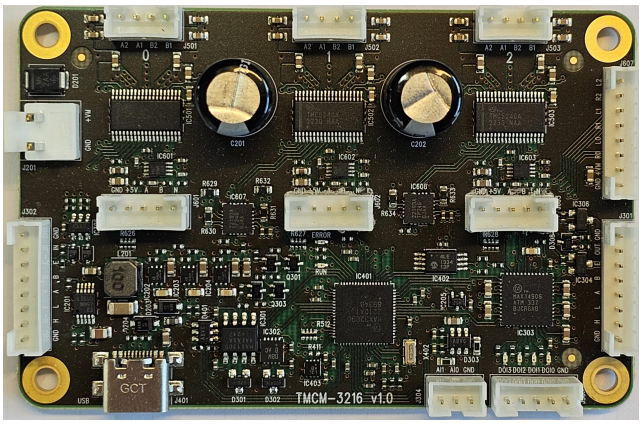
TMCM-3216 CANopen Firmware Manual

Firmware Version V1.04 | V0: 01/25

The TMCM-3216 is a three-axis controller/driver module for two-phase bipolar stepper motors with separate home and stop switch inputs and an encoder input for each axis. Dynamic current control and quiet, smooth, and efficient operation are combined with StealthChop, DcStep, StallGuard and CoolStep features. The module also offers four digital general-purpose inputs/outputs as well as two analog inputs.



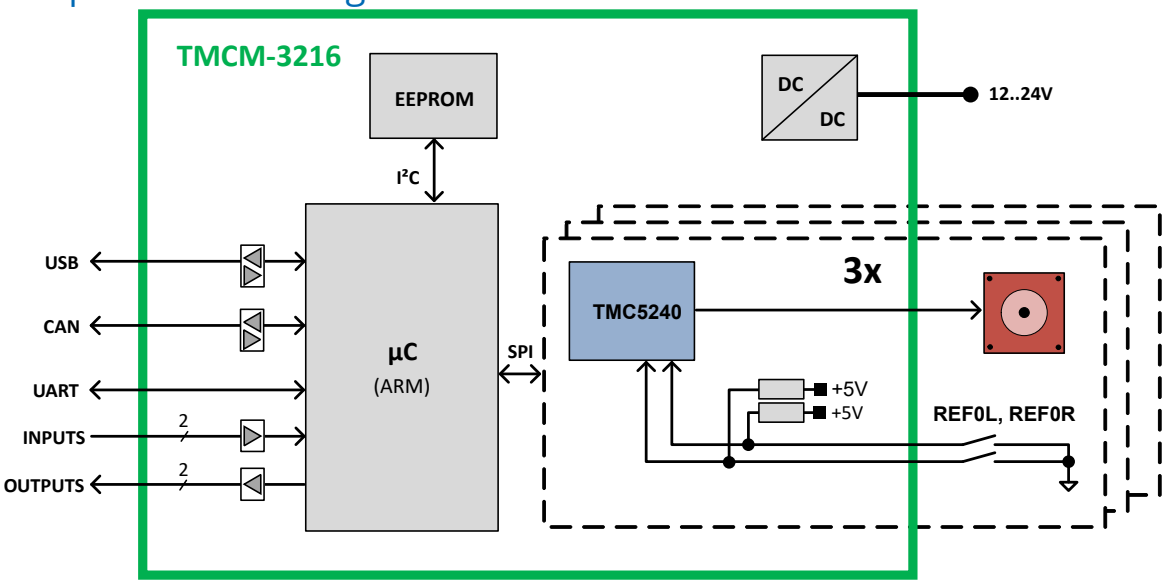
Features

- Three-axis stepper motor control
- ABN encoder input for each axis
- CANopen CiA-402 drive profile
- EightPoint ramp generator
- CoolStep
- DcStep
- StallGuard2 and StallGuard4
- StealthChop2

Applications

- Laboratory Automation
- Manufacturing
- Semiconductor Handling
- Robotics
- Factory Automation
- Test and Measurement
- Life Science
- Biotechnology
- Liquid Handling

Simplified Block Diagram



ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

Read entire documentation.

Contents

1 Preface **7**
1.1 General Features of this CANopen Implementation . . . 7
1.2 Abbreviations Used in this Manual . . . 8
1.3 Firmware Update . . . 9
1.4 Trinamic's Unique Features — Easy to Use with CANopen . . . 9
1.4.1 StallGuard2 . . . 9
1.4.2 CoolStep . . . 9
1.4.3 StallGuard4 . . . 10
1.5 EightPoint Motion Controller . . . 10

2 Communication **13**
2.1 Reference Model . . . 13
2.2 NMT State Machine . . . 15
2.3 LSS State Machine . . . 16
2.4 Indicator LEDs . . . 16
2.4.1 Run LED . . . 16
2.4.2 Error LED . . . 17
2.5 Device Model . . . 17
2.6 Object Dictionary . . . 18
2.6.1 Object Indices on Multi-Axis Modules . . . 18

3 Communication Area **20**
3.1 Detailed Object Specifications . . . 20
3.1.1 Object 1000_h: Device Type . . . 20
3.1.2 Object 1001_h: Error Register . . . 20
3.1.3 Object 1005_h: COB-ID SYNC . . . 21
3.1.4 Object 1008_h: Manufacturer Device Name . . . 22
3.1.5 Object 1009_h: Manufacturer Hardware Version . . . 22
3.1.6 Object $100A_h$: Manufacturer Software Version . . . 22
3.1.7 Object 1010_h: Store Parameters . . . 23
3.1.8 Object 1011_h: Restore Parameters . . . 24
3.1.9 Object 1014_h: COB-ID Emergency Object . . . 25
3.1.10 Object 1015_h: Inhibit Time EMCY . . . 26
3.1.11 Object 1016_h: Consumer Heartbeat Time . . . 26
3.1.12 Object 1017_h: Producer Heartbeat Time . . . 27
3.1.13 Object 1018_h: Identity Object . . . 27
3.1.14 Object 1029_h: Error Behavior . . . 28
3.1.15 Objects 1400_h to 1403_h: Receive PDO Communication Parameter . . . 28
3.1.16 Objects 1600_h to 1603_h: Receive PDO Mapping Parameter . . . 29
3.1.17 Objects 1800_h to 1803_h: Transmit PDO Communication Parameter . . . 30
3.1.18 Objects $1A00_h$ to $1A03_h$: Transmit PDO Mapping Parameter . . . 32

4 Manufacturer Specific Area **34**
4.1 Objects Related to CoolStep . . . 34
4.2 Detailed Object Specifications . . . 37
4.2.1 Object 2000_h: Microstep Resolution . . . 37
4.2.2 Object 2001_h: Fullstep Resolution . . . 37
4.2.3 Object 2002_h: Brake Delay Times . . . 37
4.2.4 Object 2003_h: Maximum Current . . . 38
4.2.5 Object 2004_h: Standby Current . . . 39
4.2.6 Object 2005_h: Switch Parameters . . . 39
4.2.7 Object $200A_h$: Enable Drive Delay Time . . . 40

4.2.8 Object $200B_h$: Encoder Parameters . . . 40
4.2.9 Object $200C_h$: Brake Current Feed . . . 41
4.2.10 Object 2010_h: Profile Start Velocity . . . 41
4.2.11 Object 2011_h: Profile A1 . . . 42
4.2.12 Object 2012_h: Profile V1 . . . 42
4.2.13 Object 2013_h: Profile D1 . . . 43
4.2.14 Object 2015_h: Ramp Wait Time . . . 43
4.2.15 Object 2016_h: Profile A2 . . . 43
4.2.16 Object 2017_h: Profile V2 . . . 44
4.2.17 Object 2018_h: Profile D2 . . . 44
4.2.18 Object 2089_h: Setting Delay . . . 45
4.2.19 Object $208C_h$: Velocity Dimension Index . . . 45
4.2.20 Object $208E_h$: Acceleration Dimension Index . . . 46
4.2.21 Object 2092_h: Chopper Blank Time . . . 46
4.2.22 Object 2093_h: Chopper Mode . . . 47
4.2.23 Object 2094_h: Chopper Disable Fast Decay Comparator . . . 47
4.2.24 Object 2095_h: Chopper Hysteresis End . . . 47
4.2.25 Object 2096_h: Chopper Hysteresis Start . . . 48
4.2.26 Object 2097_h: Chopper Off Time . . . 48
4.2.27 Object 2098_h: Smart Energy Current Minimum . . . 49
4.2.28 Object 2099_h: Smart Energy Current Down Step . . . 49
4.2.29 Object $209A_h$: Smart Energy Hysteresis . . . 50
4.2.30 Object $209B_h$: Smart Energy Current Up Step . . . 50
4.2.31 Object $209C_h$: Smart Energy Hysteresis Start . . . 51
4.2.32 Object $209D_h$: Smart Energy Filter Enable . . . 51
4.2.33 Object $209E_h$: StallGuard2 Threshold . . . 52
4.2.34 Object $20A1_h$: Short Protection Disable . . . 52
4.2.35 Object $20A2_h$: Global Current Scaler . . . 53
4.2.36 Object $20A3_h$: Driver Current Range . . . 53
4.2.37 Object $20A4_h$: Stop on Stall . . . 54
4.2.38 Object $20A5_h$: Smart Energy Threshold Speed . . . 54
4.2.39 Object $20A7_h$: Stop on Stall Option Code . . . 55
4.2.40 Object $20A8_h$: StallGuard4 Threshold Value . . . 55
4.2.41 Object $20A9_h$: StallGuard4 Filter Enable . . . 55
4.2.42 Object $20AA_h$: StallGuard4 Angle Offset . . . 56
4.2.43 Object $20B0_h$: PWM Threshold Speed . . . 56
4.2.44 Object $20B1_h$: PWM Gradient . . . 57
4.2.45 Object $20B2_h$: PWM Amplitude . . . 57
4.2.46 Object $20B3_h$: DcStep Minimum Speed . . . 58
4.2.47 Object $20B4_h$: DcStep Time . . . 58
4.2.48 Object $20B5_h$: DcStep StallGuard . . . 58
4.2.49 Object $20B6_h$: Fullstep Threshold Speed . . . 59
4.2.50 Object $20B7_h$: High-Speed Chopper Mode . . . 59
4.2.51 Object $20B8_h$: High-Speed Fullstep Mode . . . 60
4.2.52 Object $20B9_h$: Power Down Ramp . . . 60
4.2.53 Object 2100_h: Home Offset Display . . . 61
4.2.54 Object 2101_h: Actual Load Value . . . 61
4.2.55 Object 2102_h: Driver Error Flags . . . 61
4.2.56 Object 2107_h: Microstep Resolution Display . . . 62
4.2.57 Object $210A_h$: StallGuard4 Result . . . 63
4.2.58 Object 2121_h: PWM Scale Value . . . 63
4.2.59 Object 2122_h: Measured Velocity . . . 63
4.2.60 Object 2702_h: Device Digital Inputs . . . 64
4.2.61 Object 2703_h: Device Digital Outputs . . . 64

4.2.62 Object 2704_h: CAN Bit Rate . . . 65
4.2.63 Object 2705_h: Node ID . . . 66
4.2.64 Object 2706_h: Store . . . 67
4.2.65 Object 2707_h: CAN Bit Rate Load . . . 67
4.2.66 Object 2708_h: Node ID Load . . . 67
4.2.67 Object $270E_h$: Device Analog Inputs . . . 68
4.2.68 Object 2711_h: I/O Mode . . . 68

5 Profile Specific Area **70**
5.1 Detailed Object Specifications . . . 70
5.1.1 Object $605A_h$: Quick Stop Option Code . . . 70
5.1.2 Object $605B_h$: Shutdown Option Code . . . 71
5.1.3 Object $605C_h$: Disable Operation Option Code . . . 72
5.1.4 Object $605D_h$: Halt Option Code . . . 72
5.1.5 Object $605E_h$: Fault Reaction Option Code . . . 73
5.1.6 Object 6060_h: Modes of Operation . . . 73
5.1.7 Object 6061_h: Modes of Operation Display . . . 74
5.1.8 Object $606A_h$: Sensor Selection Code . . . 75
5.1.9 Object $608F_h$: Position Encoder Resolution . . . 76
5.1.10 Object $60FD_h$: Digital Inputs . . . 76
5.1.11 Object 6502_h: Supported Drive Modes . . . 76
5.1.12 Object $67FF_h$: Single Device Type . . . 77

6 Profile Position Mode **79**
6.1 Detailed Object Specifications . . . 79
6.1.1 Object 6040_h: Controlword . . . 80
6.1.2 Object 6041_h: Statusword . . . 81
6.1.3 Object 6062_h: Position Demand Value . . . 82
6.1.4 Object 6063_h: Position Actual Internal Value . . . 83
6.1.5 Object 6064_h: Position Actual Value . . . 83
6.1.6 Object 6065_h: Following Error Window . . . 84
6.1.7 Object 6067_h: Position Window . . . 84
6.1.8 Object 6068_h: Position Window Time . . . 85
6.1.9 Object $606C_h$: Velocity Actual Value . . . 85
6.1.10 Object $607A_h$: Target Position . . . 86
6.1.11 Object $607D_h$: Software Position Limit . . . 86
6.1.12 Object 6081_h: Profile Velocity . . . 87
6.1.13 Object 6082_h: End Velocity . . . 87
6.1.14 Object 6083_h: Profile Acceleration . . . 88
6.1.15 Object 6084_h: Profile Deceleration . . . 88
6.1.16 Object 6085_h: Quick Stop Deceleration . . . 88
6.1.17 Object $60F2_h$: Positioning Option Code . . . 89
6.2 How to Move a Motor in pp Mode . . . 90

7 Profile Velocity Mode **91**
7.1 Detailed Object Specifications . . . 91
7.1.1 Object 6040_h: Controlword . . . 91
7.1.2 Object 6041_h: Statusword . . . 92
7.1.3 Object 6062_h: Position Demand Value . . . 94
7.1.4 Object 6063_h: Position Actual Internal Value . . . 94
7.1.5 Object 6064_h: Position Actual Value . . . 95
7.1.6 Object 6065_h: Following Error Window . . . 95
7.1.7 Object $606C_h$: Velocity Actual Value . . . 96
7.1.8 Object $607D_h$: Software Position Limit . . . 96

7.1.9 Object 6083_h: Profile Acceleration . . . 97
7.1.10 Object 6085_h: Quick Stop Deceleration . . . 97
7.1.11 Object $60FF_h$: Target Velocity . . . 98
7.2 How to Move a Motor in pv Mode . . . 98

8 Homing Mode 99
8.1 Homing Methods . . . 100
8.1.1 Homing Method 1: Homing on negative Limit Switch and Index Pulse . . . 100
8.1.2 Homing Method 2: Homing on Positive Limit Switch and Index Pulse . . . 101
8.1.3 Homing Method 3: Homing on positive Home Switch and Index Pulse . . . 101
8.1.4 Homing Method 5: Homing on Negative Home Switch and Index Pulse . . . 101
8.1.5 Homing Method 17: Homing on Negative Limit Switch . . . 102
8.1.6 Homing Method 18: Homing on Positive Limit Switch . . . 102
8.1.7 Homing Method 19: Homing on Positive Home Switch . . . 103
8.1.8 Homing Method 21: Homing on Negative Home Switch . . . 103
8.1.9 Homing Method 33 and 34: Homing on Next Index Pulse . . . 103
8.1.10 Homing Method 35: Current Position as Home Position . . . 104
8.2 Detailed Object Specifications . . . 105
8.2.1 Object 6040_h: Controlword . . . 105
8.2.2 Object 6041_h: Statusword . . . 106
8.2.3 Object $606C_h$: Velocity Actual Value . . . 107
8.2.4 Object $607C_h$: Home Offset . . . 108
8.2.5 Object 6098_h: Homing Method . . . 109
8.2.6 Object 6099_h: Homing Speeds . . . 109
8.2.7 Object $609A_h$: Homing Acceleration . . . 109
8.3 How to Start a Homing in hm Mode . . . 110

9 Cyclic Synchronous Position Mode 111
9.1 Detailed Object Specifications . . . 111
9.1.1 Object 6040_h: Controlword . . . 111
9.1.2 Object 6041_h: Statusword . . . 112
9.1.3 Object 6062_h: Position Demand Value . . . 114
9.1.4 Object 6063_h: Position Actual Internal Value . . . 114
9.1.5 Object 6064_h: Position Actual Value . . . 115
9.1.6 Object $606C_h$: Velocity Actual Value . . . 115
9.1.7 Object $607A_h$: Target Position . . . 115
9.1.8 Object $607D_h$: Software Position Limit . . . 116
9.1.9 Object $60B0_h$: Position Offset . . . 116
9.1.10 Object $60C2_h$: Interpolation Time Period . . . 117

10 Cyclic Synchronous Velocity Mode 118
10.1 Detailed Object Specifications . . . 118
10.1.1 Object 6040_h: Controlword . . . 118
10.1.2 Object 6041_h: Statusword . . . 119
10.1.3 Object $606C_h$: Velocity Actual Value . . . 121
10.1.4 Object $60FF_h$: Target Velocity . . . 121
10.1.5 Object $607D_h$: Software Position Limit . . . 122
10.1.6 Object $60B1_h$: Velocity Offset . . . 122
10.1.7 Object $60C2_h$: Interpolation Time Period . . . 123

11 Emergency Messages 124
11.1 Format of Emergency Messages . . . 124
11.2 Error Codes Used by the TMCM-3216 Module . . . 124

12 SDO Abort Codes 127

13 Figures Index **129**

14 Tables Index **130**

15 Supplemental Directives **134**
15.1 Producer Information 134
15.2 Copyright 134
15.3 Trademark Designations and Symbols 134
15.4 Target User 134
15.5 Disclaimer: Life Support Systems 134
15.6 Disclaimer: Intended Use 134
15.7 Collateral Documents and Tools 135

16 Revision History **136**
16.1 Firmware Revision 136
16.2 Document Revision 136

1 Preface

This document specifies objects and modes of operation of the Trinamic TMCM-3216 stepper motor control module with CANopen firmware. The CANopen firmware is designed to fulfill the CANopen CiA-402 and CiA-301 standards. This manual assumes that the reader is already familiar with the basics of the CANopen protocol, defined by the CiA-301 and CiA-402 standards of the CAN-CiA.
If necessary it is always possible to turn the module into a TMCL module by loading the TMCM-3216 TMCL firmware again through the USB interface, with the help of the firmware update function of the TMCL-IDE (3.0 or higher).

