.

**Figure 56: vl_quick_stop_time transfer characteristic**

This parameter is specified in ms and has the following value range:

$0..(2^{32}-1)$ [msec] (unsigned 32).

By defining the parameter 0 for the *vl_quick_stop_time*, the ramp becomes infinite and the reference variable directly follows the setpoint.

Index	6051_h
Name	vl_quick_stop_time
Object Code	VAR
Data Type	Unsigned32

Value description

Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	Milliseconds (ms)	
Value Range	0..($2^{32}-1$)	
Mandatory Range	manufacturer defined	
Default Value	manufacturer defined	
Substitute Value	-	

18.3.22 Object 6044_h: vl_control_effort

The *vl_control_effort* is the velocity at the motor spindle or load, scaled to the unit of the *vl_target_velocity*. Depending on the system, velocity deviations may occur between the *vl_control_effort* and the physical velocity. For simple drives without closed loop control or observer this value reads the object *vl_velocity_demand*. The value ranges from -32768 to 32767 (Integer16).

Index	6044_h
Name	vl_control_effort
Object Code	VAR
Data Type	Integer16

Value description

Object Class	M: vl	O:
Access	ro	
PDO Mapping	Possible	
Units	Speed Units G1a	
Value Range	-32768..32767	
Mandatory Range	-32768..0..+32767	
Default Value	- (drive output variable)	
Substitute Value	-	

18.3.23 Object 6045_h: vl_manipulated_velocity

The *vl_manipulated_velocity* is the velocity of the motor spindle or load with a compensation value, scaled to the unit of the *vl_target_velocity*. The compensation value is generated by the controller/control function. The value ranges from -32768 to 32767 (Integer16). The parameter is read-only.

Index	6045_h
Name	vl_manipulated_velocity
Object Code	VAR
Data Type	Integer16

Value description

Object Class	M:	O: vl
Access	ro	
PDO Mapping	Possible	
Units	Speed Units G1a	
Value Range	-32768..32767	
Mandatory Range	-32768..0..+32767	
Default Value	- (drive output variable)	
Substitute Value	-	

18.3.24 Object 6052_h: vl_nominal_percentage

The *vl_nominal_percentage* is converted by the percent function to a velocity value. The *vl_nominal_percentage* has no unit (better (100 / 16383) %). Its value ranges from -32768 to 32767 (Integer16). The value 16383 corresponds to 100% of the *vl_velocity_reference*. Accordingly, a total range of the manipulated variable amounting to +/- 200% is possible. The parameter could be read and written.

Index	6052_h
Name	vl_nominal_percentage
Object Code	VAR
Data Type	Integer16

Value description

Object Class	M:	O: vl
Access	rw	
PDO Mapping	Possible	
Units	(100/16383) %	
Value Range	-32768..32767	
Mandatory Range	-32768..0..+32767	
Default Value	0	
Substitute Value	0	

18.4 Functional Description

18.4.1 Percentage Function

The Percentage Function serves to convert percentages to velocity values and vice versa.

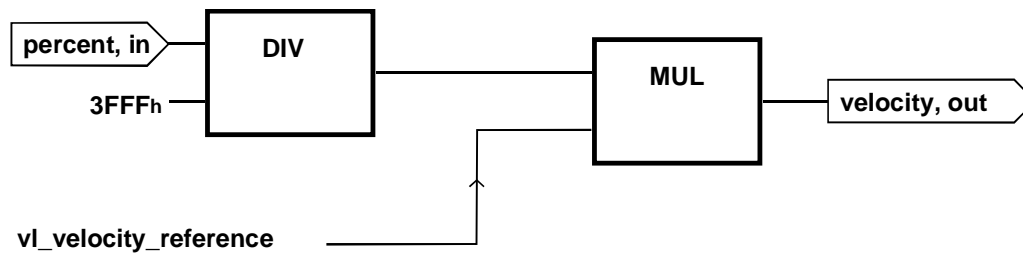


Figure 57: Percentage Function

and vice versa

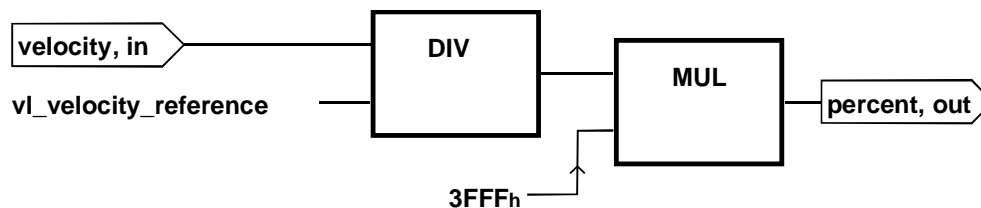


Figure 58: Reverse Percentage Function

18.4.2 Factor Function and Reverse Factor Function

The Factor Function multiplies the input variables by the assigned factors.

- The *vl_target_velocity* is multiplied by the *vl_dimension_factor* and the *vl_setpoint_factor*.
- The values of the velocity limit and the values for the Ramp Function are only multiplied by the *vl_dimension_factor*.

A factor has a value of 1 if it is not implemented.

The Factor Function for two factors is built of two function in series connection.

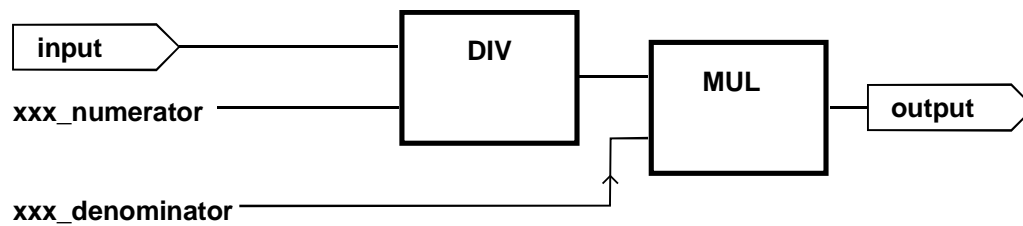


Figure 59: Factor Function

The Reverse Factor Function divides the input variables by the assigned factors.

- The output variables of the Velocity Mode are calculated by division with the *vl_dimension_factor* and the *vl_setpoint_factor* and therefore returned to the scaling of the specified setpoint.

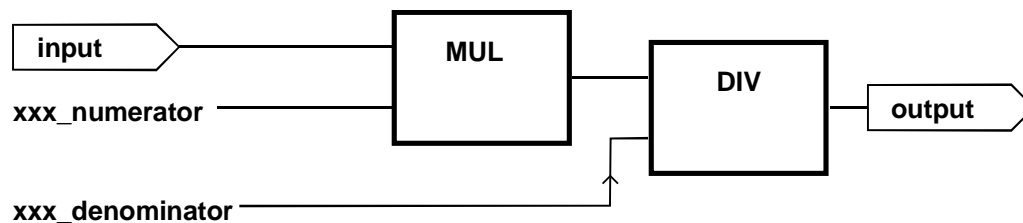


Figure 60: Reverse Factor Function

18.4.3 Pole Number Function

The Pole Number Function serves to convert frequency values to velocity values

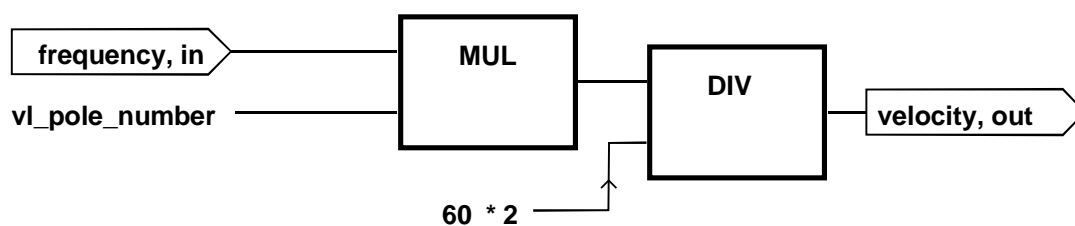


Figure 61: Pole Number Function

and vice versa

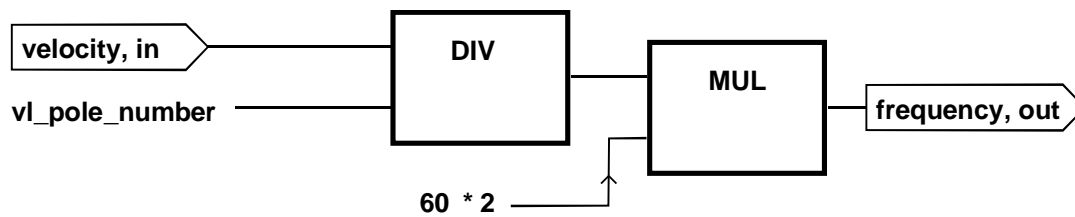


Figure 62: Reverse Pole Number Function

18.4.4 Velocity Limit Function

The velocity limit defines the valid velocity range for the drive. Limits could be specified in the user specific units by including the *vl_dimension_factor* in the speed limit.

Limit-Value

The Limit-Value message is generated if the input value of the speed limit results in a value outside the speed limit's operating range. The Limit-Value message is mapped as one bit in the *statusword*.