1.1 General Features of this CANopen Implementation

Main Characteristics

- Communication according to standard CiA-301 V4.1
- CAN bit rate: 20kBit/s…1000kBit/s
- COB-ID: 11-bit
- Node ID: 1…127 (use vendor specific objects or LSS for changing the node ID)
- LSS server
- NMT server

SDO Communication

- One server
- Expedited transfer
- Segmented transfer
- No block transfer

PDO Communication

- Producer
- Consumer
- RPDOs
  - Axis 0: 1, 2, 3, 4
  - Axis 1: 65, 66, 67, 68
  - Axis 2: 129, 130, 131, 132
  - Transmission modes: asynchronous, synchronous
  - Dynamic mapping with max. three mapping entries
  - Default mappings: According to CiA-402 for first three PDOs of each axis, manufacturer specific for other PDOs of each axis
- TPDOs
  - Axis 0: 1, 2, 3, 4
  - Axis 1: 65, 66, 67, 68

  - Axis 2: 129, 130, 131, 132
  - Transmission modes: asynchronous, asynchronous with event timer, synchronous
  - Dynamic mapping with max. three mapping entries
  - Default mappings: according to CiA-402 for first three PDOs of each axis, manufacturer specific for other PDOs of each axis

Further Characteristics

- SYNC: Consumer (TPDOs and RPDOs can be configured as synchronous PDOs)
- Emergency: Producer
- RTR: Not supported
- Heartbeat: consumer and producer

1.2 Abbreviations Used in this Manual

Abbreviations	
CAN	Controller area network
CHGND	Chassis ground/earth ground
COB	Communication object
FSA	Finite state automaton
FSM	Finite state machine
NMT	Network management
ID	Identifier
LSB	Least significant bit
MSB	Most significant bit
PDO	Process data object
PDS	Power drive system
RPDO	Receive process data object
SDO	Service data object
TPDO	Transmit process data object
EMCY	Emergency object
rw	Read and write
ro	Read only
hm	Homing mode
pp	Profile position mode
pv	Profile velocity mode
vm	Velocity mode

Table 1: Abbreviations Used in this Manual

1.3 Firmware Update

The software running on the microprocessor consists of two parts, a bootloader and the CANopen firmware itself. Whereas the bootloader is installed during production and testing at TRINAMIC and remains untouched throughout the whole lifetime, the CANopen firmware can easily be updated. The new firmware can be loaded into the module through the firmware update function of the TMCL-IDE, using the CAN interface or the USB interface of the module.

1.4 Trinamic's Unique Features — Easy to Use with CANopen

1.4.1 StallGuard2

StallGuard2 is a high-precision sensorless load measurement using the back EMF of the coils. It can be used for stall detection as well as other uses at loads below those which stall the motor. The StallGuard2 measurement value changes linearly over a wide range of load, velocity, and current settings. At maximum motor load, the value reaches zero or is near zero. This is the most energy-efficient point of operation for the motor.

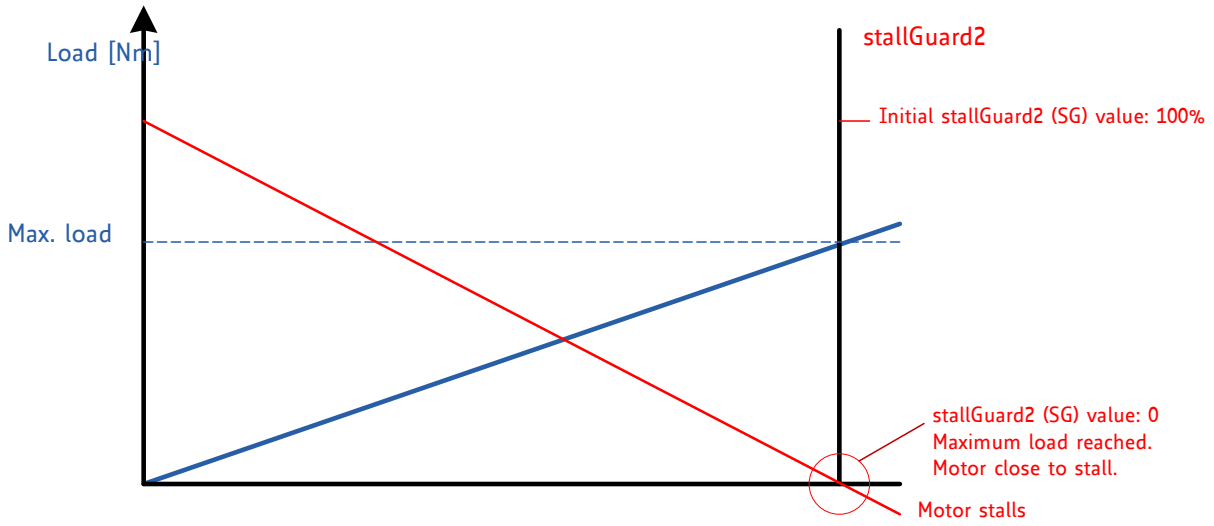


Figure 1: stallGuard2 Load Measurement as a Function of Load

1.4.2 CoolStep

CoolStep is a load-adaptive automatic current scaling based on the load measurement through StallGuard2 adapting the required current to the load. Energy consumption can be reduced by as much as 75%. CoolStep allows substantial energy savings, especially for motors which see varying loads or operate at a high duty cycle. As a stepper motor application needs to work with a torque reserve of 30% to 50%, even a constant-load application allows significant energy savings because CoolStep automatically enables torque reserve when required. Reducing power consumption keeps the system cooler, increases motor life, and allows cost reduction.

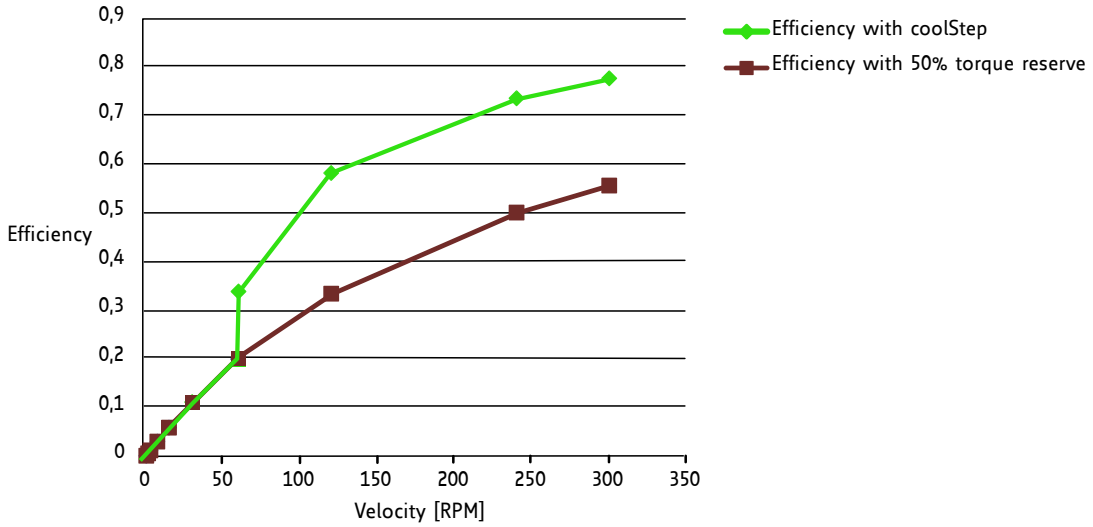


Figure 2: Energy Efficiency Example with CoolStep

1.4.3 StallGuard4

StallGuard4 also is a high-precision sensorless load measurement using the back EMF of the coils like StallGuard2 but optimized for use with StealthChop2. It can be used for stall detection. When the motor driver switches between SpreadCycle and StealthChop2 chopper scheme it also automatically switches between StallGuard2 and StallGuard4 load measurement. While StallGuard2 becomes tuned to show a zero reading for stall detection, StallGuard4 uses a comparison value to trigger stall detection, rather than shifting the measurement result by applying an offset.

1.5 EightPoint Motion Controller

TRINAMIC's EightPoint motion controller is a new type of ramp generator that offers faster machine operation compared to the classical linear acceleration ramps. It uses three different acceleration settings for the acceleration phase and also three different deceleration settings for the deceleration phase. Start and stop speeds greater than zero can also be used.

This set of three acceleration and deceleration segments can be used in two ways: either for adaptation to the motor torque curve, by using higher acceleration values at lower velocity, or to reduce the jerk (change of acceleration) when transitioning from one acceleration segment to the next. For jerk optimized ramps, typically, A1, D1, AMAX, and DMAX are set to lower values than A2 and D2. The most critical points with regards to jerk are the transition from acceleration to deceleration with no constant velocity segment, as well as the transition from deceleration to acceleration in case of on-the-fly change of target position. To address both, the EightPoint motion profile generator allows to enforce a constant velocity segment based on a minimum segment duration (TVMAX). In case this duration cannot be kept due to insufficient distance, VMAX is automatically reduced for the constant velocity segment.

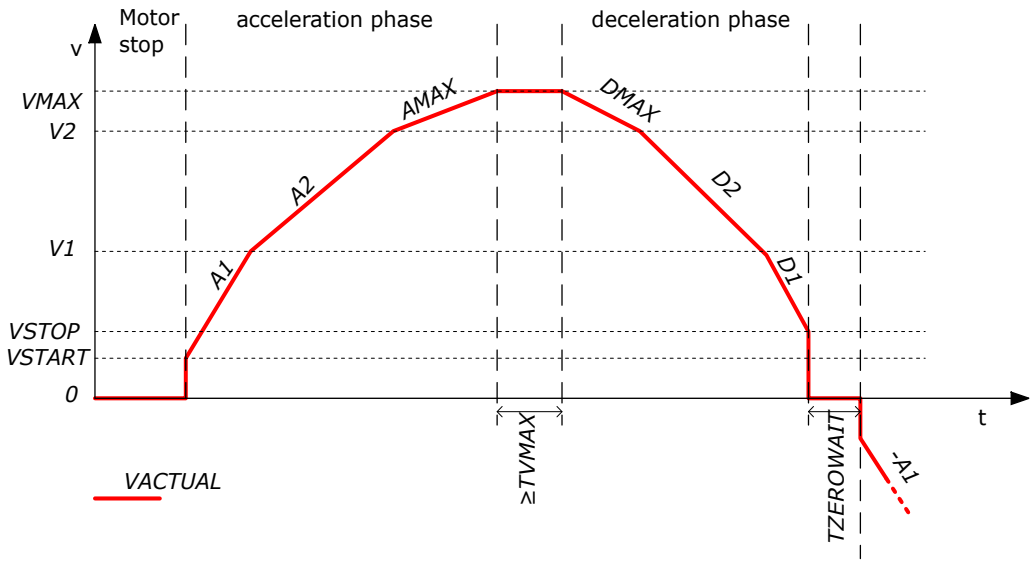


Figure 3: Typical Motion Profile with TRINAMIC EightPoint Motion Controller

An EightPoint ramp begins using the start speed VSTART (which can also be zero). Then, the acceleration value A1 is used to accelerate the motor to the speed V1. When the speed V1 is reached, the motor is accelerated to the speed V2 using acceleration A2. After reaching the speed V2, the motor is further accelerated using the acceleration value AMAX until it has reached the speed VMAX. The speed is then constant at least as long as set using TVMAX. The deceleration phase begins using the deceleration value DMAX. After reaching the speed V2 again, the deceleration value D2 is used to decelerate to the speed V1. After reaching the speed V1, the deceleration value D1 is used to decelerate to the stop speed VSTOP (which can also be zero). The speed is then zero at least as long as set by TZEROWAIT.

The EightPoint ramp can be configured using the following objects:

Parameter Name	Object Index
Start velocity (VSTART)	2010_h
Acceleration A1	2011_h
Velocity V1	2012_h
Acceleration A2	2016_h
Velocity V2	2017_h
Maximum acceleration (AMAX)	6083_h
Maximum positioning velocity (VMAX)	6081_h
Maximum deceleration (DMAX)	6084_h
Deceleration D1	2013_h
Deceleration D2	2018_h
Stop velocity (VSTOP)	6082_h
Ramp wait time (TZEROWAIT)	2015_h

Table 2: SixPoint Ramp Parameters

Setting the velocity V1 (object 2012_h) and the velocity V2 (object 2017_h) to zero switches off the EightPoint ramp. In this case, a trapezoidal ramp defined by the parameters VSTART, AMAX, VMAX, DMAX and VSTOP is used.

Note	The EightPoint ramp is only used in profile positioning mode (pp mode). Profile velocity mode (pv mode) always uses a trapezoidal ramp, defined just by the acceleration (object 6083_h), the speed given using object $60FF_h$ and the start and stop speed (objects 2010_h and 6082_h). The deceleration parameters are not used in pv mode.

Note	Do not set VSTOP lower than 10, and always set VSTOP higher than or equal to VSTART. Otherwise, the target position might not be reached. Do not set D1 and/or D2 to zero, even if V1 and V2 are set to zero.

2 Communication

2.1 Reference Model

The application layer comprises a concept to configure and communicate real-time-data as well as the mechanisms for synchronization between devices. The functionality that the application layer offers to an application is logically divided over different service data objects (SDO) in the application layer. A service object offers a specific functionality and all the related services.

Applications interact by invoking services of a service object in the application layer. To realize these services, this object exchanges data through the CAN network with peer service object(s) using a protocol.

The application and application layer interact with service primitives.

Service Primitives	
Primitive	Definition
Request	Issued by the application to the application layer to request a service.
Indication	Issued by the application layer to the application to report an internal event detected by the application layer or indicate that a service is requested.
Response	Issued by the application to the application layer to respond to a previous received indication.
Confirmation	Issued by the application layer to the application to report the result of a previously issued request.

Table 3: Service Primitives

A service type defines the primitives exchanged between the application layer and the cooperating applications for a particular service of a service object. Unconfirmed and confirmed services are collectively called remote services.

Service Types	
Type	Definition
Local service	Involves only the local service object. The application issues a request to its local service object that executes the requested service without communicating with peer service object(s).
Unconfirmed service	Involves one or more peer service objects. The application issues a request to its local service object. This request is transferred to the peer service object(s) that each passes it to their application as an indication. The result is not confirmed back.
Confirmed service	Can involve only one peer service object. The application issues a request to its local service object. This request is transferred to the peer service object that passes it to the other application as an indication. The other application issues a response that is transferred to the originating service object that passes it as a confirmation to the requesting application.
Provider initiated service	Involves only the local service object. The service object (being the service provider) detects an event not solicited by a requested service. This event is then indicated to the application.

Table 4: Service Types

2.2 NMT State Machine

The finite state machine (FSM) or simply state machine is a model of behavior composed of a finite number of states, transitions between those states, and actions. It shows which way the logic runs when certain conditions are met.

Starting and resetting the device is controlled through the state machine. The NMT state machine consists of the states shown in Figure 4.

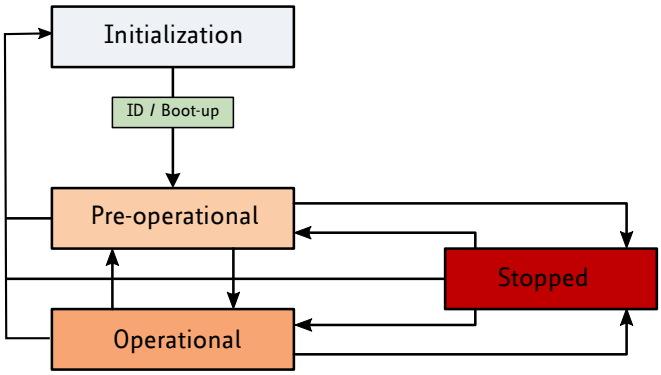


Figure 4: NMT State Machine

After power-on or reset, the device enters the initialization state. After the device initialization is finished, the device automatically transits to the **Pre-operational** state and indicates this state transition by sending the boot-up message. This way, the device indicates it is ready to work. A device that stays in pre-operational state may start to transmit SYNC-, time stamp- or heartbeat messages. In contrast to the PDO communication that is disabled in this state, the device can communicate through SDO.

The PDO communication is only possible within the **Operational** state. During operational state, the device can use all supported communication objects.

A device switched to the **Stopped** state only reacts on received NMT commands. In addition, the device indicates the current NMT state by supporting the error control protocol during stopped state.

The transitions between states are made by issuing a network management (NMT) communication object to the device. The NMT protocols are used to generate state machine change commands (example, to start and stop the device), and detect remote device boot-ups and error conditions.

The heartbeat message of a CANopen device contains the device status of the NMT state machine and is sent cyclically by the CANopen device.

The NMT state machine (or DS301 state machine) is not to be confused with the DS402 state machine. There is only one NMT state machine for the entire device, but for each motor there is a DS402 state machine that controls the motor. There are no links between these state machines, with one exception: when the NMT state machine is being switched to the stopped state, all DS402 state machines that are in OPERATION_ENABLED state switch to FAULT state.

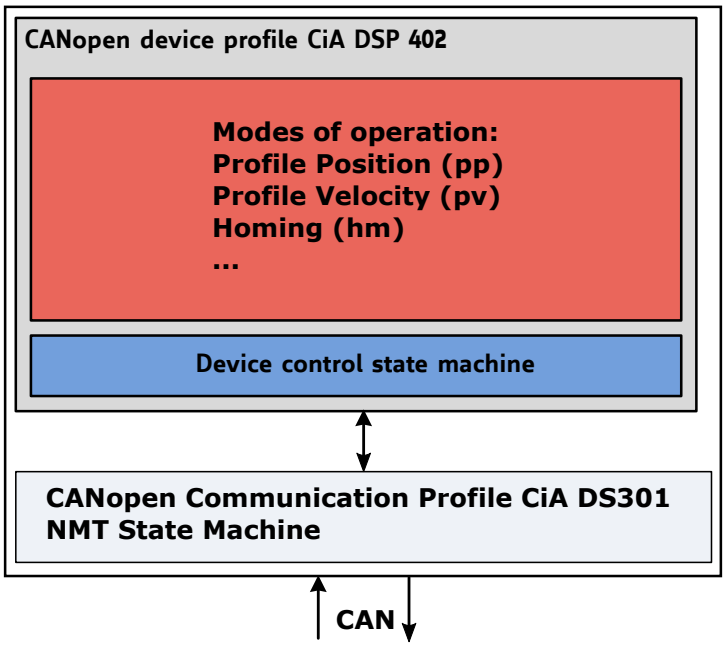


Figure 5: Communication Architecture

2.3 LSS State Machine

This version of the TMCM-3216 CANopen firmware also supports setting the node ID and the bitrate using LSS (Layer Setting Service) as defined in CiA-305. To use LSS, a CANopen manager or CANopen monitor tool is needed that supports this (for example, the TMCL-IDE or the TMCM-CANopen tool support this). The TMCL-IDE and the TMCM-CANopen tool can be downloaded from the Analog Devices website.
As an alternative the node ID and the CAN bitrate can also be set using objects 2704_h and 2705_h (for example if an LSS capable manager tool cannot be used). The factory default setting of the node ID is always 1 so that the module comes in LSS configured state. Thus an LSS capable manager tool is not necessarily needed.

2.4 Indicator LEDs

As defined by the CiA-303 standard part 3, one green and one red LED are used to indicate the NMT state and error state of the device.

2.4.1 Run LED

The green LED (also called CANopen run LED) shows the NMT state of the device:

- Flickering (50ms on, 50ms off): LSS services are in progress, as the node ID and/or bitrate have not been configured yet.
- Single flash (200ms on, 1000ms off): The device is in NMT Stopped state.
- Blinking (200ms on, 200ms off): The device is in NMT Pre-operational state.
- On: The device is in NMT Operational state.

Other run LED blink codes are not used by the TMCM-3216 module.

2.4.2 Error LED

The red LED (also called CANopen error LED) shows possible error conditions:

- Off: No error. The device is in working condition.
- Single Flash (200ms on, 1000ms off): Warning limit reached. Too many error frames on the CAN bus. Possible reasons: manager not yet initialized or wrong bit rate setting.
- Double Flash (200ms on, 200ms off, 200ms on, 1000ms off): Error control event. A node guard event or a heartbeat event (heartbeat consumer) has occured.
- On: Bus off. The CAN controller of the device is in bus off state.

Other error LED blink codes are not used by the TMCM-3216 module.

2.5 Device Model

A CANopen device consists of the following parts:

- *Communication:* This function unit provides the communication objects and the appropriate functionality to transport data items through the underlying network structure.
- *Object Dictionary:* The object dictionary is a collection of all the data items that have an influence on the behavior of the application objects, the communication objects and the state machine used on this device.
- *Application:* The application comprises the functionality of the device with respect to the interaction with the process environment.

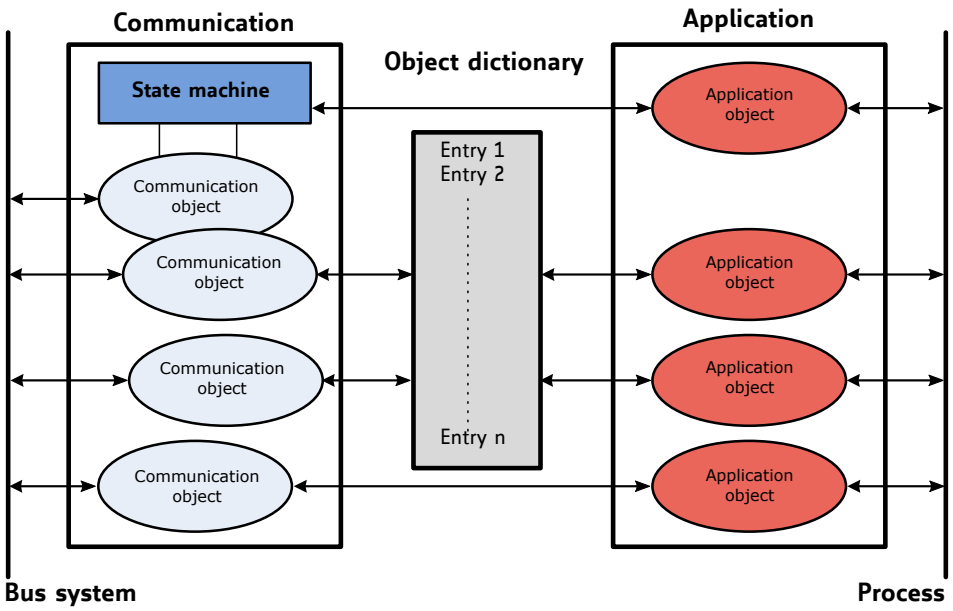


Figure 6: Device Model

2.6 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 through the network in an ordered predefined fashion. Each object within the dictionary is addressed using a 16-bit index. The overall layout of the standard object dictionary is shown in Table 5:

Object Dictionary	
Index	Object
0000_h	Not used
0001_h to $001F_h$	Static data types
0020_h to $003F_h$	Complex data types
0040_h to $005F_h$	Manufacturer specific complex data types
0060_h to $007F_h$	Device profile specific static data types
0080_h to $009F_h$	Device profile specific complex data types
$00A0_h$ to $0FFF_h$	Reserved for further use
1000_h to $1FFF_h$	Communication profile area
2000_h to $5FFF_h$	Manufacturer specific profile area
6000_h to $9FFF_h$	Standardized device profile area
$A000_h$ to $BFFF_h$	Standardized interface profile area
$C000_h$ to $FFFF_h$	Reserved for further use

Table 5: Object Dictionary

The communication profile area at indices 1000_h through $1FFF_h$ contains the communication specific parameters for the CAN network. These entries are common to all devices.

The manufacturer segment at indices 2000_h through $5FFF_h$ contains manufacturer specific objects. These objects control the special features of the Trinamic TMCM-3216 motion control device.

The standardized 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 through the network. They describe the device parameters and the device functionality of the device profile.

2.6.1 Object Indices on Multi-Axis Modules

On a multi-axis module like the TMCM-3216 each object in the manufacturer area and each object in the profile specific area is available for each motor. In this manual, only the object indices for motor #0 are shown. The objects for the other motors can be accessed by adding offsets to the object indices:

- Add an offset of $motor\_number \times 200_h$ to the index of a manufacturer specifc object to get its index for other motors.
- Add an offset of $motor\_number \times 800_h$ to the index of a profile specific object to get its index for other motors.

For example, the control word for motor #1 would be 6840_h (instead of 6040_h for motor #0), and the microstep resolution of motor #1 would be 2200_h for motor #1 (instead of 2000_h for motor #0).

Multi-Axis Object Indices		
Motor	Manufacturer area	Profile area
Motor #0	2000_h – $21FF_h$	6000_h – $67FF_h$
Motor #1	2200_h – $23FF_h$	6800_h – $6FFF_h$
Motor #2	2400_h – $25FF_h$	7000_h – $77FF_h$

Table 6: Multi-Axis Object Indices

3 Communication Area

The communication area contains all objects that define the communication parameters of the CANopen device according to the DS301 standard.

3.1 Detailed Object Specifications

3.1.1 Object 1000_h: Device Type

This object contains information about the device type. The object 1000_h describes the type of device and its functionality. It is composed of a 16-bit field that describes the device profile used and a second 16-bit field that provides additional information about optional functionality of the device.

Object Description			
Index	Name	Object Type	Data Type
1000_h	Device Type	Variable	UNSIGNED32

Table 7: Object Description (1000_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	UNSIGNED32	$FFFC0192_h$

Table 8: Entry Description (1000_h)

3.1.2 Object 1001_h: Error Register

This object contains error information. The CANopen device maps internal errors into object 1001_h. It is part of an emergency object.

Object Description			
Index	Name	Object Type	Data Type
1001_h	Error register	Variable	UNSIGNED8

Table 9: Object Description (1001_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	UNSIGNED8	0

Table 10: Entry Description (1001_h)

Error Register Bits	
Bit	Definition
0	Generic error
1	Current
2	Voltage
3	Temperature
4	Communication error
5	Device profile specific
6	Reserved (always 0)
7	Manufacturer specific

Table 11: Error Register Bits

3.1.3 Object 1005_h: COB-ID SYNC

This object defines the COB-ID of the synchronization object (SYNC). Further, it defines whether the module generates the SYNC.

Value Definition		
Bit	Name	Definition
30	Generate	0: Device does not generate SYNC message 1: Device generates SYNC message
29	Frame	Not supported, always set to 0.
28...11	29-bit ID	Not supported, always set to 0.
10...0	11-bit ID	11-bit COB-ID.

Table 12: Value Definition (1005_h)

Object Description			
Index	Name	Object Type	Data Type
1005_h	COB-ID SYNC	Variable	UNSIGNED32

Table 13: Object Description (1005_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	UNSIGNED32	80_h

Table 14: Entry Description (1005_h)

3.1.4 Object 1008_h: Manufacturer Device Name

This object contains the name of the device as given by the manufacturer.

Object Description			
Index	Name	Object Type	Data Type
1008_h	Manufacturer Device Name	Variable	Visible String

Table 15: Object Description (1008_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	—	TMCM-3216

Table 16: Entry Description (1008_h)

3.1.5 Object 1009_h: Manufacturer Hardware Version

This object contains the hardware version description.

Object Description			
Index	Name	Object Type	Data Type
1009_h	Manufacturer Hardware Version	Variable	Visible String

Table 17: Object Description (1009_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	—	Depends on device. Example: 1.00

Table 18: Entry Description (1009_h)

3.1.6 Object $100A_h$: Manufacturer Software Version

This object contains the software version description.

Object Description			
Index	Name	Object Type	Data Type
$100A_h$	Manufacturer Software Version	Variable	Visible String

Table 19: Object Description ($100A_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	—	Depends on device. Example: 1.00

Table 20: Entry Description ($100A_h$)

3.1.7 Object 1010_h: Store Parameters

This object supports the saving of parameters in non-volatile memory. By read access, the device provides information about its saving capabilities.

The TMCM-3216 module supports saving of the following parameter groups:

- Sub-index 1: save all parameters.
- Sub-index 2: save communication parameters 2704_h and 2705_h.
- Sub-index 3: save device profile parameters (not used).
- Sub-index 4: save motor 0 parameters.
- Sub-index 5: save motor 1 parameters.
- Sub-index 6: save motor 2 parameters.

Note To avoid storage of parameters by mistake, storage is only executed when a specific signature is written to the appropriate sub-Index. This signature is "save" (65766173_h, see also Table 21).

Save Signature			
e	v	a	s
65_h	76_h	61_h	73_h

Table 21: Save Signature

On reception of the correct signature in the appropriate sub-index, the device stores the parameter and then confirms the SDO transmission (initiate download response). If the storing failed, the device responds with an abort SDO transfer (abort code: 06060000_h). If a wrong signature is written, the device refuses to store and responds with abort SDO transfer (abort code: $0800002x_h$).

On read access, each sub-index provides information if it is possible to store the parameter group. It reads 1 if yes and 0 if no.

Object Description			
Index	Name	Object Type	Data Type
1010_h	Store Parameters	Array	UNSIGNED32

Table 22: Object Description (1010_h)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
0	Highest supported sub-index	ro	no	UNSIGNED8	6
1	Save all parameters	rw	no	UNSIGNED32	1
2	Save communication parameters	rw	no	UNSIGNED32	1
3	Save device profile parameters	rw	no	UNSIGNED32	0
4	Save motor 0 parameters	rw	no	UNSIGNED32	1
5	Save motor 1 parameters	rw	no	UNSIGNED32	1
6	Save motor 2 parameters	rw	no	UNSIGNED32	1

Table 23: Entry Description (1010_h)

3.1.8 Object 1011_h: Restore Parameters

With this object, the default values of parameters according to the communication or device profile are restored. By read access, the device provides information about its capabilities to restore these values.

The TMCM-3216 module supports restoring of the following parameter groups:

- Sub-index 1: restore all parameters.
- Sub-index 2: no function.
- Sub-index 3: restore device profile parameters (not used).
- Sub-index 4: restore motor 0 parameters.
- Sub-index 5: restore motor 1 parameters.
- Sub-index 6: restore motor 2 parameters.

Note To avoid restoring the parameters by mistake, restoring is only executed when a specific signature is written to the appropriate sub-Index. This signature is "load" ($64616F6C_h$, see also Table 24).

Load Signature			
d	a	o	l
64_h	61_h	$6F_h$	$6C_h$

Table 24: Load Signature

On reception of the correct signature in the appropriate sub-index, the device restores the parameter and then confirms the SDO transmission (initiate download response). If the restoring failed, the device responds with an abort SDO transfer (abort code: 06060000_h). If a wrong signature is written, the device refuses to restore and responds with abort SDO transfer (abort code: $0800002x_h$).

On read access, each sub-index provides information if it is possible to restore the parameter group. It reads 1 if yes and 0 if no.

After the default values are restored, they become active after the next reset or power cycle of the TMCM-3216.

Object Description			
Index	Name	Object Type	Data Type
1011_h	Restore Parameters	Array	UNSIGNED32

Table 25: Object Description (1011_h)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
0	Highest supported sub-index	ro	no	UNSIGNED8	6
1	Restore all parameters	rw	no	UNSIGNED32	1
2	Restore communication parameters	rw	no	UNSIGNED32	1
3	Restore device profile parameters	rw	no	UNSIGNED32	0
4	Restore motor 0 parameters	rw	no	UNSIGNED32	1
5	Restore motor 1 parameters	rw	no	UNSIGNED32	1
6	Restore motor 2 parameters	rw	no	UNSIGNED32	1

Table 26: Entry Description (1011_h)

3.1.9 Object 1014_h: COB-ID Emergency Object

This object defines the COB-ID of the emergency object (EMCY).

Object Description			
Index	Name	Object Type	Data Type
1014_h	COB-ID Emergency Object	Variable	UNSIGNED32

Table 27: Object Description (1014_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	UNSIGNED32	80_h + Node ID

Table 28: Entry Description (1014_h)

3.1.10 Object 1015_h: Inhibit Time EMCY

The inhibit time for the EMCY message can be adjusted through this entry. The time has to be a multiple of 100µs.