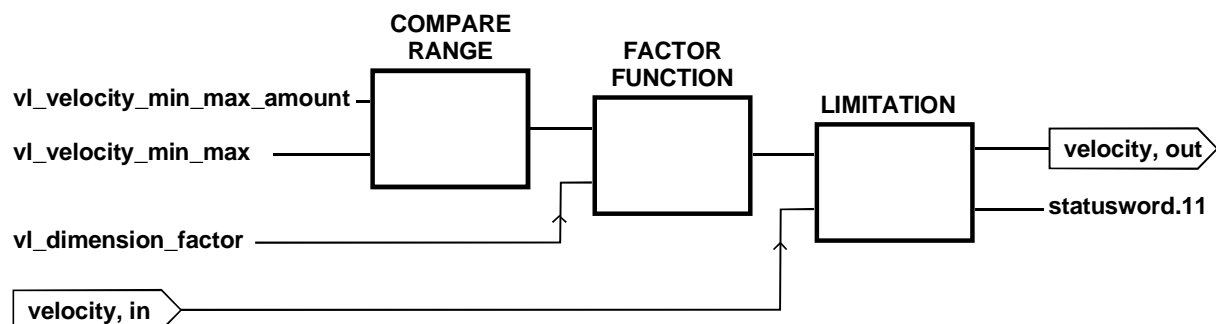


Figure 63: Velocity Limit Function

18.4.5 Velocity Motor Limit Function

The Velocity Motor Limit Function limits the motor velocity range. This parameter has a safety function that ensures that the range of the set value of motor velocity cannot be exceeded inadvertently by a modification of a factor.

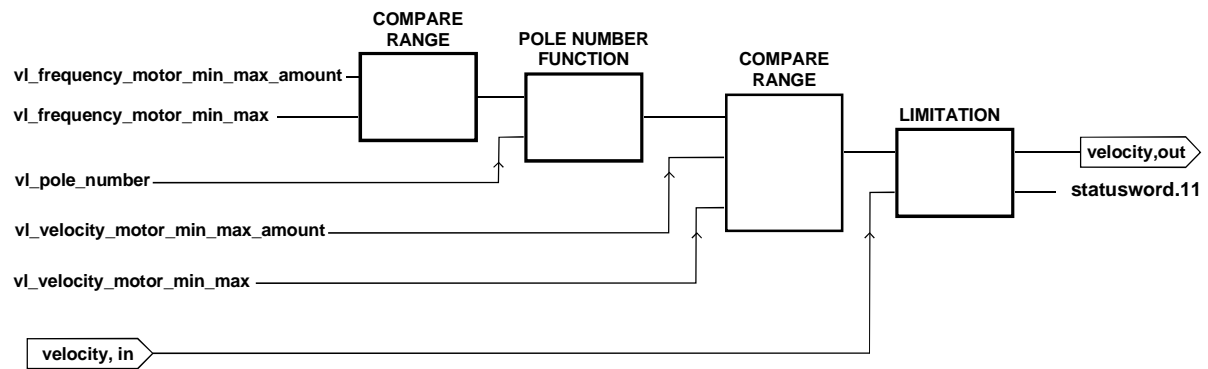


Figure 64: Velocity Motor Limit Function

18.4.6 Ramp Function

The Ramp Function is used to limit the increase and decrease of velocity. The velocity output is equal to the input as long as the changes are below $a_{B_{min}}$, $a_{V_{min}}$ or $a_{S_{min}}$.

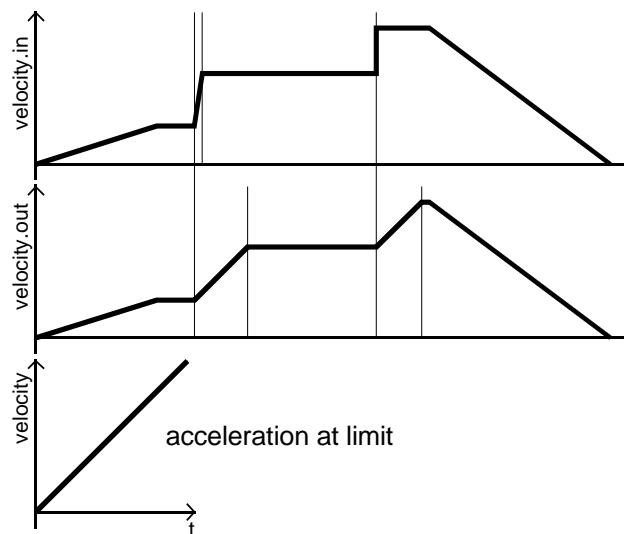


Figure 65: Velocity Profile

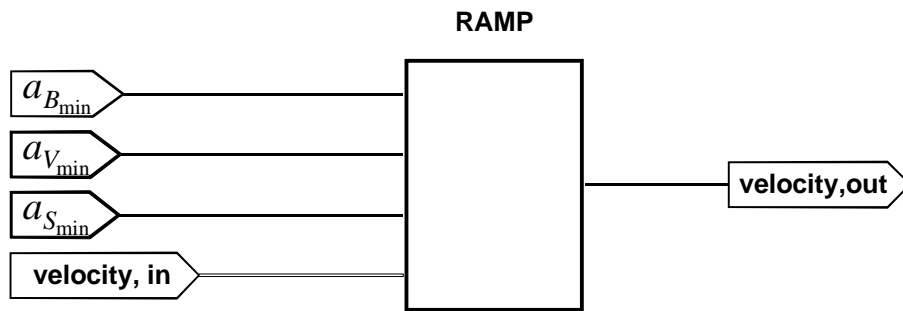


Figure 66: Ramp Function

The internal ramp values $a_{B_{min}}$ and $a_{V_{min}}$ directly consist of the *vl_velocity_acceleration* and *vl_velocity_deceleration* parameter.

The internal ramp values $a_{B_{min}}$, $a_{V_{min}}$ and $a_{S_{min}}$ are the output values of the *Ramp_min_*-function, weighted with the *vl_dimension_factor*.

18.4.7 Ramp Min Function

The Ramp Min Function selects the minimal change of velocity.

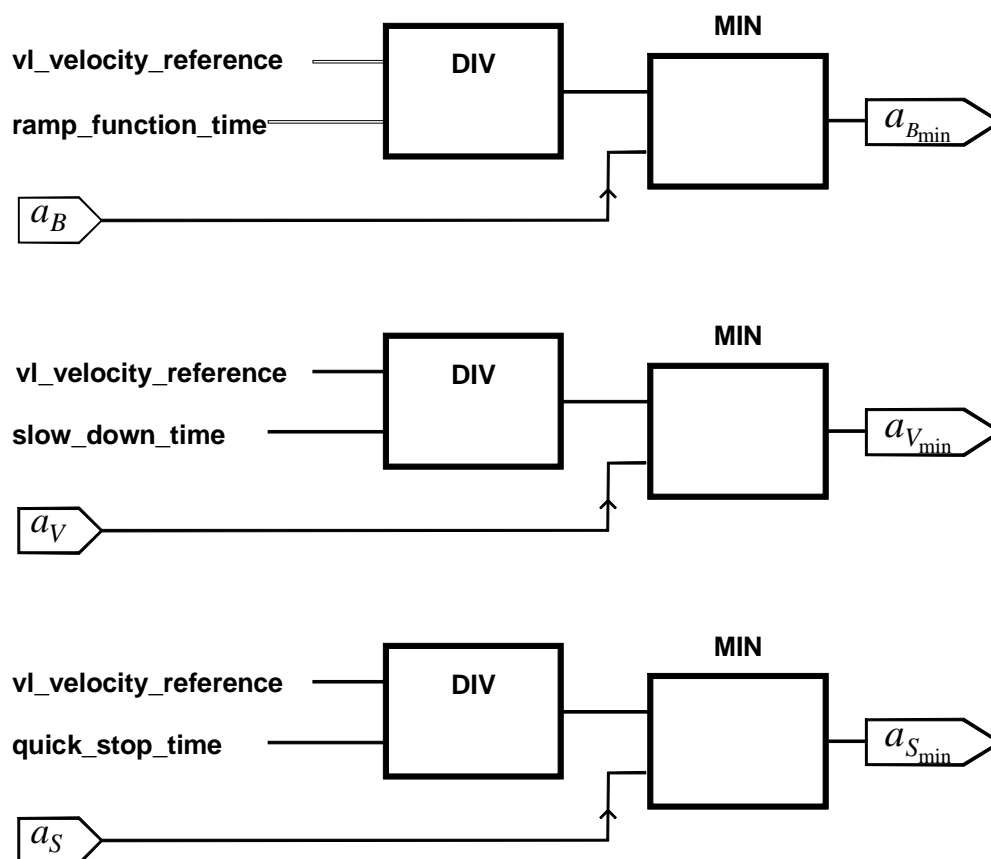


Figure 67: Ramp Min Function

The internal ramp values ($a_{B_{\min}}$, $a_{V_{\min}}$, $a_{S_{\min}}$) are recalculated as follows if one of the input parameter for the Ramp Function is modified.

$$a_{B_{\min}} = \text{MIN} \left(a_B , \frac{vl_velocity_reference}{vl_ramp_function_time} \right)$$

$$a_{V_{\min}} = \text{MIN} \left(a_V , \frac{vl_velocity_reference}{vl_slow_down_time} \right)$$

$$a_{S_{\min}} = \text{MIN} \left(a_S , \frac{vl_velocity_reference}{vl_quick_stop_time} \right)$$

The Ramp Min Function selects the lower respective value of the slopes.

18.4.8 Reference Calculation

This subfunction decides on the setpoint processing. The setpoint value may be given as an percentage and (or) as an absolute value. Therefore two objects are defined in this profile. The first object is the *vl_target_velocity* containing the absolute setpoint value. This is an mandatory object for every drive. Some drives may have implemented the object *vl_nominal_percentage* which consist of an percentage setpoint value. So these manufactures have to specify how this two values are handled within the drive. Most profile implementations will add the percentage and the absolute setpoint value to calculate the internal setpoint. It is also possible to use only one value. Then the last written object is used internally.

18.4.9 Closed Open Loop Control Function

On the basis of the *vl_control_effort*, the controller/control function returns the *vl_control_effort* and the *vl_manipulated_velocity*.

Depending on realisation of the function, the *vl_control_effort* is the *vl_control_effort* or a calculated or measured *vl_control_effort*.

Depending on realisation of the function, the *vl_manipulated_velocity* is the *vl_control_effort* or a calculated speed_output.

19 Appendix

A Object Dictionary by Chapter

A.1 Common Entries in the Object Dictionary

Object Dictionary for Chapter Common Entries in the Object Dictionary															
object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pV	tq	vI	
6007 _h	abort_connection_option_code	VAR	Integer16		rw	Possible	-	o							
6007 _h	abort_connection_option_code	VAR	Integer16		rw	Possible	-	o							
603F _h	error_code	VAR	Unsigned16		ro	Possible	-	o							
6402 _h	motor_type	VAR	Unsigned16		rw	Possible	-								
6403 _h	motor_catalogue_number	VAR	Visible String		rw	Possible	-								
6404 _h	motor_manufacturer	VAR	Visible String		rw	Possible	-								
6405 _h	http_motor_catalog_address	VAR	Visible String		rw	Possible	-								
6406 _h	motor_calibration_date	VAR	Date		rw	Possible	-								
6407 _h	motor_service_period	VAR	Unsigned32		rw	Possible	-								
6410 _h	motor_data	RECORD													
6502 _h	supported_drive_modes	VAR	Unsigned32		ro	Possible	-								
6503 _h	drive_catalogue_number	VAR	Visible String		rw	Possible	-								
6504 _h	drive_manufacturer	VAR	Visible String		rw	Possible	-								
6505 _h	http_drive_catalog_address	VAR	Visible String		rw	Possible	-								
6510 _h	drive_data	RECORD													
60FD _h	digital_inputs	VAR	Unsigned32		ro	Possible	-	o							
60FE _h	digital_outputs	RECORD													
	physical_outputs	RECORD		01	rw	Possible	-	o							
	bitmask	RECORD		02	rw	Possible	-	o							

A.2 Device Control

Object Dictionary for Chapter Device Control															
object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pV	tq	vl	
6040 _h	controlword	VAR	Unsigned16		rw	Possible	-	m							
6041 _h	statusword	VAR	Unsigned16		ro	Possible	-	m							
605B _h	shutdown_option_code	VAR	Integer16		rw	No	-	o							
605C _h	disable_operation_option_code	VAR	Integer16		rw	No	-	o							
605A _h	quick_stop_option_code	VAR	Integer16		rw	No	-	o							
605D _h	stop_option_code	VAR	Integer16		rw	No	-	o							
605E _h	fault_reaction_option_code	VAR	Integer16		rw	No	-	o							
6060 _h	modes_of_operation	VAR	Integer8		wo	Possible	-	m							
6061 _h	modes_of_operation_display	VAR	Integer8		ro	Possible	-	m							

A.3 Factor Group

Object Dictionary for Chapter Factor Group															
object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pv	tq	vl	
6089 _h	position_notation_index	VAR	Integer8		rw	Possible	-		o	o	o	o	o		
608A _h	position_dimension_index	VAR	Unsigned8		rw	Possible	-		o	o	o	o	o		
608B _h	velocity_notation_index	VAR	Integer8		rw	Possible	-		o	o	o	o	o		
608C _h	velocity_dimension_index	VAR	Unsigned8		rw	Possible	-		o	o	o	o	o		
608D _h	acceleration_notation_index	VAR	Integer8		rw	Possible	-		o	o	o	o	o		
608E _h	acceleration_dimension_index	VAR	Unsigned8		rw	Possible	-		o	o	o	o	o		
608F _h	position_encoder_resolution	ARRAY	Unsigned32												
	encoder_increments	ARRAY		01	rw	Possible	inc			o	o	o	o		
	motor_revolutions	ARRAY		02	rw	Possible	inc			o	o	o	o		
6090 _h	velocity_encoder_resolution	ARRAY	Unsigned32												
	encoder_increments_per_second	ARRAY		01	rw	Possible	inc/sec			o	o	o	o		
	motor_revolutions_per_second	ARRAY		02	rw	Possible	inc/sec			o	o	o	o		
6091 _h	gear_ratio	ARRAY	Unsigned32												
	motor_revolutions	ARRAY		01	rw	Possible	-			o	o	o	o		
	shaft_revolutions	ARRAY		02	rw	Possible	-			o	o	o	o		
6092 _h	feed_constant	ARRAY	Unsigned32												
	feed	ARRAY		01	rw	Possible	-			o	o	o	o		
	shaft_revolutions	ARRAY		02	rw	Possible	-			o	o	o	o		
6093 _h	position_factor	ARRAY	Unsigned32												
	numerator	ARRAY		01	rw	Possible	-			o	o				
	feed_constant	ARRAY		02	rw	Possible	-			o	o				
6094 _h	velocity_encoder_factor	ARRAY	Unsigned32												
	numerator	ARRAY		01	rw	Possible	-					o			
	divisor	ARRAY		02	rw	Possible	-					o			
6095 _h	velocity_factor_1	ARRAY	Unsigned32												
	numerator	ARRAY		01	rw	Possible	-			o	o	o	o		
	divisor	ARRAY		02	rw	Possible	-			o	o	o	o		
6096 _h	velocity_factor_2	ARRAY	Unsigned32												
	numerator	ARRAY		01	rw	Possible	-			o	o	o	o		
	divisor	ARRAY		02	rw	Possible	-			o	o	o	o		
6097 _h	acceleration_factor	ARRAY	Unsigned32												
	numerator	ARRAY		01	rw	Possible	-			o	o	o	o		
	divisor	ARRAY		02	rw	Possible	-			o	o	o	o		

Object Dictionary for Chapter Factor Group

object no	object name	object code	type	sub- index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pv	tq	vl	
607E _h	polarity	VAR	Unsigned8		rw	Possible	-			o	o	o	o		

A.4 Profile Position Mode

Object Dictionary for Chapter Profile Position Mode															
object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pv	tq	vl	
607A _h	target_position	VAR	Integer32		rw	Possible	position units			m	m				
607B _h	position_range_limit	ARRAY	Integer32												
	min_position_range_limit	ARRAY		01	rw	possible	position units			o	o				
	max_position_range_limit	ARRAY		02	rw	Possible	position units			o	o				
607D _h	software_position_limit	ARRAY	Integer32												
	min_position_limit	ARRAY		01	rw	Possible	position units			o	o				
	max_position_limit	ARRAY		02	rw	Possible	position units			o	o				
607F _h	max_profile_velocity	VAR	Unsigned32		rw	Possible	speed units			o	o	o	o		
6080 _h	max_motor_speed	VAR	Unsigned16		rw	Possible	rpm	o							
6081 _h	profile_velocity	VAR	Unsigned32		rw	Possible	speed units			m		m			
6082 _h	end_velocity	VAR	Unsigned32		rw	Possible	speed units			o					
6083 _h	profile_acceleration	VAR	Unsigned32		rw	Possible	acceleration units			m		m			
6084 _h	profile_deceleration	VAR	Unsigned32		rw	Possible	acceleration units			m		m			
6085 _h	quick_stop_deceleration	VAR	Unsigned32		rw	Possible	acceleration units			o	o	o	o		
6086 _h	motion_profile_type	VAR	Integer16		rw	Possible	none			m		m			
60C5 _h	max_acceleration	VAR	Unsigned32		rw	Possible	acceleration units			o	o				
60C6 _h	max_deceleration	VAR	Unsigned32		rw	Possible	acceleration units			o	o				

A.5 Homing Mode

Object Dictionary for Chapter Homing Mode															
object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pv	tq	vl	
607C _h	home_offset	VAR	Integer32		rw	Possible	position units		o						
6098 _h	homing_method	VAR	Integer8		rw	Possible	-		m						
6099 _h	homing_speeds	ARRAY	Unsigned32												
	speed_during_search_for_switch	ARRAY		01	rw	Possible	velocity Units		m						
	speed_during_search_for_zero	ARRAY		02	rw	Possible	velocity units		m						
609A _h	homing_acceleration	VAR	Unsigned32		rw	Possible	acceleration units		o						