Object Description			
Index	Name	Object Type	Data Type
1015_h	Inhibit Time EMCY	Variable	UNSIGNED16

Table 29: Object Description (1015_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	UNSIGNED16	0

Table 30: Entry Description (1015_h)

3.1.11 Object 1016_h: Consumer Heartbeat Time

The consumer heartbeat time defines the expected heartbeat cycle time and thus has to be higher than the corresponding producer heartbeat time configured on the module producing this heartbeat. The monitoring starts after the reception of the first heartbeat. If the consumer heartbeat time is 0, the corresponding entry is not used. The time has to be a multiple of 1ms.

Value Definition		
Bits	Name	Definition
31…24	Reserved	—
23…16	Node ID	Heartbeat Producer Node ID
15…0	Heartbeat time	Time in 1ms

Table 31: Value Definition (1016_h)

Object Description			
Index	Name	Object Type	Data Type
1016_h	Consumer Heartbeat Time	Array	UNSIGNED32

Table 32: Object Description (1016_h)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
0	Number of entries	ro	no	UNSIGNED8	1
1	Consumer Heartbeat Time1	rw	no	UNSIGNED32	0

Table 33: Entry Description (1016_h)

3.1.12 Object 1017_h: Producer Heartbeat Time

The producer heartbeat time defines the cycle time of the heartbeat. The producer heartbeat time is 0 if it is not used. The time has to be a multiple of 1ms.

Object Description			
Index	Name	Object Type	Data Type
1017_h	Producer Heartbeat Time	Variable	UNSIGNED16

Table 34: Object Description (1017_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	UNSIGNED16	0

Table 35: Entry Description (1017_h)

3.1.13 Object 1018_h: Identity Object

The object 1018_h contains general information about the device:

- The vendor ID (sub-index 01_h) contains a unique value allocated to each manufacturer. The vendor ID of Trinamic is 286_h.
- The manufacturer specific product code (sub-index 2_h) identifies a specific device version.
- The manufacturer specific revision number (sub-index 3_h) consists of a major revision number and a minor revision number.

Object Description			
Index	Name	Object Type	Data Type
1018_h	Identity Object	Record	Identity

Table 36: Object Description (1018_h)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
0	Number of entries	ro	no	0…3	3
1	Vendor ID	ro	no	UNSIGNED32	0286_h
2	Product code	ro	no	UNSIGNED32	3216
3	Revision number	ro	no	UNSIGNED32	e.g. 20003_h for version 2.3
4	Serial number	ro	no	UNSIGNED32	depends on module

Table 37: Entry Description (1018_h)

3.1.14 Object 1029_h: Error Behavior

If a device failure is detected in operational state, the device can be configured to alternatively enter the stopped state or remain in the current state in case of a device failure. Device failures include the following errors:

- Communication error
- Application error

Object Description			
Index	Name	Object Type	Data Type
1029_h	Error Behavior	Array	UNSIGNED8

Table 38: Object Description (1029_h)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
0	Number of error classes	ro	no	—	2
1	Communication Error	rw	no	UNSIGNED8	0 (enter stopped state)
2	Application Error	rw	no	UNSIGNED8	1 (remain in current state)

Table 39: Entry Description (1029_h)

3.1.15 Objects 1400_h to 1403_h: Receive PDO Communication Parameter

This object contains the communication parameters for the RPDOs the device is able to receive. The sub-index 0 contains the number of valid entries within the communication record. Its value normally is 2, as this object consists of two other entries.

Sub-index 1 contains the COB-ID used by this PDO (in bits 10…0). Bit 30 (RTR bit) defines if this PDO uses RTRs. As RTRs are not supported for PDOs by this CANopen implementation, this bit must always be set to turn off RTR support for this PDO. Bit 31 defines if this PDO is active or not. If this bit is set, the PDO is inactive, and if this bit is clear, the PDO is active. Before making any changes to a PDO definition, set this

bit to inactivate the PDO.

Sub-index 2 contains the transmission type of the RPDO. This can be FF_h or FE_h for event-driven, or 1…240 for synchronous (1 means that the PDO is processed with every SYNC message, and 4, for example, means that the PDO is processed with every fourth SYNC message). Other values are not supported.

The RPDOs defined by objects 1400_h to 1403_h (RPDO 1…3) are normally used for motor #0. For the other motors, the following RPDOs are used:

- RPDO 65…68 (objects 1440_h – 1443_h) for motor #1.
- RPDO 129…132 (objects 1480_h – 1483_h) for motor #2.

Object Description			
Index	Name	Object Type	Data Type
1400_h to 1403_h	Receive PDO Communication Parameter	RECORD	RPDO CommPar
1400_h	RPDO 1	RECORD	RPDO CommPar
1401_h	RPDO 2	RECORD	RPDO CommPar
1402_h	RPDO 3	RECORD	RPDO CommPar
1403_h	RPDO 4	RECORD	RPDO CommPar

Table 40: Object Description (1400_h)

Entry Description				
Sub-index	Description	Access	Value Range	Default Value
0	Largest sub-index supported	ro	2	2
1	COB-ID used by PDO	rw	UNSIGNED32	Index 1400_h: 200_h + Node-ID Index 1401_h: 300_h + Node-ID Index 1402_h: 400_h + Node-ID Index 1403_h: 500_h + Node-ID
2	Transmission Type	rw	UNSIGNED8	Index 1400_h: FF_h Index 1401_h: FF_h Index 1402_h: FF_h Index 1403_h: FE_h

Table 41: Entry Description (1400_h)

3.1.16 Objects 1600_h to 1603_h: Receive PDO Mapping Parameter

These objects contain the mapping parameters for the RPDOs the device is able to receive. The sub-index 0 contains the number of valid entries within the mapping record. This number of entries is also the number of the application variables that are received with the corresponding RPDO. The sub-indices from 1 to the number of entries contain the information about the mapped application variables. These entries describe the PDO contents by their index, sub-index, and length.

The RPDOs defined by objects 1600_h to 1603_h (RPDO 1…3) are normally used for motor #0. For the other motors, the following RPDOs are used:

- RPDO 65…68 (objects 1640_h – 1643_h) for motor #1.
- RPDO 129…132 (objects 1680_h – 1683_h) for motor #2.

Object Description			
Index	Name	Object Type	Data Type
1600_h to 1603_h	Receive PDO Mapping Parameter	RECORD	PDO Mapping
1600_h	RPDO 1	RECORD	PDO Mapping
1601_h	RPDO 2	RECORD	PDO Mapping
1602_h	RPDO 3	RECORD	PDO Mapping
1603_h	RPDO 4	RECORD	PDO Mapping

Table 42: Object Description (1600_h)

Entry Description				
Sub-index	Description	Access	Value Range	Default Value
0	Number of mapped application objects in PDO	rw	0…3	Index 1600_h: 1 Index 1601_h: 2 Index 1602_h: 2 Index 1603_h: 2
1	Mapping Entry 1	rw	UNSIGNED32	Index 1600_h: 60400010_h Index 1601_h: 60400010_h Index 1602_h: 60400010_h Index 1603_h: 60400010_h
2	Mapping Entry 2	rw	UNSIGNED32	Index 1600_h: 0 Index 1601_h: 60600008_h Index 1602_h: $607A0020_h$ Index 1603_h: $60FF0020_h$
3	Mapping Entry 3	rw	UNSIGNED32	Index 1600_h: 0_h Index 1601_h: 0_h Index 1602_h: 0_h Index 1603_h: 0_h

Table 43: Entry Description (1600_h)

Before making changes to PDO definitions, first mark the PDO as inactive by setting bit 31 of its COB-ID (see Section 3.1.15). Then, set its number of mapped PDO entries to zero (sub-index 0 of the appropriate PDO mapping object). Now, the mapppings themselves can be changed. After that, set the number of map objects to the desired value, and finally activate the PDO by clearing bit 31 of its COB-ID.

3.1.17 Objects 1800_h to 1803_h: Transmit PDO Communication Parameter

This object contains the communication parameters for the TPDOs the device is able to transmit. The sub-index 0 contains the number of valid entries within the communication record. Its value normally is

5, as this object consists of five other entries.

Sub-index 1 contains the COB-ID used by this PDO (in bits 10...0). Bit 30 (RTR bit) defines if this PDO uses RTRs. As RTRs are not supported for PDOs by this CANopen implementation, this bit must always be set to turn off RTR support for this PDO. Bit 31 defines if this PDO is active or not. If this bit is set, the PDO is inactive, and if this bit is clear, the PDO is active. Before making any changes to a PDO definition, set this bit to inactivate the PDO.

Sub-index 2 contains the transmission type of the RPDO. This can be FF_h or FE_h for event-driven or 1...240 for synchronous (1 means that the PDO is sent with every SYNC message, and 4, for example, means that the PDO is sent with every fourth SYNC message). Other values are not supported.

Sub-index 3 contains the inhibit time, given in units of 0.1ms. After a TPDO is sent, it is not sent again before the inhibit time has elapsed.

Sub-index 4 is not used.

Sub-index 5 contains the event timer value in milliseconds. When this is set to a value greater than 0, the TPDO is sent repeatedly each time the event timer has elapsed. It is also sent when the value has changed before the event timer has elapsed, but not before the inhibit time has elapsed.

The TPDOs defined by objects 1800_h to 1803_h (TPDO 1...3) are normally used for motor #0. For the other motors, the following TPDOs are used:

- TPDO 65...68 (objects 1840_h to 1843_h) for motor #1.
- TPDO 129...132 (objects 1880_h to 1883_h) for motor #2.

Object Description			
Index	Name	Object Type	Data Type
1800_h to 1803_h	Transmit PDO Communication Parameter	RECORD	TPDO CommPar
1800_h	TPDO 1	RECORD	TPDO CommPar
1801_h	TPDO 2	RECORD	TPDO CommPar
1802_h	TPDO 3	RECORD	TPDO CommPar
1803_h	TPDO 4	RECORD	TPDO CommPar

Table 44: Object Description (1800_h)

Entry Description				
Sub-index	Description	Access	Value Range	Default Value
0	Largest sub-index supported	ro	5	5
1	COB-ID	rw	UNSIGNED32	Index 1800_h: 180_h + Node-ID Index 1801_h: 280_h + Node-ID Index 1802_h: 380_h + Node-ID Index 1803_h: 480_h + Node-ID
2	Transmission Type	rw	UNSIGNED8	Index 1800_h: FF_h Index 1801_h: FF_h Index 1802_h: 01_h Index 1803_h: 01_h
3	Inhibit Time	rw	UNSIGNED16	0
4	Compatibility Entry	ro	UNSIGNED8	0
5	Event Timer	rw	UNSIGNED16	0

Table 45: Entry Description (1800_h)

3.1.18 Objects $1A00_h$ to $1A03_h$: Transmit PDO Mapping Parameter

These objects contain the mapping parameters for the TPDOs the device is able to transmit. The sub-index 0 contains the number of valid entries within the mapping record. This number of entries is also the number of the application variables that are transmitted with the corresponding TPDO. The sub-indices from 1 to the number of entries contain the information about the mapped application variables. These entries describe the PDO contents by their index, sub-index, and length.

The TPDOs defined by objects $1A00_h$ to $1A03_h$ (TPDO 1…3) are normally used for motor #0. For the other motors, the following TPDOs are used:

- TPDO 65…68 (objects $1A40_h$ to $1A43_h$) for motor #1.
- TPDO 129…132 (objects $1A80_h$ to $1A83_h$) for motor #2.

Object Description			
Index	Name	Object Type	Data Type
$1A00_h$ to $1A03_h$	Transmit PDO Mapping Parameter	RECORD	PDO Mapping
$1A00_h$	TPDO 1	RECORD	PDO Mapping
$1A01_h$	TPDO 2	RECORD	PDO Mapping
$1A02_h$	TPDO 3	RECORD	PDO Mapping
$1A03_h$	TPDO 4	RECORD	PDO Mapping

Table 46: Object Description ($1A00_h$)

Entry Description				
Sub-index	Description	Access	Value Range	Default Value
0	Number of mapped application objects in PDO	rw	0...3	Index $1A00_h$: 1 Index $1A01_h$: 2 Index $1A02_h$: 2 Index $1A03_h$: 2
1	Mapping Entry 1	rw	UNSIGNED32	Index $1A00_h$: 60410010_h Index $1A01_h$: 60410010_h Index $1A02_h$: 60410010_h Index $1A03_h$: 60410010_h
2	Mapping Entry 2	rw	UNSIGNED32	Index $1A00_h$: 0 Index $1A01_h$: 60610008_h Index $1A02_h$: 60640020_h Index $1A03_h$: $606C0020_h$
3	Mapping Entry 3	rw	UNSIGNED32	Index $1A00_h$: 0_h Index $1A01_h$: 0_h Index $1A02_h$: 0_h Index $1A03_h$: 0_h

Table 47: Entry Description ($1A00_h$)

Before making changes to PDO definitions, first mark the PDO as inactive by setting bit 31 of its COB-ID (see Section 3.1.17). Then, set its number of mapped PDO entries to zero (sub-index 0 of the appropriate PDO mapping object). Now, the mappings themselves can be changed. After that, set the number of map objects to the desired value, and finally activate the PDO by clearing bit 31 of its COB-ID.

4 Manufacturer Specific Area

The manufacturer segment contains manufacturer specific objects. These objects control the special features of the Trinamic Motion Control device TMCM-3216.

Info

This section of the manual only shows the object indices for motor #0. Of course the same objects are also available for the other motors. For the other motors, add an offset of $motor\_number \times 200_h$ to the object index. So for example the microstep resolution (object 2000_h for motor #0) can be accessed as object 2200_h for motor #1 and as object 2400_h for motor #2. See also Section 2.6.1.

Multi-Axis Object Indices	
Motor	Object Index Range
Motor #0	2000_h – $21FF_h$
Motor #1	2200_h – $23FF_h$
Motor #2	2400_h – $25FF_h$

Table 48: Multi-Axis Object Indices (Manufacturer Specific Area)

4.1 Objects Related to CoolStep

Figure 7 shows an overview of the CoolStep related objects for motor #0. Bear in mind that the figure only shows one example for a drive. There are objects which concern the configuration of the current. Other objects are for velocity regulation and for time adjustment. The CoolStep feature is sometimes also called SmartEnergy.

The following adjustments have to be made:

- Thresholds for current and velocity have to be identified and set.
- The StallGuard2 feature has to be adjusted and enabled.
- The reduction or increasing of the current in the CoolStep area (depending on the load) has to be configured.

CoolStep™ Adjustment Points and Thresholds

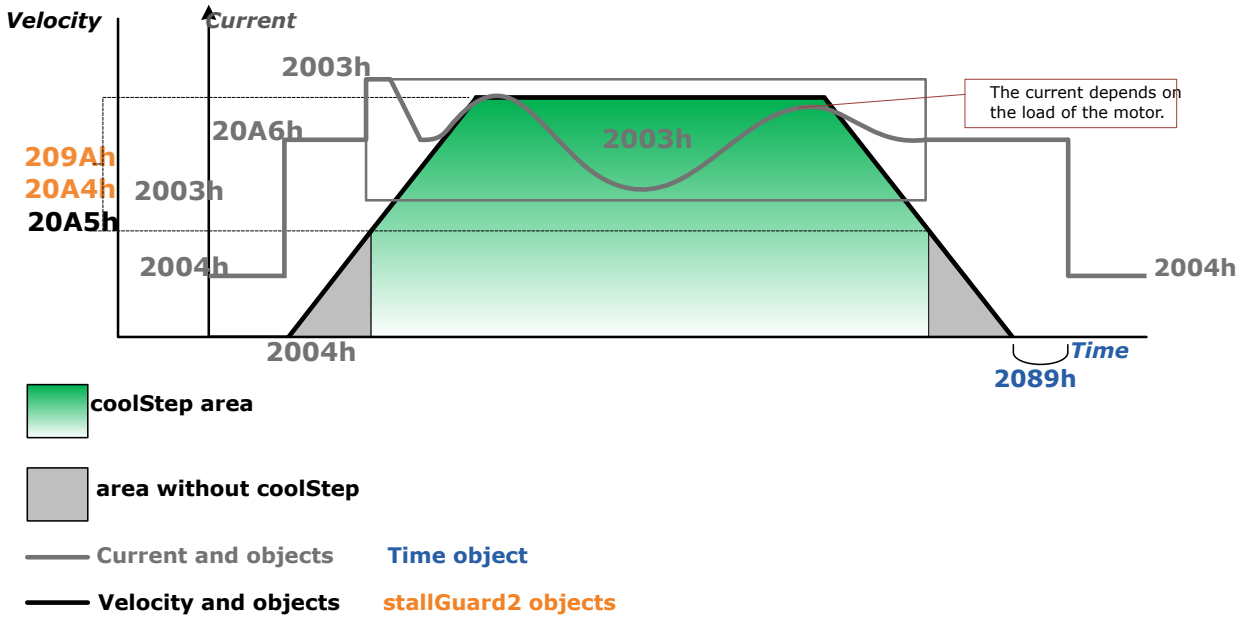


Figure 7: CoolStep Adjustment Points and Thresholds

CoolStep Adjustment Objects		
Object	Name	Description
2003_h	Absolute maximum current	The maximum value is 255. This value means 100% of the maximum current of the module. The current adjustment is within the range 0…255 and can be adjusted in 32 steps (0…255 divided by eight; step 0 = 0…7, step 1 = 8…15 and so on). The most important motor setting, since too high values might cause motor damage!
2004_h	Standby current	The current limit two seconds after the motor has stopped.
2098_h	SmartEnergy current minimum	Sets the lower motor current limit for CoolStep operation by scaling the run current (object 2003_h) value. This can be: 0: for 1/2 of the run current 1: for 1/4 of the run current
2099_h	SmartEnergy current down step	Sets the speed of current decrement when the StallGuard2 reading is above the upper threshold. 0: slow decrement 3: fast decrement
$209B_h$	SmartEnergy current up step	Sets the current increment step when the StallGuard2 below the lower threshold. 0: slow increment 3: fast increment / fast reaction to rising load
$209A_h$	SmartEnergy hysteresis	Sets the distance between the lower and the upper threshold for StallGuard2 reading. Above the upper threshold the motor current becomes decreased.
$20A4_h$	Stop on stall	Below this speed the motor is not stopped. Above this speed the motor stops in case StallGuard2 load value reaches zero.
$20A5_h$	SmartEnergy threshold speed.	Above this speed CoolStep becomes enabled.
2089_h	Standby delay	Standstill period before the current is changed down to standby current. The standard value is 200 which is 2 seconds.

Table 49: CoolStep-Related Objects

4.2 Detailed Object Specifications

4.2.1 Object 2000_h: Microstep Resolution

This object sets the microstep resolution of the drive. A value of 8 selects 256 (2^8) microsteps per full step.

Object Description			
Index	Name	Object Type	Data Type
2000_h	Microstep Resolution	Variable	UNSIGNED8

Table 50: Object Description (2000_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0…8	8

Table 51: Entry Description (2000_h)

4.2.2 Object 2001_h: Fullstep Resolution

This object sets the fullstep resolution of the motor connected to the drive. Its default value is 200 because most motors are 1.8° motors.

Object Description			
Index	Name	Object Type	Data Type
2001_h	Motor full step resolution	Variable	UNSIGNED16

Table 52: Object Description (2001_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0…65535	200

Table 53: Entry Description (2001_h)

4.2.3 Object 2002_h: Brake Delay Times

With this object the delay times for applying and releasing an (optional) brake can be defined. See also object $200A_h$ for an additional delay between enabling the power stage and releasing the brake. Both times are given in ms.

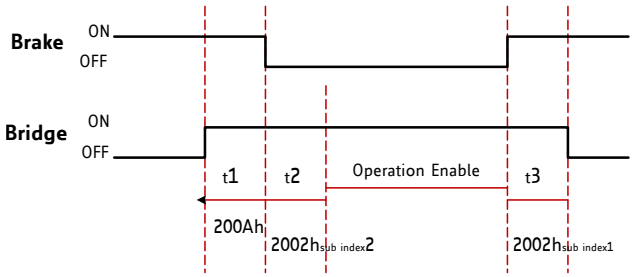


Figure 8: Brake Output Timing

Object Description			
Index	Name	Object Type	Data Type
2002_h	Brake delay times	Array	UNSIGNED16

Table 54: Object Description (2002_h)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
1	Time between applying brake / disabling power stage	rw	no	0…65535	0
2	Time between releasing brake / switching the state machine to operational	rw	no	0..65535	0

Table 55: Entry Description (2002_h)

4.2.4 Object 2003_h: Maximum Current

This object defines the current used when the motor is moving. A value of 255 means 100% of the maximum current of the drive.

Object Description			
Index	Name	Object Type	Data Type
2003_h	Maximum current	Variable	UNSIGNED8

Table 56: Object Description (2003_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0…255	128

Table 57: Entry Description (2003_h)

4.2.5 Object 2004_h: Standby Current

This object defines the current used when the motor is standing (two seconds after the last move). A value of 255 means 100% of the maximum current of the drive.

Object Description			
Index	Name	Object Type	Data Type
2004_h	Maximum current	Variable	UNSIGNED8

Table 58: Object Description (2004_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0…255	8

Table 59: Entry Description (2004_h)

4.2.6 Object 2005_h: Switch Parameters

This object defines which limit switches are to be used. Bit 0 stands for the left and bit 1 stands for the right limit switch. If a bit is set, the corresponding limit switch is not used. So this object has to be set to the value 3 if limit switches are not connected. The object can only be written when the drive is in the SWITCHED_ON_DISABLED state (but is always readable).

The limit switches can also be inverted using bit 2 and bit 3:

- Bit 2 inverts the left limit switch
- Bit 3 inverts the right limit switch

The polarity of the home switch can be set using bit 5.

Object Description			
Index	Name	Object Type	Data Type
2005_h	Switch Parameters	Variable	UNSIGNED32

Table 60: Object Description (2005_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0…63	0

Table 61: Entry Description (2005_h)

Bit Definitions	
Bit	Definition
0	Left limit switch deactivated if set.
1	Right limit switch deactivated if set.
2	Left limit switch inverted if set.
3	Right limit switch inverted if set.
4	Home switch deactivated if set.
5	Home switch inverted if set.

Table 62: Bit Definitions (2005_h)

4.2.7 Object $200A_h$: Enable Drive Delay Time

This is an additional delay time (in milliseconds) between enabling the power stage and releasing the brake. It can be used to prevent the brake from being released too early (before the hold current in the motor has been reached). See also object 2002_h.

Object Description			
Index	Name	Object Type	Data Type
$200A_h$	Enable drive delay time	Variable	UNSIGNED16

Table 63: Object Description ($200A_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...65535	0

Table 64: Entry Description $(200A)_h$

4.2.8 Object $200B_h$: Encoder Parameters

This object defines encoder parameters. These are the polarity of the encoder null channel, the direction of rotaion (set it to 1 if the direction is reversed compared to the motor) and if the position is to be intialized with the encoder position. It is only writable in SWITCHED_ON_DISABLED state.

Object Description			
Index	Name	Object Type	Data Type
$200B_h$	Encoder parameters	Array	UNSIGNED8

Table 65: Object Description ($200B_h$)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
1	Null channel polarity	rw	no	0/1	0
2	Direction of rotation	rw	no	0/1	0
3	Initialize position	rw	no	0/1	1

Table 66: Entry Description ($200B_h$)

4.2.9 Object $200C_h$: Brake Current Feed

This object configures how much current has to be fed into the brake to apply and to release it. Zero means 0%, 255 means 100% of the maximum current (this depends on the module). In most cases it is needed to feed current into the brake to release it. Setting both values to 0 disables the automatic brake control. This object is only writable in SWITCHED_ON_DISABLED state.

Object Description			
Index	Name	Object Type	Data Type
$200C_h$	Brake current feed	Array	UNSIGNED8

Table 67: Object Description ($200C_h$)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
1	Apply current	rw	no	0…255	0
2	Release current	rw	no	0…255	0

Table 68: Entry Description ($200C_h$)

4.2.10 Object 2010_h: Profile Start Velocity

This object contains the velocity with which a positioning ramp is started.

Object Description			
Index	Name	Object Type	Data Type
2010_h	Profile Start Velocity	Variable	UNSIGNED32

Table 69: Object Description (2010_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...268435455	0

Table 70: Entry Description (2010_h)

4.2.11 Object 2011_h: Profile A1

This object contains the acceleration value used for ramping up from the start velocity (object 2011_h, see Section 4.2.10) to the velocity V1 (object ($_h$2012), see Section 4.2.12).

Object Description			
Index	Name	Object Type	Data Type
2011_h	Profile A1	Variable	UNSIGNED32

Table 71: Object Description (2011_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...16777215	0

Table 72: Entry Description (2011_h)

4.2.12 Object 2012_h: Profile V1

This object contains the velocity used for the first segment of a positioning ramp.

Object Description			
Index	Name	Object Type	Data Type
2012_h	Profile V1	Variable	UNSIGNED32

Table 73: Object Description (2012_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...16777215	0

Table 74: Entry Description (2012_h)

4.2.13 Object 2013_h: Profile D1

This object contains the deceleration value used for decelerating from the maximum positioning velocity to the velocity V1 (object 2012_h, see Section 4.2.12).

Object Description			
Index	Name	Object Type	Data Type
2013_h	Profile D1	Variable	UNSIGNED32

Table 75: Object Description (2013_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...16777215	0

Table 76: Entry Description (2013_h)

4.2.14 Object 2015_h: Ramp Wait Time

This object defines the waiting time after ramping down to zero velocity before the next movement or direction inversion can start. Time range is 0 to 2 seconds. This setting avoids excess acceleration e.g. from positive stop velocity to negative start velocity.

Object Description			
Index	Name	Object Type	Data Type
2015_h	Ramp Wait Time	Variable	UNSIGNED16

Table 77: Object Description (2015_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...65535 [0.000032s]	0

Table 78: Entry Description (2015_h)

4.2.15 Object 2016_h: Profile A2

This object contains the acceleration value used for ramping up from velocity V1 (object 2012_h, see Section 4.2.12) to velocity V2 (object ($_h$2017), see Section 4.2.16).

Object Description			
Index	Name	Object Type	Data Type
2016_h	Profile A2	Variable	UNSIGNED32

Table 79: Object Description (2016_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...16777215	0

Table 80: Entry Description (2016_h)

4.2.16 Object 2017_h: Profile V2

This object contains the velocity used for the second segment of a positioning ramp.

Object Description			
Index	Name	Object Type	Data Type
2017_h	Profile V2	Variable	UNSIGNED32

Table 81: Object Description (2017_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...16777215	0

Table 82: Entry Description (2017_h)

4.2.17 Object 2018_h: Profile D2

This object contains the deceleration value used for decelerating from velocity V2 (object ($_h$2017), see Section 4.2.16) to the velocity V1 (object 2012_h, see Section 4.2.12).

Object Description			
Index	Name	Object Type	Data Type
2018_h	Profile D2	Variable	UNSIGNED32

Table 83: Object Description (2018_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...16777215	0

Table 84: Entry Description (2018_h)

4.2.18 Object 2089_h: Setting Delay

This object has to be used for setting a standstill period before the current is changed down to standby current.

Unit: 10msec

Object Description			
Index	Name	Object Type	Data Type
2089_h	Setting Delay	Variable	UNSIGNED16

Table 85: Object Description (2089_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...400	0

Table 86: Entry Description (2089_h)

4.2.19 Object $208C_h$: Velocity Dimension Index

With this object different units can be chosen:

- Writing 0 selects internal units.
- Writing 181 sets PPS for velocity and PPS/s for acceleration.

This can only be changed in SWITCHED_ON_DISABLED mode.

Object Description			
Index	Name	Object Type	Data Type
$208C_h$	Velocity Dimension Index	Variable	UNSIGNED8

Table 87: Object Description ($208C_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0/181	181

Table 88: Entry Description ($208C_h$)

4.2.20 Object $208E_h$: Acceleration Dimension Index

With this object, the unit for acceleration can be read out. The unit can be set using object $208C_h$. Object $208E_h$ reads 0 when internal units are selected and 179 when PPS/s is selected.

Object Description			
Index	Name	Object Type	Data Type
$208E_h$	Acceleration Dimension Index	Variable	UNSIGNED8

Table 89: Object Description ($208E_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	0/179	179

Table 90: Entry Description ($208E_h$)

4.2.21 Object 2092_h: Chopper Blank Time

This object serves for selecting the comparator blank time. This time needs to safely cover the switching event and the duration of the ringing on the sense resistor. For low current drivers, a setting of 1 or 2 is good. For higher current applications, a setting of 2 or 3 is required.

Object Description			
Index	Name	Object Type	Data Type
2092_h	Chopper Blank Time	Variable	UNSIGNED8

Table 91: Object Description (2092_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...3	2

Table 92: Entry Description (2092_h)

4.2.22 Object 2093_h: Chopper Mode

Select the chopper mode using this object:

- 0 – SpreadCycle chopper
- 1 – classic constant off time chopper

Object Description			
Index	Name	Object Type	Data Type
2093_h	Chopper Mode	Variable	UNSIGNED8

Table 93: Object Description (2093_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0/1	0

Table 94: Entry Description (2093_h)

4.2.23 Object 2094_h: Chopper Disable Fast Decay Comparator

When classic constant off time chopper mode is selected instead of SpreadCycle chopper, setting this parameter to 1 disables the current comparator usage for termination of the fast decay cycle. In SpreadCycle chopper mode this setting has no effect.

- 0 – current comparator enabled
- 1 – current comparator disabled

Object Description			
Index	Name	Object Type	Data Type
2094_h	Chopper Disable Fast Decay Comparator	Variable	UNSIGNED8

Table 95: Object Description (2094_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0/1	0

Table 96: Entry Description (2094_h)

4.2.24 Object 2095_h: Chopper Hysteresis End

This object provides the setting of the hysteresis end value after a number of decrements. The decrement interval time is controlled by object 2094_h (Section 4.2.23).
Possible values are as follows:

- -3...-1 – negative hysteresis end setting
- 0 – zero hysteresis end setting
- 1...12 – positive hysteresis end setting

Object Description			
Index	Name	Object Type	Data Type
2095_h	Chopper Hysteresis End	Variable	SIGNED8

Table 97: Object Description (2095_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	-3...12	0

Table 98: Entry Description (2095_h)

4.2.25 Object 2096_h: Chopper Hysteresis Start

This object provides the hysteresis start setting. Note that this value is an offset to the hysteresis end value.

Object Description			
Index	Name	Object Type	Data Type
2096_h	Chopper Hysteresis Start	Variable	UNSIGNED8

Table 99: Object Description (2096_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...8	3

Table 100: Entry Description (2096_h)

4.2.26 Object 2097_h: Chopper Off Time

The off time setting controls the minimum chopper frequency. Under normal circumstances, an off time within the range of 5μs to 20μs is used. Off time setting for constant t_{OFF} chopper: $N_{CLK} = 12 + 32 \times t_{OFF}$.
Minimum is 64 clocks.
Setting this parameter to zero completely disables all driver transistors and so lets the motor freewheel.

Object Description			
Index	Name	Object Type	Data Type
2097_h	Chopper Off Time	Variable	UNSIGNED8

Table 101: Object Description (2097_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0 / 2...15	5

Table 102: Entry Description (2097_h)

4.2.27 Object 2098_h: Smart Energy Current Minimum

This object provides the setting of the lower motor current limit for CoolStep operation by scaling the CS value.
Possible values are as follows:

- 0 – $1/2$ of maximum motor current setting (Section 4.2.4)
- 1 – $1/4$ of maximum motor current setting (Section 4.2.4)

Object Description			
Index	Name	Object Type	Data Type
2098_h	Smart Energy Current Minimum	Variable	UNSIGNED8

Table 103: Object Description (2098_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0/1	0

Table 104: Entry Description (2098_h)

4.2.28 Object 2099_h: Smart Energy Current Down Step

This object provides the setting of the number of StallGuard2 readings above the upper threshold necessary for each current decrement of the motor current.
Possible values are as follows:

- 0 – 32 measurements – slowest decrement
- 1 – 8 measurements
- 2 – 2 measurements
- 3 – 1 measurements – fastest decrement

Object Description			
Index	Name	Object Type	Data Type
2099_h	Smart Energy Current Down Step	Variable	UNSIGNED8

Table 105: Object Description (2099_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...3	0

Table 106: Entry Description (2099_h)

4.2.29 Object $209A_h$: Smart Energy Hysteresis

This object sets the distance between the lower and the upper threshold for StallGuard2 reading. Above the upper threshold the motor current becomes decreased.

Hysteresis: $(\text{SmartEnergy hysteresis value} + 1) \times 32$
Upper StallGuard2 threshold: $(\text{SmartEnergy hysteresis start} + \text{SmartEnergy hysteresis} + 1) \times 32$

Object Description			
Index	Name	Object Type	Data Type
$209A_h$	Smart Energy Hysteresis	Variable	UNSIGNED8

Table 107: Object Description ($209A_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...15	0

Table 108: Entry Description ($209A_h$)

4.2.30 Object $209B_h$: Smart Energy Current Up Step

This object sets the current increment step. The current becomes incremented for each measured StallGuard2 value below the lower threshold (see smart energy hysteresis start (object $209C_h$, Section 4.2.31). Possible values are as follows:

- 0 – 1 step – slowest increment
- 1 – 2 steps
- 2 – 4 steps
- 3 – 8 steps – fastest increment

Object Description			
Index	Name	Object Type	Data Type
$209B_h$	Smart Energy Current Up Step	Variable	UNSIGNED8

Table 109: Object Description ($209B_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...3	0

Table 110: Entry Description ($209B_h$)

4.2.31 Object $209C_h$: Smart Energy Hysteresis Start

This object serves to set the lower threshold for the StallGuard2 value (see smart Energy current up step (Section 4.2.30). Setting this to 0 disables the CoolStep function.