A.6 Position Control Function

Object Dictionary for Chapter Position Control Function															
object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pV	tq	vI	
6062 _h	position_demand_value	VAR	Integer32		ro	Possible	position units								
6063 _h	position_actual_value*	VAR	Integer32		ro	Possible	increments		o						
6064 _h	position_actual_value	VAR	Integer32		ro	Possible	position units		o						
6065 _h	following_error_window	VAR	Unsigned32		rw	Possible	position units								
6066 _h	following_error_time_out	VAR	Unsigned16		rw	Possible	milliseconds								
6067 _h	position_window	VAR	Unsigned32		rw	Possible	position units								
6068 _h	position_window_time	VAR	Unsigned16		rw	Possible	milliseconds								
60F4 _h	following_error_actual_value	VAR	Integer32		ro	Possible	position units		o						
60FA _h	control_effort	VAR	Integer32		ro	Possible	-		o						
60FB _h	position_control_parameter_set	RECORD													
60FC _h	position_demand_value*	VAR	Integer32		ro	Possible	increments		o						

A.7 Interpolated Position Mode

Object Dictionary for Chapter Interpolated Position Mode															
object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pv	tq	vl	
60C0 _h	interpolation_submode_select	VAR	Integer16		rw	Possible	-				o				
60C1 _h	interpolation_data_record	RECORD													
60C2 _h	interpolation_time_period	RECORD													
	ip_time_units	RECORD		01	rw	Possible	10ip_time_index (seconds				o				
	ip_time_index	RECORD		02	rw	Possible	-				o				
60C3 _h	interpolation_sync_definition	ARRAY	Unsigned8												
	synchronize on group	ARRAY		01	rw	Possible	number				o				
	ip_sync every n event	ARRAY		02	rw	Possible	counts				o				
60C4 _h	interpolation_data_configuration	RECORD													
	max_buffer_size	RECORD		01	ro	Possible	number				o				
	actual_size	RECORD		02	rw	Possible	number				o				
	buffer_organisation	RECORD		03	rw	Possible	number				o				
	buffer_position	RECORD		04	rw	Possible	number				o				
	size_of_data_record	RECORD		05	wo	Possible	number				o				
	buffer_clear	RECORD		06	wo	Possible	-				o				

A.8 Profile Velocity Mode

Object Dictionary for Chapter Profile Velocity Mode															
object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pv	tq	vl	
6069 _h	velocity_sensor_actual_value	VAR	Integer32		ro	Possible	increments/sec					m			
6069 _h	velocity_sensor_actual_value	VAR	Integer32		ro	Possible	increments/sec					m			
606A _h	sensor_selection_code	VAR	Integer16		rw	Possible	-					m			
606B _h	velocity_demand_value	VAR	Integer32		ro	Possible	velocity units G2					m			
606C _h	velocity_actual_value	VAR	Integer32		ro	Possible	velocity units G2					m			
606D _h	velocity_window	VAR	Unsigned16		rw	Possible	velocity units G2					o			
606E _h	velocity_window_time	VAR	Unsigned16		rw	Possible	millisecond					o			
606F _h	velocity_threshold	VAR	Unsigned16		rw	Possible	velocity units G2					o			
6070 _h	velocity_threshold_time	VAR	Unsigned16		r/w	Possible	millisecond					o			
60FF _h	target_velocity	VAR	Integer32		rw	Possible	velocity units G2					m			
60F8 _h	max_slippage	VAR	Integer32		ro	Possible	velocity units G2					m			
60F9 _h	velocity_control_parameter_set	RECORD													
	V: gain	RECORD		01	rw	Possible	-					o			
	Ti: integration time constant	RECORD		02	rw	Possible	-					o			
	vl_velocity_min_neg	RECORD		03	rw	Possible	G1b							o	

A.9 Profile Torque Mode

Object Dictionary for Chapter Profile Torque Mode															
object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pv	tq	vl	
6071 _h	target_torque	VAR	Integer16		rw	optional	per thousand of rated torque						m		
6072 _h	max_torque	VAR	Unsigned16		rw	optional	per thousand of rated torque			o			o		
6073 _h	max_current	VAR	Unsigned16		rw	optional	per thousand of rated current			o			o		
6074 _h	torque_demand_value	VAR	Integer16		ro	optional	per thousand of rated torque			o			o		
6075 _h	motor Rated current	VAR	Unsigned32		rw	optional	1 mA			o			o		
6076 _h	motor Rated torque	VAR	Unsigned32		rw	optional	0.001 Nm			o			o		
6077 _h	torque_actual_value	VAR	Integer16		ro	optional	per thousand of rated torque			o			o		
6078 _h	current_actual_value	VAR	Integer16		ro	optional	per thousand of rated current			o			o		
6079 _h	DC link circuit voltage	VAR	Unsigned32		ro	optional	milli volts						o		
6087 _h	torque_slope	VAR	Unsigned32		rw	optional	per thousand of rated torque per second						m		
6088 _h	torque_profile_type	VAR	Integer16		rw	optional	none						m		
60F7 _h	power_stage_parameters	RECORD													
		RECORD		01	rw	Possible	-			o			o		
60F6 _h	torque_control_parameters	RECORD													
		RECORD		01	rw	Possible	-			o			o		

A.10 Velocity Mode

Object Dictionary for Chapter Velocity Mode															
object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pv	tq	vl	
6042 _h	vl_target_velocity	VAR	Integer16		rw	Possible	Speed Units G1a							m	
6043 _h	vl_velocity_demand	VAR	Integer16		ro	Possible	Speed Units G1a							m	
6053 _h	vl_percentage_demand	VAR	Integer16		ro	Possible	(100/16383) %							o	
6054 _h	vl_actual_percentage	VAR	Integer16		ro	Possible	(100/16383) %							o	
6055 _h	vl_manipulated_percentage	VAR	Integer16		ro	Possible	(100/16383) %							o	
604E _h	vl_velocity_reference	VAR	Unsigned32		rw	Possible	Speed Units G1a							o	
604C _h	vl_dimension_factor	ARRAY	Integer32												
	vl_dimension_factor_numerator	ARRAY		01	rw	Possible	-(no units)							o	
	vl_dimension_factor_denominator	ARRAY		02	rw	Possible	-(no units)							o	
604B _h	vl_setpoint_factor	ARRAY	Integer16												
	vl_setpoint_factor_numerator	ARRAY		01	rw	Possible	-(no units)							o	
	vl_setpoint_factor_denominator	ARRAY		02	rw	Possible	-(no units)							o	
604D _h	vl_pole_number	VAR	Unsigned8		rw	Possible	- (number of poles)							o	
6046 _h	vl_velocity_min_max_amount	ARRAY	Unsigned32												
	vl_velocity_min_amount	ARRAY		01	rw	Possible	G1b							m	
	vl_velocity_max_amount	ARRAY		02	rw	Possible	G1b							m	
6047 _h	vl_velocity_min_max	ARRAY	Unsigned32												
	vl_velocity_min_pos	ARRAY		01	rw	Possible	G1b							o	
	vl_velocity_max_pos	ARRAY		02	rw	Possible	G1b							o	
	vl_velocity_min_neg	ARRAY		03	rw	Possible	G1b							o	
	vl_velocity_max_neg	ARRAY		04	rw	Possible	G1b							o	
6058 _h	vl_frequency_motor_min_max_amount	ARRAY	Unsigned32												
	vl_frequency_motor_min_amount	ARRAY		01	rw	Possible	-							o	
	vl_frequency_motor_max_amount	ARRAY		02	rw	Possible	-							o	
6059 _h	vl_frequency_motor_min_max	ARRAY	Unsigned32												
	vl_frequency_motor_min_pos	ARRAY		01	rw	Possible	-							o	
	vl_frequency_motor_max_pos	ARRAY		02	rw	Possible	-							o	
	vl_frequency_motor_min_neg	ARRAY		03	rw	Possible	- (1/1000 rpm)							o	
	vl_frequency_motor_max_neg	ARRAY		04	rw	Possible	-							o	

Object Dictionary for Chapter Velocity Mode

object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pv	tq	vl	
6056 _h	vl_velocity_motor_min_max_amount	ARRAY	Unsigned32												
	vl_velocity_motor_min_amount	ARRAY		01	rw	Possible	(1/1000 rpm)							o	
	vl_velocity_motor_max_amount	ARRAY		02	rw	Possible	(1/1000 rpm)							o	
6057 _h	vl_velocity_motor_min_max	ARRAY	Unsigned32												
	vl_velocity_motor_min_pos	ARRAY		01	rw	Possible	- (1/1000 rpm)							o	
	vl_velocity_motor_max_pos	ARRAY		02	rw	Possible	- (1/1000 rpm)							o	
	vl_velocity_motor_min_neg	ARRAY		03	rw	Possible	- (1/1000 rpm)							o	
	vl_velocity_motor_max_neg	ARRAY		04	rw	Possible	- (1/1000 rpm)							o	
6048 _h	vl_velocity_acceleration	RECORD													
	delta_speed	RECORD		01	rw	Possible	G1b							m	
	delta_time	RECORD		02	rw	Possible	Seconds (s)							m	
6049 _h	vl_velocity_deceleration	RECORD													
	delta_speed	RECORD		01	rw	Possible	G1b							m	
	delta_time	RECORD		02	rw	Possible	Seconds (s)							m	
604A _h	vl_velocity_quick_stop	RECORD													
	delta_speed	RECORD		01	rw	Possible	G1b							o	
	delta_time	RECORD		02	rw	Possible	Seconds (s)							o	
604F _h	vl_ramp_function_time	VAR	Unsigned32		rw	Possible	Milliseconds (ms)							o	
6050 _h	vl_slow_down_time	VAR	Unsigned32		rw	Possible	Milliseconds (ms)							o	
6051 _h	vl_quick_stop_time	VAR	Unsigned32		rw	Possible	Milliseconds (ms)							o	
6044 _h	vl_control_effort	VAR	Integer16		ro	Possible	Speed Units G1a							m	
6045 _h	vl_manipulated_velocity	VAR	Integer16		ro	Possible	Speed Units G1a							o	
6052 _h	vl_nominal_percentage	VAR	Integer16		rw	Possible	(100/16383) %							o	

B Object Dictionary by Index

Object Dictionary sorted by Index															
object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pV	tq	vl	
6007 _h	abort_connection_option_code	VAR	Integer16		rw	Possible	-	o							
603F _h	error_code	VAR	Unsigned16		ro	Possible	-	o							
6040 _h	controlword	VAR	Unsigned16		rw	Possible	-	m							
6041 _h	statusword	VAR	Unsigned16		ro	Possible	-	m							
6042 _h	vI_target_velocity	VAR	Integer16		rw	Possible	Speed Units G1a							m	
6043 _h	vI_velocity_demand	VAR	Integer16		ro	Possible	Speed Units G1a							m	
6044 _h	vI_control_effort	VAR	Integer16		ro	Possible	Speed Units G1a							m	
6045 _h	vI_manipulated_velocity	VAR	Integer16		ro	Possible	Speed Units G1a							o	
6046 _h	vI_velocity_min_max_amount	ARRAY	Unsigned32												
6047 _h	vI_velocity_min_max	ARRAY	Unsigned32												
6048 _h	vI_velocity_acceleration	RECORD													
6049 _h	vI_velocity_deceleration	RECORD													
604A _h	vI_velocity_quick_stop	RECORD													
604B _h	vI_setpoint_factor	ARRAY	Integer16												
604C _h	vI_dimension_factor	ARRAY	Integer32												
604D _h	vI_pole_number	VAR	Unsigned8		rw	Possible	- (number of poles)							o	
604E _h	vI_velocity_reference	VAR	Unsigned32		rw	Possible	Speed Units G1a							o	
604F _h	vI_ramp_function_time	VAR	Unsigned32		rw	Possible	Milliseconds (ms)							o	
6050 _h	vI_slow_down_time	VAR	Unsigned32		rw	Possible	Milliseconds (ms)							o	
6051 _h	vI_quick_stop_time	VAR	Unsigned32		rw	Possible	Milliseconds (ms)							o	
6052 _h	vI_nominal_percentage	VAR	Integer16		rw	Possible	(100/16383) %							o	
6053 _h	vI_percentage_demand	VAR	Integer16		ro	Possible	(100/16383) %							o	
6054 _h	vI_actual_percentage	VAR	Integer16		ro	Possible	(100/16383) %							o	
6055 _h	vI_manipulated_percentage	VAR	Integer16		ro	Possible	(100/16383) %							o	
6056 _h	vI_velocity_motor_min_max_amount	ARRAY	Unsigned32												
6057 _h	vI_velocity_motor_min_max	ARRAY	Unsigned32												
6058 _h	vI_frequency_motor_min_max_amount	ARRAY	Unsigned32												
6059 _h	vI_frequency_motor_min_max	ARRAY	Unsigned32												

Object Dictionary sorted by Index

object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pV	tq	vl	
605A _h	quick_stop_option_code	VAR	Integer16		rw	No	-	o							
605B _h	shutdown_option_code	VAR	Integer16		rw	No	-	o							
605C _h	disable_operation_option_code	VAR	Integer16		rw	No	-	o							
605D _h	stop_option_code	VAR	Integer16		rw	No	-	o							
605E _h	fault_reaction_option_code	VAR	Integer16		rw	No	-	o							
6060 _h	modes_of_operation	VAR	Integer8		wo	Possible	-	m							
6061 _h	modes_of_operation_display	VAR	Integer8		ro	Possible	-	m							
6062 _h	position_demand_value	VAR	Integer32		ro	Possible	position units								
6063 _h	position_actual_value*	VAR	Integer32		ro	Possible	increments		o						
6064 _h	position_actual_value	VAR	Integer32		ro	Possible	position units		o						
6065 _h	following_error_window	VAR	Unsigned32		rw	Possible	position units								
6066 _h	following_error_time_out	VAR	Unsigned16		rw	Possible	milliseconds								
6067 _h	position_window	VAR	Unsigned32		rw	Possible	position units								
6068 _h	position_window_time	VAR	Unsigned16		rw	Possible	milliseconds								
6069 _h	velocity_sensor_actual_value	VAR	Integer32		ro	Possible	increments/sec					m			
6069 _h	velocity_sensor_actual_value	VAR	Integer32		ro	Possible	increments/sec					m			
606A _h	sensor_selection_code	VAR	Integer16		rw	Possible	-					m			
606B _h	velocity_demand_value	VAR	Integer32		ro	Possible	velocity units G2					m			
606C _h	velocity_actual_value	VAR	Integer32		ro	Possible	velocity units G2					m			
606D _h	velocity_window	VAR	Unsigned16		rw	Possible	velocity units G2					o			
606E _h	velocity_window_time	VAR	Unsigned16		rw	Possible	millisecond					o			
606F _h	velocity_threshold	VAR	Unsigned16		rw	Possible	velocity units G2					o			
6070 _h	velocity_threshold_time	VAR	Unsigned16		r/w	Possible	millisecond					o			
6071 _h	target_torque	VAR	Integer16		rw	optional	per thousand of rated torque						m		
6072 _h	max_torque	VAR	Unsigned16		rw	optional	per thousand of rated torque			o			o		
6073 _h	max_current	VAR	Unsigned16		rw	optional	per thousand of rated current			o			o		
6074 _h	torque_demand_value	VAR	Integer16		ro	optional	per thousand of rated torque			o			o		
6075 _h	motor rated current	VAR	Unsigned32		rw	optional	1 mA			o			o		

Object Dictionary sorted by Index															
object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pv	tq	vl	
6076h	motor Rated torque	VAR	Unsigned32		rw	optional	0.001 Nm			o			o		
6077h	torque_actual_value	VAR	Integer16		ro	optional	per thousand of rated torque			o			o		
6078h	current_actual_value	VAR	Integer16		ro	optional	per thousand of rated current			o			o		
6079h	DC link circuit voltage	VAR	Unsigned32		ro	optional	milli volts						o		
607A _h	target_position	VAR	Integer32		rw	Possible	position units			m	m				
607B _h	position_range_limit	ARRAY	Integer32												
607C _h	home_offset	VAR	Integer32		rw	Possible	position units		o						
607D _h	software_position_limit	ARRAY	Integer32												
607E _h	polarity	VAR	Unsigned8		rw	Possible	-			o	o	o	o		
607F _h	max_profile_velocity	VAR	Unsigned32		rw	Possible	speed units			o	o	o	o		
6080 _h	max_motor_speed	VAR	Unsigned16		rw	Possible	rpm	o							
6081 _h	profile_velocity	VAR	Unsigned32		rw	Possible	speed units			m		m			
6082 _h	end_velocity	VAR	Unsigned32		rw	Possible	speed units			o					
6083 _h	profile_acceleration	VAR	Unsigned32		rw	Possible	acceleration units			m		m			
6084 _h	profile_deceleration	VAR	Unsigned32		rw	Possible	acceleration units			m		m			
6085 _h	quick_stop_deceleration	VAR	Unsigned32		rw	Possible	acceleration units			o	o	o	o		
6086 _h	motion_profile_type	VAR	Integer16		rw	Possible	none			m		m			
6087h	torque_slope	VAR	Unsigned32		rw	optional	per thousand of rated torque per second						m		
6088h	torque_profile_type	VAR	Integer16		rw	optional	none						m		
6089 _h	position_notation_index	VAR	Integer8		rw	Possible	-		o	o	o	o	o		
608A _h	position_dimension_index	VAR	Unsigned8		rw	Possible	-		o	o	o	o	o		
608B _h	velocity_notation_index	VAR	Integer8		rw	Possible	-		o	o	o	o	o		
608C _h	velocity_dimension_index	VAR	Unsigned8		rw	Possible	-		o	o	o	o	o		
608D _h	acceleration_notation_index	VAR	Integer8		rw	Possible	-		o	o	o	o	o		
608E _h	acceleration_dimension_index	VAR	Unsigned8		rw	Possible	-		o	o	o	o	o		
608F _h	position_encoder_resolution	ARRAY	Unsigned32												
6090 _h	velocity_encoder_resolution	ARRAY	Unsigned32												