Object Description			
Index	Name	Object Type	Data Type
$209C_h$	Smart Energy Hysteresis Start	Variable	UNSIGNED8

Table 111: Object Description ($209C_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...15	0

Table 112: Entry Description ($209C_h$)

4.2.32 Object $209D_h$: Smart Energy Filter Enable

This object is used to set the StallGuard2 filter for more precision of the measurement. It reduces the measurement frequency to one measurement per four fullsteps if set.
In most cases it is expedient to set the filtered mode when using CoolStep. Use the standard mode for step loss detection.
Possible values are as follows:

- 0 – standard mode
- 1 – filtered mode

Object Description			
Index	Name	Object Type	Data Type
$209D_h$	Smart Energy Filter Enable	Variable	UNSIGNED8

Table 113: Object Description ($209D_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0/1	0

Table 114: Entry Description ($209D_h$)

4.2.33 Object $209E_h$: StallGuard2 Threshold

This signed value controls the StallGuard2 threshold level for stall output and sets the optimum measurement range for readout. A lower value gives a higher sensitivity. Zero is the starting value. A higher value makes StallGuard2 less sensitive and requires more torque to indicate a stall.

Object Description			
Index	Name	Object Type	Data Type
$209E_h$	StallGuard2 Threshold	Variable	SIGNED8

Table 115: Object Description ($209E_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	-63…63	0

Table 116: Entry Description ($209E_h$)

4.2.34 Object $20A1_h$: Short Protection Disable

This object is used to enable or to disable the short to ground protection. Normally there is no need to change this. Use the default value.
Possible values are as follows:

- 0 – Short to GND protection enabled
- 1 – Short to GND protection disabled

Object Description			
Index	Name	Object Type	Data Type
$20A1_h$	Short Protection Disable	Variable	UNSIGNED8

Table 117: Object Description ($20A1_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0/1	0

Table 118: Entry Description ($20A1_h$)

4.2.35 Object $20A2_h$: Global Current Scaler

Global scaling of the motor current. This value is multiplied with the current scaling in order to adapt the driver to a certain motor type. It should be chosen before tuning other parameters as it also influences the chopper hysteresis. This setting is intended for fine tuning the maximum motor current. The following settings are possible:

- 0 – Full scale (100% of maximum)
- 1…31 – Do not use
- 32…255 – 32/256…255/256 of maximum

Object Description			
Index	Name	Object Type	Data Type
$20A2_h$	Global Current Scaler	Variable	Record

Table 119: Object Description ($20A2_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0…255	0

Table 120: Entry Description ($20A2_h$)

4.2.36 Object $20A3_h$: Driver Current Range

This setting allows a basic adaption of the driver current sensing to the motor current range. Select the lowest fitting range for best current precision. Possible settings are as follows:

- 0 – maximum motor current 0.7A RMS / 1A peak
- 1 – maximum motor current 1.4A RMS / 2A peak
- 2 – maximum motor current 2.1A RMS / 3A peak
- 3 – maximum motor current 2.1A RMS / 3A peak (same as 2)

Object Description			
Index	Name	Object Type	Data Type
$20A3_h$	Vsense	Variable	UNSIGNED8

Table 121: Object Description ($20A3_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...3	3

Table 122: Entry Description ($20A3_h$)

4.2.37 Object $20A4_h$: Stop on Stall

Below this speed the motor is not stopped. Above this speed the motor is stopped in case the StallGuard2 load value reaches zero.

Object Description			
Index	Name	Object Type	Data Type
$20A4_h$	Stop on Stall	Variable	UNSIGNED32

Table 123: Object Description ($20A4_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...2147483647	0

Table 124: Entry Description ($20A4_h$)

4.2.38 Object $20A5_h$: Smart Energy Threshold Speed

The CoolStep functionality is enabled when the actual speed is above this speed. It is disabled again when the actual speed drops below this value.

Object Description			
Index	Name	Object Type	Data Type
$20A5_h$	Smart Energy Threshold Speed	Variable	UNSIGNED32

Table 125: Object Description ($20A5_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...2147483647	0

Table 126: Entry Description ($20A5_h$)

4.2.39 Object $20A7_h$: Stop on Stall Option Code

This object controls what to do when the motor is stopped by the stop on stall function. Possible options are as follows:

- 0: enter FAULT state on stall
- 1: just stop motor on stall, stay in OPERATIONAL state

Object Description			
Index	Name	Object Type	Data Type
$20A7_h$	Stop on Stall Option Code	Variable	UNSIGNED8

Table 127: Object Description ($20A7_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0/1	0

Table 128: Entry Description ($20A7_h$)

4.2.40 Object $20A8_h$: StallGuard4 Threshold Value

When StallGuard4 is active, the motor is stopped when the StallGuard4 result falls below this value.

Object Description			
Index	Name	Object Type	Data Type
$20A8_h$	StallGuard4 Threshold Value	Variable	UNSIGNED16

Table 129: Object Description ($20A8_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...511	0

Table 130: Entry Description ($20A8_h$)

4.2.41 Object $20A9_h$: StallGuard4 Filter Enable

This object is used to set the StallGuard4 filter for more precision of the measurement. The StallGuard4 result then always gives the average of the last four measurements.
Possible values are as follows:

- 0 – standard mode
- 1 – filtered mode

Object Description			
Index	Name	Object Type	Data Type
$20A9_h$	StallGuard4 Filter Enable	Variable	UNSIGNED8

Table 131: Object Description ($20A9_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0/1	0

Table 132: Entry Description ($20A9_h$)

4.2.42 Object $20AA_h$: StallGuard4 Angle Offset

This object is used to set the StallGuard4 angle offset. It optimizes the switchover between StealthChop2 and SpreadCycle chopper mode (and hence between StallGuard4 and StallGuard2) when switched on. Possible values are as follows:

- 0 – off
- 1 – on

Object Description			
Index	Name	Object Type	Data Type
$20AA_h$	StallGuard4 Angle Offset	Variable	UNSIGNED8

Table 133: Object Description ($20AA_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0/1	0

Table 134: Entry Description ($20AA_h$)

4.2.43 Object $20B0_h$: PWM Threshold Speed

The StealthChop feature is switched on when the value of this object is greater than zero and the actual velocity is lower than the value set by this object.

Object Description			
Index	Name	Object Type	Data Type
$20B0_h$	PWM Threshold Speed	Variable	UNSIGNED32

Table 135: Object Description ($20B0_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...16777215	0

Table 136: Entry Description ($20B0_h$)

4.2.44 Object $20B1_h$: PWM Gradient

Velocity dependent gradient for the PWM amplitude (StealthChop). Setting this value to 0 turns off StealthChop.

Object Description			
Index	Name	Object Type	Data Type
$20B1_h$	PWM Gradient	Variable	UNSIGNED8

Table 137: Object Description ($20B1_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...255	0

Table 138: Entry Description ($20B1_h$)

4.2.45 Object $20B2_h$: PWM Amplitude

Maximum PWM amplitude when switching to StealthChop mode. Do not set too low. Values above 64 are recommended.

Object Description			
Index	Name	Object Type	Data Type
$20B2_h$	PWM Amplitude	Variable	UNSIGNED8

Table 139: Object Description ($20B2_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...255	0

Table 140: Entry Description ($20B2_h$)

4.2.46 Object $20B3_h$: DcStep Minimum Speed

Minimum speed for switching to DcStep. The motor driver does not switch to DcStep mode when the actual velocity is below this value. Setting this object to zero completely switches off DcStep.

Object Description			
Index	Name	Object Type	Data Type
$20B3_h$	DcStep Minimum Speed	Variable	UNSIGNED32

Table 141: Object Description ($20B3_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0…16777215	0

Table 142: Entry Description ($20B3_h$)

4.2.47 Object $20B4_h$: DcStep Time

This setting controls the reference pulse width for DcStep load measurement. It must be optimized for robust operation with maximum motor torque. A higher value allows higher torque and higher velocity, a lower value allows operation down to a lower velocity as set by the minimum DcStep speed (object $20B3_h$, see Section 4.2.46).

Object Description			
Index	Name	Object Type	Data Type
$20B4_h$	DcStep Time	Variable	UNSIGNED16

Table 143: Object Description ($20B4_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0…1023	0

Table 144: Entry Description ($20B4_h$)

4.2.48 Object $20B5_h$: DcStep StallGuard

This setting controls stall detection in DcStep mode. Increase this value for higher sensitivity.

Object Description			
Index	Name	Object Type	Data Type
$20B5_h$	DcStep StallGuard	Variable	UNSIGNED8

Table 145: Object Description ($20B5_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0…255	0

Table 146: Entry Description ($20B5_h$)

4.2.49 Object $20B6_h$: Fullstep Threshold Speed

Depending on the settings of objects $20B7_h$ and $20B8_h$ (see Sections 4.2.50 and 4.2.51) the driver switches to fullstep mode and/or to a different chopper mode when the speed set by this object is exceeded.

Object Description			
Index	Name	Object Type	Data Type
$20B6_h$	Fullstep Threshold Speed	Variable	UNSIGNED32

Table 147: Object Description ($20B6_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0…16777215	0

Table 148: Entry Description ($20B6_h$)

4.2.50 Object $20B7_h$: High-Speed Chopper Mode

The motor driver switches to a different chopper mode when this object is set to 1 and the measured speed is greater than the threshold speed set by object $20B6_h$ (see Section 4.2.49).

Object Description			
Index	Name	Object Type	Data Type
$20B7_h$	High Speed Chopper Mode	Variable	UNSIGNED8

Table 149: Object Description ($20B7_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0/1	0

Table 150: Entry Description ($20B7_h$)

4.2.51 Object $20B8_h$: High-Speed Fullstep Mode

The motor driver switches to fullstep mode when this object is set to 1 and the measured speed is greater than the threshold speed set by object $20B6_h$ (see Section 4.2.49).

Object Description			
Index	Name	Object Type	Data Type
$20B8_h$	High Speed Fullstep Mode	Variable	UNSIGNED8

Table 151: Object Description ($20B8_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0/1	0

Table 152: Entry Description ($20B8_h$)

4.2.52 Object $20B9_h$: Power Down Ramp

The value set by this object controls the number of clock cycles for motor power down after a motion as soon as the motor has stopped and the setting time (set by object 2089_h, see Section 4.2.18) has expired. The smooth transition avoids a motor jerk upon power down.

- 0=instant power down.
- 15=longest possible power down ramp.

Object Description			
Index	Name	Object Type	Data Type
$20B9_h$	Power Down Ramp	Variable	UNSIGNED8

Table 153: Object Description ($20B9_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0…15	7

Table 154: Entry Description ($20B9_h$)

4.2.53 Object 2100_h: Home Offset Display

This object shows the home offset. The value is given in microsteps.

Object Description			
Index	Name	Object Type	Data Type
2100_h	Home Offset Display	Variable	SIGNED32

Table 155: Object Description (2100_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	-2147483648...2147483647	0

Table 156: Entry Description (2100_h)

4.2.54 Object 2101_h: Actual Load Value

This object shows the actual load value used for stall detection (StallGuard2).

Object Description			
Index	Name	Object Type	Data Type
2101_h	Actual Load Value	Variable	UNSIGNED16

Table 157: Object Description (2101_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	0...1023	0

Table 158: Entry Description (2101_h)

4.2.55 Object 2102_h: Driver Error Flags

This object shows the error flags of the motor driver IC.

Error Flags		
Bit	Name	Meaning
7	OT	Overtemperature
6	OTPW	Temperature pre-warning
5	UV	Undervoltage
4	OCHS	Overcurrent high side
3	OLB	Open load on bridge B
2	OLA	Open load on bridge A
1	OCB	Overcurrent on bridge B
0	OCA	Overcurrent on bridge A

Table 159: Driver Error Flags (2102_h)

Object Description			
Index	Name	Object Type	Data Type
2102_h	Driver Error Flags	Variable	UNSIGNED8

Table 160: Object Description (2102_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	0...255	0

Table 161: Entry Description (2102_h)

4.2.56 Object 2107_h: Microstep Resolution Display

This object shows the microstep resolution, set by object 2000_h (see Section 4.2.1).

Object Description			
Index	Name	Object Type	Data Type
2107_h	Microstep resolution display	Variable	UNSIGNED8

Table 162: Object Description (2107_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	0…8	8

Table 163: Entry Description (2107_h)

4.2.57 Object $210A_h$: StallGuard4 Result

This object shows the actual load value used for stall detection when StallGuard4 is active.

Object Description			
Index	Name	Object Type	Data Type
$210A_h$	StallGuard4 Result	Variable	UNSIGNED16

Table 164: Object Description ($210A_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	0…511	0

Table 165: Entry Description ($210A_h$)

4.2.58 Object 2121_h: PWM Scale Value

Actual PWM scale value used when the motor driver is operaing in StealthChop mode.

Object Description			
Index	Name	Object Type	Data Type
2121_h	PWM Scale Value	Variable	UNSIGNED8

Table 166: Object Description (2121_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	0…255	—

Table 167: Entry Description (2121_h)

4.2.59 Object 2122_h: Measured Velocity

This object contains the velocity measured by the motor driver. This value is important only when the motor driver is operating in DcStep mode.

Object Description			
Index	Name	Object Type	Data Type
2122_h	Measured Velocity	Variable	SIGNED32

Table 168: Object Description (2122_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	-16777215…16777215	—

Table 169: Entry Description (2122_h)

4.2.60 Object 2702_h: Device Digital Inputs

Bits 23…16 of this object reflect the states of the general-purpose inputs of the module. The number of available inputs depends on the module type.

Bit Definitions	
Bit	Description
16	IN0
17	IN1
18	IN2
19	IN3

Table 170: Bit Definitions (2702_h)

Object Description			
Index	Name	Object Type	Data Type
2702_h	Device Digital Inputs	Variable	UNSIGNED32

Table 171: Object Description (2702_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	yes	–	0

Table 172: Entry Description (2702_h)

4.2.61 Object 2703_h: Device Digital Outputs

With this object the digital outputs (general-purpose outputs) can be set. Bits 23…16 of sub index 1 switch the outputs of the module. Bits 23…16 of sub index 2 determine which outputs can be switched. The

number of available digital outputs depends on the module type.

Bit Definitions	
Bit	Description
16	OUT0
17	OUT1
18	OUT2
19	OUT3

Table 173: Bit Definitions (2703_h)

Object Description			
Index	Name	Object Type	Data Type
2703_h	Device Digital Outputs	Variable	ARRAY

Table 174: Object Description (2703_h)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
1	Physical outputs	rw	yes	UNSIGNED32	0
2	Output mask	rw	no	UNSIGNED32	0

Table 175: Entry Description (2703_h)

Note Some outputs can also be used for automatically controlling a brake that can be connected to the module through these outputs. In order to be able to control these outputs through this object the automatic brake control function has to be disabled. Do this by writing 0 to sub-index 1 and sub-index 2 of object $200C_h$.

4.2.62 Object 2704_h: CAN Bit Rate

With this object it is possible to change the CAN bit rate. To do this, first write the new value to this object. Then, store the new setting by writing the save signature to object 2706h. After that, reset the module. The new setting then becomes active.

Note The CAN bit rate can also be set using LSS.

Object Description			
Index	Name	Object Type	Data Type
2704_h	CAN Bit Rate	Variable	UNSIGNED16

Table 176: Object Description (2704_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	20,50,100,125,250,500,800,1000	1000

Table 177: Entry Description $(2704)_h$

How to change the bit rate of a module:

- Write new bit rate in object 2704_h.
- Write the save signature 65766173_h to sub-index 1 of object 2706_h.
- Reset module.

4.2.63 Object 2705_h: Node ID

On modules that do not have address switches the node ID can be selected using this object. On modules with address switches the node ID is normally selected using the address switches. Only when the address switches are set to an invalid value (0 or >127) this object overrides the address switch setting.
To change the node ID, first write the new node ID to this object. Then, store the new setting by writing the save signature to object 2706_h. After that, reset the module. The new setting then becomes active.

Note The node ID can also be set using LSS. Setting object 2706_h to 255 sets the device into LSS unconfigured state. In this case the device cannot be reached by SDO until the node ID has been set to a valid value using LSS.

Object Description			
Index	Name	Object Type	Data Type
2705_h	Node ID	Variable	UNSIGNED8

Table 178: Object Description (2705_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	1…127 and 255	1

Table 179: Entry Description (2705_h)

How to change the node ID of a module:

- Write new node ID to object 2705_h.
- Write save signature 65766173_h to sub-index 1 of object 2706_h.
- Reset the module.

4.2.64 Object 2706_h: Store

Writing the save signature to this object permanently saves changes made to objects 2704_h and 2705_h. The save signature is 65766173_h.

Object Description			
Index	Name	Object Type	Data Type
2706_h	Store	Array	UNSIGNED32

Table 180: Object Description (2706_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
1	rw	no	save signature: 65766173_h	1

Table 181: Entry Description (2706_h)

4.2.65 Object 2707_h: CAN Bit Rate Load

This object shows the selected CAN bit rate.

Object Description			
Index	Name	Object Type	Data Type
2707_h	CAN Bit Rate Load	Variable	UNSIGNED8

Table 182: Object Description (2707_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	20…1000	depends on bit rate

Table 183: Entry Description $(2707)_h$

4.2.66 Object 2708_h: Node ID Load

This object shows the selected node ID.