Object Dictionary sorted by Index

object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pV	tq	vl	
6091 _h	gear_ratio	ARRAY	Unsigned32												
6092 _h	feed_constant	ARRAY	Unsigned32												
6093 _h	position_factor	ARRAY	Unsigned32												
6094 _h	velocity_encoder_factor	ARRAY	Unsigned32												
6095 _h	velocity_factor_1	ARRAY	Unsigned32												
6096 _h	velocity_factor_2	ARRAY	Unsigned32												
6097 _h	acceleration_factor	ARRAY	Unsigned32												
6098 _h	homing_method	VAR	Integer8		rw	Possible	-		m						
6099 _h	homing_speeds	ARRAY	Unsigned32												
609A _h	homing_acceleration	VAR	Unsigned32		rw	Possible	acceleration units		o						
60C0 _h	interpolation_submode_select	VAR	Integer16		rw	Possible	-				o				
60C1 _h	interpolation_data_record	RECORD													
60C2 _h	interpolation_time_period	RECORD													
60C3 _h	interpolation_sync_definition	ARRAY	Unsigned8												
60C4 _h	interpolation_data_configuration	RECORD													
60C5 _h	max_acceleration	VAR	Unsigned32		rw	Possible	acceleration units			o	o				
60C6 _h	max_deceleration	VAR	Unsigned32		rw	Possible	acceleration units			o	o				
60F4 _h	following_error_actual_value	VAR	Integer32		ro	Possible	position units		o						
60F6 _h	torque_control_parameters	RECORD													
60F7 _h	power_stage_parameters	RECORD													
60F8 _h	max_slippage	VAR	Integer32		ro	Possible	velocity units G2					m			
60F9 _h	velocity_control_parameter_set	RECORD													
60FA _h	control_effort	VAR	Integer32		ro	Possible	-		o						
60FB _h	position_control_parameter_set	RECORD													
60FC _h	position_demand_value*	VAR	Integer32		ro	Possible	increments		o						
60FD _h	digital_inputs	VAR	Unsigned32		ro	Possible	-	o							
60FE _h	digital_outputs	RECORD													
60FF _h	target_velocity	VAR	Integer32		rw	Possible	velocity units G2					m			
6402 _h	motor_type	VAR	Unsigned16		rw	Possible	-								
6403 _h	motor_catalogue_number	VAR	Visible String		rw	Possible	-								
6404 _h	motor_manufacturer	VAR	Visible String		rw	Possible	-								

Object Dictionary sorted by Index															
object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pv	tq	vl	
6405 _h	http_motor_catalog_address	VAR	Visible String		rw	Possible	-								
6406 _h	motor_calibration_date	VAR	Date		rw	Possible	-								
6407 _h	motor_service_period	VAR	Unsigned32		rw	Possible	-								
6410 _h	motor_data	RECORD													
6502 _h	supported_drive_modes	VAR	Unsigned32		ro	Possible	-								
6503 _h	drive_catalogue_number	VAR	Visible String		rw	Possible	-								
6504 _h	drive_manufacturer	VAR	Visible String		rw	Possible	-								
6505 _h	http_drive_catalog_address	VAR	Visible String		rw	Possible	-								
6510 _h	drive_data	RECORD													

C Object Dictionary by Name

Object Dictionary sorted by Name															
object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pv	tq	vl	
6007 _h	abort_connection_option_code	VAR	Integer16		rw	Possible	-	o							
608E _h	acceleration_dimension_index	VAR	Unsigned8		rw	Possible	-		o	o	o	o	o		
6097 _h	acceleration_factor	ARRAY	Unsigned32												
608D _h	acceleration_notation_index	VAR	Integer8		rw	Possible	-		o	o	o	o	o		
60FA _h	control_effort	VAR	Integer32		ro	Possible	-		o						
6040 _h	controlword	VAR	Unsigned16		rw	Possible	-	m							
6078 _h	current_actual_value	VAR	Integer16		ro	optional	per thousand of rated current			o			o		
6079 _h	DC link circuit voltage	VAR	Unsigned32		ro	optional	milli volts						o		
60FD _h	digital_inputs	VAR	Unsigned32		ro	Possible	-	o							
60FE _h	digital_outputs	RECORD													
605C _h	disable_operation_option_code	VAR	Integer16		rw	No	-	o							
6503 _h	drive_catalogue_number	VAR	Visible String		rw	Possible	-								
6510 _h	drive_data	RECORD													
6504 _h	drive_manufacturer	VAR	Visible String		rw	Possible	-								
6082 _h	end_velocity	VAR	Unsigned32		rw	Possible	speed units			o					
603F _h	error_code	VAR	Unsigned16		ro	Possible	-	o							
605E _h	fault_reaction_option_code	VAR	Integer16		rw	No	-	o							
6092 _h	feed_constant	ARRAY	Unsigned32												
60F4 _h	following_error_actual_value	VAR	Integer32		ro	Possible	position units		o						
6066 _h	following_error_time_out	VAR	Unsigned16		rw	Possible	milliseconds								
6065 _h	following_error_window	VAR	Unsigned32		rw	Possible	position units								
6091 _h	gear_ratio	ARRAY	Unsigned32												
607C _h	home_offset	VAR	Integer32		rw	Possible	position units		o						
609A _h	homing_acceleration	VAR	Unsigned32		rw	Possible	acceleration units		o						
6098 _h	homing_method	VAR	Integer8		rw	Possible	-		m						
6099 _h	homing_speeds	ARRAY	Unsigned32												
6505 _h	http_drive_catalog_address	VAR	Visible String		rw	Possible	-								

Object Dictionary sorted by Name															
object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pv	tq	vl	
6405 _h	http_motor_catalog_address	VAR	Visible String		rw	Possible	-								
60C4 _h	interpolation_data_configuration	RECORD													
60C1 _h	interpolation_data_record	RECORD													
60C0 _h	interpolation_submode_select	VAR	Integer16		rw	Possible	-				o				
60C3 _h	interpolation_sync_definition	ARRAY	Unsigned8												
60C2 _h	interpolation_time_period	RECORD													
60C5 _h	max_acceleration	VAR	Unsigned32		rw	Possible	acceleration units			o	o				
6073 _h	max_current	VAR	Unsigned16		rw	optional	per thousand of rated current			o			o		
60C6 _h	max_deceleration	VAR	Unsigned32		rw	Possible	acceleration units			o	o				
6080 _h	max_motor_speed	VAR	Unsigned16		rw	Possible	rpm	o							
607F _h	max_profile_velocity	VAR	Unsigned32		rw	Possible	speed units			o	o	o	o		
60F8 _h	max_slippage	VAR	Integer32		ro	Possible	velocity units G2					m			
6072 _h	max_torque	VAR	Unsigned16		rw	optional	per thousand of rated torque			o			o		
6060 _h	modes_of_operation	VAR	Integer8		wo	Possible	-	m							
6061 _h	modes_of_operation_display	VAR	Integer8		ro	Possible	-	m							
6086 _h	motion_profile_type	VAR	Integer16		rw	Possible	none			m		m			
6406 _h	motor_calibration_date	VAR	Date		rw	Possible	-								
6403 _h	motor_catalogue_number	VAR	Visible String		rw	Possible	-								
6410 _h	motor_data	RECORD													
6404 _h	motor_manufacturer	VAR	Visible String		rw	Possible	-								
6075 _h	motor rated current	VAR	Unsigned32		rw	optional	1 mA			o			o		
6076 _h	motor rated torque	VAR	Unsigned32		rw	optional	0.001 Nm			o			o		
6407 _h	motor_service_period	VAR	Unsigned32		rw	Possible	-								
6402 _h	motor_type	VAR	Unsigned16		rw	Possible	-								
607E _h	polarity	VAR	Unsigned8		rw	Possible	-			o	o	o	o		
6064 _h	position_actual_value	VAR	Integer32		ro	Possible	position units		o						
6063 _h	position_actual_value*	VAR	Integer32		ro	Possible	increments		o						
60FB _h	position_control_parameter_set	RECORD													
6062 _h	position_demand_value	VAR	Integer32		ro	Possible	position units								

Object Dictionary sorted by Name

object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pv	tq	vl	
60FC _h	position_demand_value*	VAR	Integer32		ro	Possible	increments		o						
608A _h	position_dimension_index	VAR	Unsigned8		rw	Possible	-		o	o	o	o	o		
608F _h	position_encoder_resolution	ARRAY	Unsigned32												
6093 _h	position_factor	ARRAY	Unsigned32												
6089 _h	position_notation_index	VAR	Integer8		rw	Possible	-		o	o	o	o	o		
607B _h	position_range_limit	ARRAY	Integer32												
6067 _h	position_window	VAR	Unsigned32		rw	Possible	position units								
6068 _h	position_window_time	VAR	Unsigned16		rw	Possible	milliseconds								
60F7 _h	power_stage_parameters	RECORD													
6083 _h	profile_acceleration	VAR	Unsigned32		rw	Possible	acceleration units			m		m			
6084 _h	profile_deceleration	VAR	Unsigned32		rw	Possible	acceleration units			m		m			
6081 _h	profile_velocity	VAR	Unsigned32		rw	Possible	speed units			m		m			
6085 _h	quick_stop_deceleration	VAR	Unsigned32		rw	Possible	acceleration units			o	o	o	o		
605A _h	quick_stop_option_code	VAR	Integer16		rw	No	-	o							
606A _h	sensor_selection_code	VAR	Integer16		rw	Possible	-					m			
605B _h	shutdown_option_code	VAR	Integer16		rw	No	-	o							
607D _h	software_position_limit	ARRAY	Integer32												
6041 _h	statusword	VAR	Unsigned16		ro	Possible	-	m							
605D _h	stop_option_code	VAR	Integer16		rw	No	-	o							
6502 _h	supported_drive_modes	VAR	Unsigned32		ro	Possible	-								
607A _h	target_position	VAR	Integer32		rw	Possible	position units			m	m				
6071 _h	target_torque	VAR	Integer16		rw	optional	per thousand of rated torque						m		
60FF _h	target_velocity	VAR	Integer32		rw	Possible	velocity units G2					m			
6077 _h	torque_actual_value	VAR	Integer16		ro	optional	per thousand of rated torque			o			o		
60F6 _h	torque_control_parameters	RECORD													
6074 _h	torque_demand_value	VAR	Integer16		ro	optional	per thousand of rated torque			o			o		
6088 _h	torque_profile_type	VAR	Integer16		rw	optional	none						m		

Object Dictionary sorted by Name															
object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pv	tq	vl	
6087h	torque_slope	VAR	Unsigned32		rw	optional	per thousand of rated torque per second						m		
606Ch	velocity_actual_value	VAR	Integer32		ro	Possible	velocity units G2					m			
60F9h	velocity_control_parameter_set	RECORD													
606Bh	velocity_demand_value	VAR	Integer32		ro	Possible	velocity units G2					m			
608Ch	velocity_dimension_index	VAR	Unsigned8		rw	Possible	-		o	o	o	o	o		
6094h	velocity_encoder_factor	ARRAY	Unsigned32												
6090h	velocity_encoder_resolution	ARRAY	Unsigned32												
6095h	velocity_factor_1	ARRAY	Unsigned32												
6096h	velocity_factor_2	ARRAY	Unsigned32												
608Bh	velocity_notation_index	VAR	Integer8		rw	Possible	-		o	o	o	o	o		
6069h	velocity_sensor_actual_value	VAR	Integer32		ro	Possible	increments/sec					m			
6069h	velocity_sensor_actual_value	VAR	Integer32		ro	Possible	increments/sec					m			
606Fh	velocity_threshold	VAR	Unsigned16		rw	Possible	velocity units G2					o			
6070h	velocity_threshold time	VAR	Unsigned16		r/w	Possible	millisecond					o			
606Dh	velocity_window	VAR	Unsigned16		rw	Possible	velocity units G2					o			
606Eh	velocity_window time	VAR	Unsigned16		rw	Possible	millisecond					o			
6054h	vl_actual_percentage	VAR	Integer16		ro	Possible	(100/16383) %							o	
6044h	vl_control_effort	VAR	Integer16		ro	Possible	Speed Units G1a							m	
604Ch	vl_dimension_factor	ARRAY	Integer32												
6059h	vl_frequency_motor_min_max	ARRAY	Unsigned32												
6058h	vl_frequency_motor_min_max amount	ARRAY	Unsigned32												
6055h	vl_manipulated_percentage	VAR	Integer16		ro	Possible	(100/16383) %							o	
6045h	vl_manipulated_velocity	VAR	Integer16		ro	Possible	Speed Units G1a							o	
6052h	vl_nominal_percentage	VAR	Integer16		rw	Possible	(100/16383) %							o	
6053h	vl_percentage_demand	VAR	Integer16		ro	Possible	(100/16383) %							o	
604Dh	vl_pole number	VAR	Unsigned8		rw	Possible	- (number of poles)							o	
6051h	vl_quick_stop_time	VAR	Unsigned32		rw	Possible	Milliseconds (ms)							o	
604Fh	vl_ramp function time	VAR	Unsigned32		rw	Possible	Milliseconds (ms)							o	
604Bh	vl_setpoint factor	ARRAY	Integer16												

Object Dictionary sorted by Name

object no	object name	object code	type	sub-index	attr.	PDO	units	mandatory for							description
								all	hm	pp	ip	pv	tq	vl	
6050h	vl_slow_down_time	VAR	Unsigned32		rw	Possible	Milliseconds (ms)							o	
6042h	vl_target_velocity	VAR	Integer16		rw	Possible	Speed Units G1a							m	
6048h	vl_velocity_acceleration	RECORD													
6049h	vl_velocity_deceleration	RECORD													
6043h	vl_velocity_demand	VAR	Integer16		ro	Possible	Speed Units G1a							m	
6047h	vl_velocity_min_max	ARRAY	Unsigned32												
6046h	vl_velocity_min_max_amount	ARRAY	Unsigned32												
6057h	vl_velocity_motor_min_max	ARRAY	Unsigned32												
6056h	vl_velocity_motor_min_max_amount	ARRAY	Unsigned32												
604Ah	vl_velocity_quick_stop	RECORD													
604Eh	vl_velocity_reference	VAR	Unsigned32		rw	Possible	Speed Units G1a							o	

D Definition of Dimension Indices

D.1 Dimension/Notation Index Table

physical dimension	dimension index	units exponent	unit typ	notation index
non	0	units		0
length	1		metre	0
		milli	metre	-3
		kilo	metre	3
		micro	metre	-6
area	2	square	metre	0
		square milli	metre	-6
		square kilo	metre	6
volume	3	cubic	metre	0
time	4		second	0
			minute	70
			hour	74
			day	77
		milli	second	-3
		micro	second	-6
actual power	9		watt	0
		kilo	watt	3
		mega	watt	6
		milli	watt	-3
apparent power	10		voltampere	0
		kilo	voltampere	3
		mega	voltampere	6
no. of revolutions	11		per second	0
			per minute	73
			per hour	74
angle	12		radian	0
			second	75
			minute	76
			degree	77
			newdegree	78

physical dimension	dimension index	units exponent	unit typ	notation index
velocity	13		metre p. second	0
		milli	metre p. second	-3
		milli	metre p. minute	79
			metre p. minute	80
		kilo	metre p. minute	81
		milli	metre p. hour	82
			metre p. hour	83
		kilo	metre p. hour	84
torque	16		newton metre	0
		kilo	newton metre	3
		mega	newton metre	6
temperature	17		kelvin	0
			centigrade	94
			Fahrenheit	95
voltage	21		Volt	0
		kilo	Volt	3
		milli	Volt	-3
		micro	Volt	-6
current	22		Ampere	0
		kilo	Ampere	3
		milli	Ampere	-3
		micro	Ampere	-6
ratio	24		percent	0
frequency	28		Hertz	0
		kilo	Hertz	3
		mega	Hertz	6
		giga	Hertz	9
steps	32		steps	0
encoder resolution	33	revolution	steps per	0

Table 14: Dimension/Notation Index Table

D.2 Examples for Notation Indices

Examples for notation indices < 64:

For notation index <64 the value is used as an exponent. The unit is defined by the physical dimension and calculated by unit type and exponent, all declared in the dimension/notation index table above.

position unit

dimension index = 1: length

notation index = -6: micro meter

$$\begin{aligned}\text{position_units} &= 10^{\text{notation_index}} \times f(\text{dimension_index}) \\ &= 10^{-6} \text{ m}\end{aligned}$$

dimension index = 12: angle

notation index = 0: radian

$$\begin{aligned}\text{position_units} &= 10^{\text{notation_index}} \times f(\text{dimension_index}) \\ &= \text{radian}\end{aligned}$$

velocity unit

dimension index = 13: velocity

notation index = -3: milli metre per second

$$\begin{aligned}\text{velocity_units} &= 10^{\text{notation_index}} \times f(\text{dimension_index}) \\ &= 10^{-3} \text{ m/s}\end{aligned}$$

frequency units

dimension index = 28: frequency

notation index = 3: kilo hertz

$$\begin{aligned}\text{frequency_units} &= 10^{\text{notation_index}} \times f(\text{dimension_index}) \\ &= 10^3 \text{ Hz}\end{aligned}$$

Examples for notation indices > 64:

The unit is defined by the physical dimension and unit type, both declared in the dimension/notation index table.

time units

dimension index = 4: time

notation index = 77: day

$$\begin{aligned}\text{time_units} &= f(\text{dimension_index}, \text{notation_index}) \\ &= \text{day}\end{aligned}$$

position unit

dimension index = 12: angle

notation index = 76: minute

$$\begin{aligned}\text{position_units} &= f(\text{dimension_index}, \text{notation_index}) \\ &= \text{minute}\end{aligned}$$