Object Description			
Index	Name	Object Type	Data Type
2708_h	Node ID Load	Variable	UNSIGNED8

Table 184: Object Description (2708_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	1..127	Depends on node ID setting

Table 185: Entry Description (2708_h)

4.2.67 Object $270E_h$: Device Analog Inputs

This object shows the values of the analog inputs of the device.

Object Description			
Index	Name	Object Type	Data Type
$270E_h$	Device Analog Inputs	Array	UNSIGNED32

Table 186: Object Description ($270E_h$)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
1	Analog input 0	ro	yes	0…4095	–
2	Analog input 1	ro	yes	0…4095	–

Table 187: Entry Description ($270E_h$)

4.2.68 Object 2711_h: I/O Mode

This object is a bit vector that selects input or output mode for each general-purpose input/output. A bit that is set to 0 selects input mode and a bit that is set to 1 selects output mode. Furthermore, it is possible to select between high side mode and push-pull mode for outputs.

Value Definition	
Bit	Description
0	0: DOI0 is an input. 1: DOI0 is an output.
1	0: DOI1 is an input. 1: DOI1 is an output.
2	0: DOI2 is an input. 1: DOI2 is an output.
3	0: DOI3 is an input. 1: DOI3 is an output.
8	0: DOI0 uses high side mode when in output mode. 1: DOI0 uses push-pull mode when in output mode.
9	0: DOI1 uses high side mode when in output mode. 1: DOI1 uses push-pull mode when in output mode.
10	0: DOI2 uses high side mode when in output mode. 1: DOI2 uses push-pull mode when in output mode.
11	0: DOI3 uses high side mode when in output mode. 1: DOI3 uses push-pull mode when in output mode.

Table 188: Value Description (2711_h)

Object Description			
Index	Name	Object Type	Data Type
2711_h	I/O mode	Variable	UNSIGNED16

Table 189: Object Description (2711_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0…3855	0

Table 190: Entry Description (2711_h)

5 Profile Specific Area

The profile segment contains CiA-402 standard motion control objects. These objects control the motion control functions of the TMCM-3216. Since it is not possible to operate the modes in parallel, it is possible to activate the required function by selecting a mode of operation. The control device writes to the modes of operation object to select the operation mode. The drive device provides the modes of operation display object to indicate the actual activated operation mode. Controlword, statusword, and setpoints are used mode-specific. This implies the responsibility of the control device to avoid inconsistencies and erroneous behavior.

The following operating modes (selectable through object 6060_h, see 5.1.6) are implemented on the TMCM-3216:

- Profile position mode (pp)
- Profile velocity mode (pv)
- Homing mode (hm)
- Cyclic position mode (csp)
- Cyclic velocity mode (csv)

Info

This section of the manual only shows the object indices for motor #0. Of course the same objects are also available for the other motors. For the other motors, add an offset of $motor\_number \cdot 800_h$ to the object index. So, for example, the control word (object 6040_h for motor #0) can be accessed as object 6840_h for motor #1 and as object 7040_h for motor #2. See also Section 2.6.1.

Multi-Axis Object Indices	
Motor	Object Index Range
Motor #0	6000_h – $67FF_h$
Motor #1	6800_h – $6FFF_h$
Motor #2	7000_h – $77FF_h$

Table 191: Multi-Axis Object Indices (Profile Specific Area)

5.1 Detailed Object Specifications

5.1.1 Object $605A_h$: Quick Stop Option Code

This object indicates what action is performed when the quick stop function is executed. The slow down ramp is the deceleration value of the used mode of operation. The following quick stop option codes are supported in the current version of the CANopen firmware:

Value Definition	
Value	Definition
1	Slow down on *slow down ramp* and transit into *switch on disabled*
2	Slow down on *quick stop ramp* and transit into *switch on disabled*
5	Slow down on *slow down ramp* and stay in *quick stop active*
6	Slow down on *quick stop ramp* and stay in *quick stop active*

Table 192: Value Description ($605A_h$)

Object Description			
Index	Name	Object Type	Data Type
$605A_h$	Quick Stop Option Code	Variable	SIGNED16

Table 193: Object Description ($605A_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	1/2/5/6	2

Table 194: Entry Description ($605A_h$)

5.1.2 Object $605B_h$: Shutdown Option Code

This object indicates what action is performed if there is a transition from *operation enabled* state to *ready to switch on state*. The shutdown option code always has the value 0 as only this is supported.

Value Definition	
Value	Definition
0	Disable drive function (switch off the power stage)

Table 195: Value Description ($605B_h$)

Object Description			
Index	Name	Object Type	Data Type
$605B_h$	Shutdown Option Code	Variable	SIGNED16

Table 196: Object Description ($605B_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0	0

Table 197: Entry Description ($605B_h$)

5.1.3 Object $605C_h$: Disable Operation Option Code

This object indicates what action is performed if there is a transition from *operation enabled* state to *switched on* state. The disable operation option code always has the value 1 as only this is supported. The slow down ramp is the deceleration value of the used mode of operation.

Value Definition	
Value	Definition
1	Slow down on slow down ramp

Table 198: Value Description ($605C_h$)

Object Description			
Index	Name	Object Type	Data Type
$605C_h$	Disable Operation Option Code	Variable	SIGNED16

Table 199: Object Description ($605C_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	1	1

Table 200: Entry Description ($605C_h$)

5.1.4 Object $605D_h$: Halt Option Code

This object indicates what action is performed when the halt function is executed. The slow down ramp is the deceleration value of the used mode of operation.

Value Definition	
Value	Definition
1	Slow down on slow down ramp and stay in *operation enabled*

Table 201: Value Description ($605D_h$)

Object Description			
Index	Name	Object Type	Data Type
$605D_h$	Halt Option Code	Variable	SIGNED16

Table 202: Object Description ($605D_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	1	1

Table 203: Entry Description ($605D_h$)

5.1.5 Object $605E_h$: Fault Reaction Option Code

This object indicates what action is performed when fault is detected in the power drive system. The slow down ramp is the deceleration value of the used mode of operation. The fault reaction option code always has the value 2 as only this is supported.

Value Definition	
Value	Definition
2	Slow down on quick stop ramp

Table 204: Value Description ($605E_h$)

Object Description			
Index	Name	Object Type	Data Type
$605E_h$	Fault Reaction Option Code	Variable	SIGNED16

Table 205: Object Description ($605E_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	2	2

Table 206: Entry Description ($605E_h$)

5.1.6 Object 6060_h: Modes of Operation

This object indicates the requested operation mode. Supported operating modes are as follows:

Value Definition	
Value	Mode
0	No mode
1	Profile position mode (pp)
3	Profile velocity mode (pv)
6	Homing mode (hm)
8	Cyclic synchronous position mode (csp)
9	Cyclic synchronous velocity mode (csv)

Table 207: Value Description (6060_h)

The motor does not run when the operating mode is set to 0. It is stopped when the motor is running in one of the supported operating modes and the operating mode is then switched to 0.

Object Description			
Index	Name	Object Type	Data Type
6060_h	Modes of Operation	Variable	SIGNED8

Table 208: Object Description (6060_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	See Table 207	0

Table 209: Entry Description (6060_h)

5.1.7 Object 6061_h: Modes of Operation Display

This object shows the operating mode that is currently set.

Value Definition	
Value	Mode
0	No mode
1	Profile position mode (pp)
3	Profile velocity mode (pv)
6	Homing mode (hm)
8	Cyclic synchronous position mode (csp)
9	Cyclic synchronous velocity mode (csv)

Table 210: Value Description (6061_h)

The motor does not run when the operating mode is set to 0. It is stopped when the motor is running in one of the supported operating modes and the operating mode is then switched to 0.

Object Description			
Index	Name	Object Type	Data Type
6061_h	Modes of Operation Display	Variable	SIGNED8

Table 211: Object Description (6061_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	See Table 210	0

Table 212: Entry Description (6061_h)

5.1.8 Object $606A_h$: Sensor Selection Code

This object provides the source of the position sensor actual value. It selects whether an encoder is to be used or not.

Value Definition	
Value	Mode
0	Encoder used
-1	No encoder

Table 213: Value Description ($606A_h$)

Object Description			
Index	Name	Object Type	Data Type
$606A_h$	Sensor selection code	Variable	SIGNED16

Table 214: Object Description ($606A_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0/-1	-1

Table 215: Entry Description ($606A_h$)

5.1.9 Object 608F$_h$: Position Encoder Resolution

This object defines the resolution of the encoder. The position encoder resolution is calculated by the following formula:

$$position\ encoder\ resolution = \frac{encoder\ increments}{motor\ revolutions}$$

All values are dimensionless.

Object Description			
Index	Name	Object Type	Data Type
608F$_h$	Position Encoder Resolution	Array	UNSIGNED32

Table 216: Object Description (608F$_h$)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
0	Highest sub-index supported	ro	no	2	2
1	Encoder increments	rw	no	0...2147483647	1
2	Motor revolutions	ro	no	1	1

Table 217: Entry Description (608F$_h$)

5.1.10 Object 60FD$_h$: Digital Inputs

This object contains the states of the digital inputs of the module. Starting from bit 0, every bit reflects the state of one digital input. The number of valid bits depends on the number of digital inputs on the module used.

Object Description			
Index	Name	Object Type	Data Type
60FD$_h$	Digital Inputs	Variable	UNSIGNED32

Table 218: Object Description (60FD$_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	yes	UNSIGNED32	0

Table 219: Entry Description (60FD$_h$)

5.1.11 Object 6502$_h$: Supported Drive Modes

This object provides information on the supported drive modes. A bit that is set means that the mode is supported, a bit that is not set means that the mode is not supported by the drive.

Value Definition	
Bit	Mode
0	Profile position mode (pp)
1	Velocity mode (vl)
2	Profile velocity mode (pv)
3	Torque mode (tq)
4	Reserved
5	Homing mode (hm)
6	Interpolated position mode (ip)
7	Cyclic synchronous position mode (csp)
8	Cyclic synchronous velocity mode (csv)
9	Cyclic synchronous torque mode (cst)

Table 220: Value Definition (6502_h)

Object Description			
Index	Name	Object Type	Data Type
6502_h	Supported Drive Modes	Variable	UNSIGNED32

Table 221: Object Description (6502_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	UNSIGNED32	Depends on supported modes.

Table 222: Entry Description (6502_h)

5.1.12 Object $67FF_h$: Single Device Type

This object provides information on the device profile used for the individual axis. Its structure is similar to object 1000_h. The lower sixteen bits contain the device profile number, which is always 402 (0192_h) with Trinamic motion control modules. The upper sixteen bits contain more information about the drive profile.

Value Definition	
Bit	Meaning
0…15	Device profile number
16	Frequency converter (0 = no, 1 = yes)
17	Servo drive (0 = no, 1 = yes)
18	Stepper motor (0 = no, 1 = yes)
19…21	Reserved
22	PDO set for generic drive device (0 = yes, 1 = no)
23	Multi device module (0 = no, 1 = yes)
24…31	Reserved

Table 223: Value Definition ($67FF_h$)

Object Description			
Index	Name	Object Type	Data Type
$67ff_h$	Single Device Type	Variable	UNSIGNED32

Table 224: Object Description ($67FF_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	UNSIGNED32	Depends on individual axis.

Table 225: Entry Description ($67FF_h$)

6 Profile Position Mode

A target position is applied to the trajectory generator. It generates a position demand value for the position control loop described in the position control function.

See object 6060_h (Section 5.1.6) on how to choose an operation mode. Object 6061_h (Section 5.1.7) shows the operation mode that is set.

6.1 Detailed Object Specifications

The following text offers detailed object specifications. For a better understanding, it is necessary to see how the state machine works.

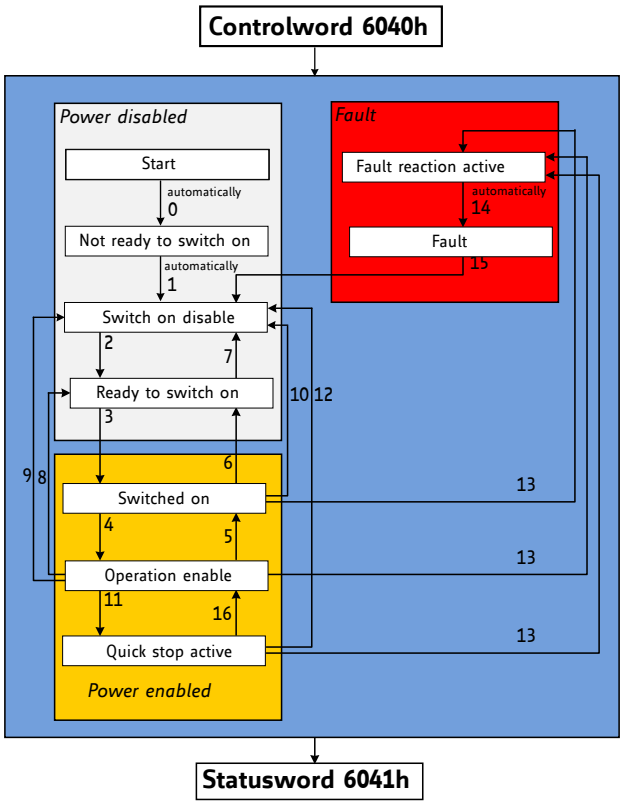


Figure 9: DS402 Finite State Machine

Notes on state transitions:

- Commands directing a change in state are processed completely and the new state achieved before additional state change commands are processed.
- Transitions 0 and 1 occur automatically at drive power-on or reset. Transition 14 occurs automatically too. All other state changes must be directed by the host.
- Drive function disabled indicates that no current is being supplied to the motor.
- Drive function enabled indicates that current is available for the motor and profile position and profile velocity reference values may be processed.

6.1.1 Object 6040_h: Controlword

This object indicates the received command controlling the power drive system finite state machine (PDS FSM). The CiA-402 state machine can be controlled using this object. See Figure 9 for detailed information.

Structure of the Controlword										
15 11	10	9	8	7	6 4	3	2	1	0	
nu	r	oms	h	fr	oms	eo	qs	ev	so	
MSB										LSB

Legend: nu = not used; r = reserved; oms = operation mode specific; h = halt; fr = fault reset; eo = enable operation; qs = quick stop; ev = enable voltage; so = switch on.

Table 226: Structure of the Controlword in pp Mode

Operation Mode Specific Bits in pp Mode		
Bit	Name	Definition
4	New set point	0-to-1: the next positioning is started.
5	Change immediately	Not supported.
6	Absolute / relative	0: New position is absolute. 1: New position is relative.
9	Change set point	Not supported.

Table 227: Operation Mode Specific Bits in pp Mode

Command Coding						
Command	Bits of Controlword					Transitions
	Bit 7	Bit 3	Bit 2	Bit 1	Bit 0	
Shutdown	0	x	1	1	0	2, 6, 8
Switch on	0	0	1	1	1	3
Switch on and enable operation	0	1	1	1	1	3, 4
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	0-to-1	x	x	x	x	15

Table 228: Command Coding

Object Description			
Index	Name	Object Type	Data Type
6040_h	Controlword	Variable	UNSIGNED16

Table 229: Object Description (6040_h in pp Mode)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	See command coding above.	

Table 230: Entry Description (6040_h in pp Mode)

6.1.2 Object 6041_h: Statusword

This object provides the status of the PDS FSA. It reflects the status of the CiA-402 state machine. See Figure 9 for detailed information. The object is structured as defined below.

For more information about the coding, refer to the CANopen drives and motion control device profile, part 2.

Structure of the Statusword															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
dir	mot	oms		ila	tr	rm	ms	w	sod	qs	ve	f	oe	so	rtso
MSB															LSB

Legend: nu = not used; r = reserved; oms = operation mode specific; h = halt; fr = fault reset; oe = operation enable; qs = quick stop; ve = voltage enable; so = switch on.

Table 231: Structure of the Statusword in pp Mode

Trinamic Specific Bits		
Bit	Name	Definition
14	Motor activity	0: Motor stands still. 1: Motor rotates.
15	Direction of rotation	This bit shows the direction of rotation.

Table 232: Trinamic Specific Bits

Operation Mode Specific Bits in pp Mode		
Bit	Name	Definition
10	Target reached	Set when the motor is within the position window.
12	Set point acknowledged	0: Set point processed. 1: Set point still in process.
13	Following error	Not supported.

Table 233: Operation Mode Specific Bits in pp Mode

State Coding	
Statusword	FSA State
xxxx xxxx x0xx 0000_h	Not ready to switch on
xxxx xxxx x1xx 0000_h	Switch on disabled
xxxx xxxx x01x 0001_h	Ready to switch on
xxxx xxxx x01x 0011_h	Switched on
xxxx xxxx x01x 0111_h	Operation enabled
xxxx xxxx x00x 0111_h	Quick stop active
xxxx xxxx x0xx 1111_h	Fault reaction active
xxxx xxxx x0xx 1000_h	Fault

Table 234: State Coding

Object Description			
Index	Name	Object Type	Data Type
6041_h	Controlword	Variable	UNSIGNED16

Table 235: Object Description (6041_h in pp Mode)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	See state coding above.	