E Index

*	74
Abbreviation	10
abort_connection_option_code	37
acceleration_dimension_index	83
acceleration_factor	92
acceleration_notation_index	82
Acronyms	10
Appendix	19-1
asterisk	74
broadcast	134
buffer_clear	150
buffer_position	150
CAN	10
ce	10
change_set_immediatly	106
Chapters	
Common entries in the object dictionary	35
Device Control	49
Emergency Messages	19
Factor Group	74
Homing Mode	109
Interpolated Position Mode	134
Object Dictionary	34
Operating Principle	16
Position Control Function	120
Predefinitions	24
Profile Position Mode	94
Profile Velocity Mode	151
Profile Torque Mode	163
Velocity Mode	174
CiA	10
CMS	10
COB	10
COB-ID	10
Communication Object	10
Communication Profile Area	18
Complex Data Types	17
Control Loop	15
control_effort	130
Control-Functions	55
controlword	55
current_actual_value	169
DBT	10
dc	10
DC_link_circuit_voltage	170
Definitions	10
Device Control	49
Device Control Commands	57
device profile area	18
Device type	24, 25
Device-Controlling	49
digital_inputs	46
digital_outputs	47
Dimension Indices, Definition	19-23

disable_operation_option_code	64
Drive Data	35
drive disabling function	71
Drive States	51
Drive Supervisor	53
drive_catalogue_number	44
drive_data	45
drive_manufacturer	44
Emergency messages	19
end_velocity	102
error classification	19
Error code	19
Error Register	25
error_code	38
Factors	74
Fault	52
Fault Reaction	72
Fault Reaction Active	52
fault_reaction_option_code	67
feed_constant	87
fifo	149
Following_error	120
following_error_actual_value	129
following_error_time_out	127
following_error_window	126
force	163
gear_ratio	86
general sync signal	134
group sync signal	134
hm	10, 109
home_offset	110
Homing Function	109
Homing Methods	114
Homing Mode	109
homing_acceleration	113
homing_method	111
homing_speeds	112
http_drive_catalog_address	45
http_motor_catalog_address	40
IDO	11
Index	18
Index Pulse	115, 116, 118
Input Buffer	149
Internal Data Object	11
internal units	74
Interpolated Position Mode	134
interpolation algorithm	134
interpolation data buffer organisation	150
interpolation motion controller	136
Interpolation Position Mode	147
interpolation_data	134
interpolation_data_configuration	134, 144, 149
interpolation_data_record	134, 140, 149
interpolation_submode_select	140
interpolation_sync_definition	143
interpolation_time_period	134, 142
ip 10	
ip_data_record	149
Limit Switch	114, 115
linear interpolation	149

Linear Interpolation Position Mode	147
LMT	10
MAC	10
Manufacturer Specific Data Types	17
max_current	167
max_acceleration	105
max_deceleration	106
max_motor_speed	101
max_profile_velocity	100
max_slippage	161
max_torque	166
modes_of_operation	68
modes_of_operation_display	69
motion_profile_type	104
Motor Data	35
motor_calibration_date	41
motor_catalogue_number	39
motor_data	42
motor_manufacturer	40
motorRatedCurrent	168
motorRatedTorque	168
motor_service_period	41
motor_type	38
Naming conventions	24
NMT	10
Not Ready to Switch On	52
Object 1000 _h	24
Object 1001 _h	25
Object 6007 _h	37
Object 603F _h	38
Object 6040 _h	55
Object 6041 _h	58
Object 6042 _h	180
Object 6043 _h	181
Object 6044 _h	208
Object 6045 _h	208
Object 6046 _h	190
Object 6047 _h	191
Object 6048 _h	200
Object 6049 _h	202
Object 604A _h	204
Object 604B _h	188
Object 604C _h	186
Object 604D _h	189
Object 604E _h	185
Object 604F _h	205
Object 6050 _h	206
Object 6051 _h	207
Object 6052 _h	209
Object 6053 _h	182
Object 6054 _h	183
Object 6055 _h	184
Object 6056 _h	197
Object 6057 _h	198
Object 6058 _h	193
Object 6059 _h	195
Object 605A _h	65
Object 605B _h	63
Object 605C _h	64
Object 605D _h	66

Object 605E _h	67
Object 6060 _h	68
Object 6061 _h	69
Object 6062 _h	124
Object 6063 _h	124
Object 6064 _h	125
Object 6065 _h	126
Object 6066 _h	127
Object 6067 _h	128
Object 6068 _h	129
Object 6069 _h	156
Object 606A _h	156
Object 606B _h	157
Object 606C _h	158
Object 606D _h	158
Object 606E _h	159
Object 606F _h	159
Object 6070 _h	160
Object 6071 _h	166
Object 6072 _h	166
Object 6073 _h	167
Object 6074 _h	167
Object 6075 _h	168
Object 6076 _h	168
Object 6077 _h	169
Object 6078 _h	169
Object 6079 _h	170
Object 607A _h	97
Object 607B _h	98
Object 607C _h	110
Object 607D _h	99
Object 607E _h	93
Object 607F _h	100
Object 6080 _h	101
Object 6081 _h	101
Object 6082 _h	102
Object 6083 _h	102
Object 6084 _h	103
Object 6085 _h	103
Object 6086 _h	104
Object 6087 _h	170
Object 6088 _h	171
Object 6089 _h	78
Object 608A _h	79
Object 608B _h	80
Object 608C _h	81
Object 608D _h	82
Object 608E _h	83
Object 608F _h	84
Object 6090 _h	85
Object 6091 _h	86
Object 6092 _h	87
Object 6093 _h	88
Object 6094 _h	89
Object 6095 _h	90
Object 6096 _h	91
Object 6097 _h	92
Object 6098 _h	111
Object 6099 _h	112
Object 609A _h	113

Object 60C0 _h	140
Object 60C1 _h	140
Object 60C2 _h	142
Object 60C3 _h	143
Object 60C4 _h	144
Object 60C5 _h	105
Object 60C6 _h	106
Object 60F4 _h	129
Object 60F6 _h	173
Object 60F7 _h	172
Object 60F8 _h	161
Object 60F9 _h	162
Object 60FA _h	130
Object 60FB _h	131
Object 60FC _h	132
Object 60FD _h	46
Object 60FE _h	47
Object 60FF _h	160
Object 6402 _h	38
Object 6403 _h	39
Object 6404 _h	40
Object 6405 _h	40
Object 6406 _h	41
Object 6407 _h	41
Object 6410 _h	42
Object 6502 _h	43
Object 6503 _h	44
Object 6504 _h	44
Object 6505 _h	45
Object 6510 _h	45
Object 67FF _h	25
Object Dictionary.....	17, 34
Object Dictionary by Chapter.....	19-1
Object Dictionary by Index.....	19-13
Object Dictionary by Name.....	19-18
Operation Enable.....	52
Operation Mode.....	13
Operation mode function.....	70
Operation Modes.....	
hm.....	109
Profile Velocity Mode.....	151
pv mode.....	151
Profile Torque Mode.....	163
tq 163.....	
tq mode.....	163
Velocity Mode.....	174
pc 10.....	
PDO.....	11
PDO Mapping.....	25
physical in-/outputs.....	
digital inputs.....	46
digital outputs.....	47
physical units.....	74
polarity.....	93
Position Control Function.....	120
Position Range Limit.....	98
position_actual_value.....	125
position_actual_value*.....	124
position_control_parameter_set.....	131
position_demand_value.....	124, 134

position_demand_value*	132
position_dimension_index	79
position_encoder	155
position_encoder_resolution	84
position_factor	88
position_notation_index	78
Position_reached	121
position_window	128
position_window_time	129
power_stage_parameters	172
pp 10	
pre_defined_error_field	38
Predefined Objects	24
Process Data Object	11
Profile Position Mode	94
Profile Velocity Mode	151
profile_acceleration	102
profile_deceleration	103
profile_velocity	101
pv 10, 151	
Quick Stop Active	52
quick stop function	71
quick_stop_deceleration	103
quick_stop_option_code	65
Ready to Switch On	52
Receive PDOs	26
reserved areas	18
Ring Buffer	149
SDO	11
sensor_selection_code	156
Service Data Object	11
Setpoint, Set of	106
Setpoint, Single	106
setpoint_acknowledge	106
shutdown_option_code	63
software_position_limit	99
standard device	16
State Diagram	51
State Transitions	53
statemachine	49
Statemachine	50
Static Data Types	17
statusword	58
stop_option_code	66
sub-index	18
supported_drive_modes	43
Switch On Disabled	52
Switched On State	52
synch_signal	148
target_position	97
target_torque	163, 166
target_velocity	160
Profile Torque Mode	163
torque_actual_value	169
torque_control_parameters	163, 173
torque_demand_value	167
torque_profile_type	171
torque_slope	170
tq 10	
trajectory generator	95

Trajectory Generator	14
Transmit PDOs.....	30
Velocity Mode	174, 175, 177
Closed_open_loop_control_function.....	215
Factor_function.....	210
General Description.....	174
Percentage_function	210
Pole_number_function.....	211
Ramp_function	213
Ramp_min_function	214
Reference_calculation.....	215
Reverse_factor_function.....	210
Structure of	175
Velocity_limit_function.....	212
Velocity_motor_limit_function	212
vl_actual_percentage	183
vl_dimension_factor	186
vl_frequency_motor_min_max_amount	193
vl_manipulated_percentage	184
vl_manipulated_velocity.....	208
vl_nominal_percentage.....	209
vl_percentage_demand	182
vl_pole_number	189
vl_ramp_function_time.....	205
vl_setpoint_factor	188
vl_velocity_acceleration	200
vl_velocity_deceleration.....	202
vl_velocity_demand.....	181
vl_velocity_min_max	191
vl_velocity_min_max_amount.....	190
vl_velocity_motor_min_max	198
vl_velocity_quick_stop.....	204
vl_velocity_reference.....	185
velocity_actual_value.....	158
velocity_control_parameter_set	162
velocity_demand_value.....	157
velocity_dimension_index.....	81
velocity_encoder_factor.....	89
velocity_encoder_resolution	85
velocity_factor_1	90
velocity_factor_2	91
velocity_notation_index	80
velocity_sensor_actual_value.....	156
velocity_threshold.....	159
velocity_threshold_time	160
velocity_window	158
velocity_window_time	159
vl 10	
vl_control_effort	208
vl_frequency_motor_min_max	195
vl_quick_stop_time	207
vl_slow_down_time	206
vl_target_velocity.....	180
vl_velocity_motor_min_max_amount.....	197
vm	174