Table 236: Entry Description (6041_h in pp Mode)

6.1.3 Object 6062_h: Position Demand Value

This object provides the demanded position value. The value is given in microsteps. Object 6062_h indicates the actual position of the motor. Do not confuse it with objects 6063_h and 6064_h.

Object Description			
Index	Name	Object Type	Data Type
6062_h	Position Demand Value	Variable	SIGNED32

Table 237: Object Description (6062_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	yes	SIGNED32	no

Table 238: Entry Description (6062_h)

6.1.4 Object 6063_h: Position Actual Internal Value

This object provides the actual value of the encoder or the motor. Use the sensor selection object $606A_h$ (see Section 5.1.8) for selecting the motor or the encoder first. Object 6063_h indicates the actual position of the encoder or the motor, rescaled to the microstep resolution. The value is given in microsteps.

Object Description			
Index	Name	Object Type	Data Type
6063_h	Position Actual Internal Value	Variable	SIGNED32

Table 239: Object Description (6063_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	yes	SIGNED32	no

Table 240: Entry Description (6063_h)

6.1.5 Object 6064_h: Position Actual Value

This object provides the actual value of the position measurement device. It always contains the same value as object 6063_h.

Object Description			
Index	Name	Object Type	Data Type
6064_h	Position Actual Value	Variable	SIGNED32

Table 241: Object Description (6064_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	yes	SIGNED32	no

Table 242: Entry Description (6064_h)

6.1.6 Object 6065_h: Following Error Window

This object indicates the configured range of tolerated position values symmetrically to the position demand value. If the position actual value is out of the following error window, a following error occurs. A following error may occur when a drive is blocked, unreachable profile velocity occurs, or at wrong closed-loop coefficients. The value shall be given in microsteps.

When the difference between motor position (object 6062_h) and encoder position (object 6063_h or 6064_h) is greater than the value set here, the motor is stopped and an emergency message is sent. Setting this object to zero turns off this feature completely.

Note Setting this object to a too low value leads to false alarms.

Object Description			
Index	Name	Object Type	Data Type
6065_h	Following Error Window	Variable	UNSIGNED32

Table 243: Object Description (6065_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...2147483647	0

Table 244: Entry Description (6065_h)

6.1.7 Object 6067_h: Position Window

This object indicates the configured symmetrical range of accepted positions relative to the target position. If the actual value of the position encoder is within the position window, this target position is regarded as reached. The value is given in increments. If the value of the position window is $FFFFFFFF_h$, the position window control is switched off. If this object is set to zero, the target reached event is signaled when the demand position (6062_h) has reached the target position (6064_h). When the position window is set to a value greater than zero, the target reached event is signaled when the actual encoder position value (6064_h) is within $(target\_position - position\_window)$ and $(target\_position + position\_window)$.

Object Description			
Index	Name	Object Type	Data Type
6067_h	Position Window	Variable	UNSIGNED32

Table 245: Object Description (6067_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	UNSIGNED32	$FFFFFFFF_h$

Table 246: Entry Description (6067_h)

6.1.8 Object 6068_h: Position Window Time

This object indicates the configured time, during which the actual position within the position window is measured. The value is given in ms. If this object is set to a value greater than zero and also the position window (6067_h) is set to a value greater than zero, the target reached event is not signaled until the actual position (6064_h) is at least as many milliseconds within the position window as defined by this object.

Object Description			
Index	Name	Object Type	Data Type
6068_h	Position Window Time	Variable	UNSIGNED16

Table 247: Object Description (6068_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	UNSIGNED16	0

Table 248: Entry Description (6068_h)

6.1.9 Object $606C_h$: Velocity Actual Value

This object shows the actual velocity value of the motor. The value is given in internal or user-defined velocity units (depending on object $208C_h$, described in Section 4.2.19).

Object Description			
Index	Name	Object Type	Data Type
$606C_h$	Velocity Actual Value	Variable	SIGNED32

Table 249: Object Description ($606C_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	yes	SIGNED32	no

Table 250: Entry Description ($606C_h$)

6.1.10 Object $607A_h$: Target Position

The target position is the position the drive should move to in profile position mode using the current settings of motion control parameters (such as velocity, acceleration, deceleration, motion profile type etc.). The value of this object is interpreted as absolute or relative depending on the ABS/REL flag in the controlword. It is given in microsteps.

Object Description			
Index	Name	Object Type	Data Type
$607A_h$	Target Position	Variable	SIGNED32

Table 251: Object Description ($607A_h$ in pp Mode)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	SIGNED32	0

Table 252: Entry Description ($607A_h$ in pp Mode)

6.1.11 Object $607D_h$: Software Position Limit

This object indicates the configured maximal and minimal software position limits. These parameters define the absolute position limits for the position demand value and position actual value. Every new target position is checked against these limits. The limit positions are always relative to the machine home position. Before being compared with the target position, they are corrected internally by the home offset as follows:

$$Corrected\_min\_position\_limit = min\_position\_limit - home\_offset$$
$$Corrected\_max\_position\_limit = max\_position\_limit - home\_offset$$

Object Description			
Index	Name	Object Type	Data Type
$607D_h$	Software Position Limit	Array	SIGNED32

Table 253: Object Description ($607D_h$)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
1	Minimum Position Limit	rw	no	SIGNED32	-2147483648
2	Maximum Position Limit	rw	no	SIGNED32	2147483647

Table 254: Entry Description ($607D_h$)

6.1.12 Object 6081_h: Profile Velocity

This object indicates the configured velocity normally attained at the end of the acceleration ramp during a profiled motion and is valid for both directions of motion. The profile velocity is the maximum velocity used when driving to a new position. It is given in internal or user specific units (depending on object $208C_h$, Section 4.2.19).

Object Description			
Index	Name	Object Type	Data Type
6081_h	Profile Velocity	Variable	UNSIGNED32

Table 255: Object Description (6081_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	UNSIGNED32	0

Table 256: Entry Description (6081_h)

6.1.13 Object 6082_h: End Velocity

This object indicates the configured velocity normally attained at the end of the deceleration ramp during a profiled motion and is valid for both directions of motion. The end velocity is the velocity used when reaching the new position. It is given in internal or user specific units (depending on object $208C_h$, Section 4.2.19).

Object Description			
Index	Name	Object Type	Data Type
6082_h	End Velocity	Variable	UNSIGNED32

Table 257: Object Description (6082_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	UNSIGNED32	0

Table 258: Entry Description (6082_h)

6.1.14 Object 6083_h: Profile Acceleration

This object indicates the configured acceleration. Object 6083_h sets the maximum acceleration to be used in profile position and profile velocity mode.
The units for object 6083_h can be chosen with object $208E_h$, described in Section 4.2.20.
In profile velocity mode, this object also sets the deceleration to be used (the deceleration ramp is always the same as the acceleration ramp in pv mode).

Object Description			
Index	Name	Object Type	Data Type
6083_h	Profile Acceleration	Variable	UNSIGNED32

Table 259: Object Description (6083_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	UNSIGNED32	0

Table 260: Entry Description (6083_h)

6.1.15 Object 6084_h: Profile Deceleration

This object indicates the configured deceleration. Object 6084_h sets the maximum deceleration to be used in profile positioning mode.
The units for object 6084_h can be chosen with object $208E_h$, described in Section 4.2.20.

Object Description			
Index	Name	Object Type	Data Type
6084_h	Profile Deceleration	Variable	UNSIGNED32

Table 261: Object Description (6084_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	UNSIGNED32	0

Table 262: Entry Description (6084_h)

6.1.16 Object 6085_h: Quick Stop Deceleration

This object indicates the configured deceleration used to stop the motor when the quick stop function is activated and the quick stop code object $605A_h$ is set to 2 (or 6). The value is given in the same unit as profile acceleration object 6083_h.

Object Description			
Index	Name	Object Type	Data Type
6085_h	Quick Stop Deceleration	Variable	UNSIGNED32

Table 263: Object Description (6085_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	UNSIGNED32	51200

Table 264: Entry Description (6085_h)

6.1.17 Object $60F2_h$: Positioning Option Code

This object indicates the positioning behavior in profile position mode. Only bits 0 and 1 (relative option) are supported.

Bit Definitions		
Bit 1	Bit 0	Definition
0	0	Positioning moves shall be performed relative to the preceding (internal absolute) target position.
0	1	Positioning moves shall be performed relative to the actual position demand value (object 6062_h).
1	0	Positioning moves shall be performed relative to the position actual value (object 6064_h).
1	1	reserved

Table 265: Bit Definitions of Object $60F2_h$

Object Description			
Index	Name	Object Type	Data Type
$60F2_h$	Positioning option code	Variable	UNSIGNED16

Table 266: Object Description ($60F2_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	UNSIGNED16	0

Table 267: Entry Description ($60F2_h$)

6.2 How to Move a Motor in pp Mode

Here is a small example that shows how to get a motor running in pp mode. In this example, assume that the module is reset (and then switched to pre-operational or operational) by NMT commands before. Note that the values are decimal.

- If there are no limit switches connected, first disable the limit switch inputs by writing 3 to object 2005_h.
- Select pp mode by writing 1 to object 6060_h.
- Write 6 to object 6040_h to switch to READY_TO_SWITCH_ON state.
- Write 7 to object 6040_h to switch to SWITCHED_ON state.
- Write 15 to object 6040_h to switch to OPERATION_ENABLED state.
- Write the desired target position (example: 500000) to object $607A_h$.
- Mark the new target position active by writing 31 to object 6040_h. The motor starts moving now.
- Reset the activation by writing 15 to object 6040_h (this can be done while the motor is still moving).

7 Profile Velocity Mode

The profile velocity mode is used to control the velocity of the drive without a special regard of the position. It contains limit functions and trajectory generation.

The profile velocity mode covers the following sub-functions:

- Demand value input through trajectory generator.
- 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 include:

- Profile velocity
- Profile acceleration
- Profile deceleration
- Emergency stop
- Motion profile type

7.1 Detailed Object Specifications

7.1.1 Object 6040_h: Controlword

This object indicates the received command controlling the power drive system finite state machine (PDS FSM). The CiA-402 state machine can be controlled using this object. See Figure 9 for detailed information.

In pv mode, the Controlword does not contain any operation mode specific bits.

Structure of the Controlword											
15	11	10	9	8	7	6	4	3	2	1	0
nu		r	r	h	fr	r		eo	qs	ev	so
MSB											LSB

Legend: nu = not used; r = reserved; h = halt; fr = fault reset; eo = enable operation; qs = quick stop; ev = enable voltage; so = switch on.

Table 268: Structure of the Controlword in pv Mode

Command Coding						
Command	Bits of Controlword					Transitions
	Bit 7	Bit 3	Bit 2	Bit 1	Bit 0	
Shutdown	0	x	1	1	0	2,6,8
Switch on	0	0	1	1	1	3
Switch on and enable operation	0	1	1	1	1	3, 4
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	0-to-1	x	x	x	x	15

Table 269: Command Coding

Object Description			
Index	Name	Object Type	Data Type
6040_h	Controlword	Variable	UNSIGNED16

Table 270: Object Description (6040_h in pv Mode)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	See command coding above.	

Table 271: Entry Description (6040_h in pv Mode)

7.1.2 Object 6041_h: Statusword

This object provides the status of the PDS FSA. It reflects the status of the CiA-402 state machine. See Figure 9 for detailed information. The object is structured as defined below.

For more information about the coding, refer to the CANopen drives and motion control device profile, part 2.

Structure of the Statusword															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
dir	mot	oms		ila	tr	rm	ms	w	sod	qs	ve	f	oe	so	rtso
MSB															LSB

Legend: nu = not used; r = reserved; oms = operation mode specific; h = halt; fr = fault reset; oe = operation enable; qs = quick stop; ve = voltage enable; so = switch on.

Table 272: Structure of the Statusword in pv Mode

Trinamic Specific Bits		
Bit	Name	Definition
14	Motor activity	0: Motor stands still. 1: Motor rotates.
15	Direction of rotation	This bit shows the direction of rotation.

Table 273: Trinamic Specific Bits

Operation Mode Specific Bits in pv Mode		
Bit	Name	Definition
10	Target reached	Indicates the target speed is reached.
12	Speed	Not supported.
13	Max. slippage error	Not supported.

Table 274: Operation Mode Specific Bits in pv Mode

State Coding	
Statusword	FSA State
xxxx xxxx x0xx 0000_h	Not ready to switch on
xxxx xxxx x1xx 0000_h	Switch on disabled
xxxx xxxx x01x 0001_h	Ready to switch on
xxxx xxxx x01x 0011_h	Switched on
xxxx xxxx x01x 0111_h	Operation enabled
xxxx xxxx x00x 0111_h	Quick stop active
xxxx xxxx x0xx 1111_h	Fault reaction active
xxxx xxxx x0xx 1000_h	Fault

Table 275: State Coding

Object Description			
Index	Name	Object Type	Data Type
6041_h	Controlword	Variable	UNSIGNED16

Table 276: Object Description (6041_h in pv Mode)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	See state coding above	

Table 277: Entry Description (6041_h in pv Mode)

7.1.3 Object 6062_h: Position Demand Value

This object provides the demanded position value. The value is given in microsteps. Object 6062_h indicates the actual position of the motor. Do not confuse it with objects 6063_h and 6064_h.

Object Description			
Index	Name	Object Type	Data Type
6062_h	Position Demand Value	Variable	SIGNED32

Table 278: Object Description (6062_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	yes	SIGNED32	no

Table 279: Entry Description (6062_h)

7.1.4 Object 6063_h: Position Actual Internal Value

This object provides the actual value of the encoder or the motor. Use the sensor selection object $606A_h$ (see Section 5.1.8) for selecting the motor or the encoder first. Object 6063_h indicates the actual position of the encoder or the motor, rescaled to the microstep resolution. The value is given in microsteps.

Object Description			
Index	Name	Object Type	Data Type
6063_h	Position Actual Internal Value	Variable	SIGNED32

Table 280: Object Description (6063_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	yes	SIGNED32	no

Table 281: Entry Description (6063_h)

7.1.5 Object 6064_h: Position Actual Value

This object provides the actual value of the position measurement device. It always contains the same value as object 6063_h.

Object Description			
Index	Name	Object Type	Data Type
6064_h	Position Actual Value	Variable	SIGNED32

Table 282: Object Description (6064_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	yes	SIGNED32	no

Table 283: Entry Description (6064_h)

7.1.6 Object 6065_h: Following Error Window

This object indicates the configured range of tolerated position values symmetrically to the position demand value. If the position actual value is out of the following error window, a following error occurs. A following error may occur when a drive is blocked, unreachable profile velocity occurs, or at wrong closed-loop coefficients. The value shall be given in microsteps.

When the difference between motor position (object 6062_h) and encoder position (object 6063_h or 6064_h) is greater than the value set here, the motor is stopped and an emergency message is sent. Setting this object to zero turns off this feature completely.

Note Setting this object to a too low value leads to false alarms.

Object Description			
Index	Name	Object Type	Data Type
6065_h	Following Error Window	Variable	UNSIGNED32

Table 284: Object Description (6065_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	0...2147483647	0

Table 285: Entry Description (6065_h)

7.1.7 Object $606C_h$: Velocity Actual Value

This object shows the actual velocity value of the motor. The value is given in internal or user-defined velocity units (depending on object $208C_h$, described in Section 4.2.19).

Object Description			
Index	Name	Object Type	Data Type
$606C_h$	Velocity Actual Value	Variable	SIGNED32

Table 286: Object Description ($606C_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	yes	SIGNED32	no

Table 287: Entry Description ($606C_h$)

7.1.8 Object $607D_h$: Software Position Limit

This object indicates the configured maximal and minimal software position limits. These parameters define the absolute position limits for the position demand value and position actual value. Every new target position is checked against these limits. The limit positions are always relative to the machine home position. Before being compared with the target position, they are corrected internally by the home offset as follows:

$$Corrected\_min\_position\_limit = min\_position\_limit - home\_offset$$
$$Corrected\_max\_position\_limit = max\_position\_limit - home\_offset$$

Object Description			
Index	Name	Object Type	Data Type
$607D_h$	Software Position Limit	Array	SIGNED32

Table 288: Object Description ($607D_h$)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
1	Minimum Position Limit	rw	no	SIGNED32	-2147483648
2	Maximum Position Limit	rw	no	SIGNED32	2147483647

Table 289: Entry Description ($607D_h$)

7.1.9 Object 6083_h: Profile Acceleration

This object indicates the configured acceleration. Object 6083_h sets the maximum acceleration to be used in profile position and profile velocity mode.
The units for object 6083_h can be chosen with object $208E_h$, described in Section 4.2.20.
In profile velocity mode, this object also sets the deceleration to be used (the deceleration ramp is always the same as the acceleration ramp in pv mode).

Object Description			
Index	Name	Object Type	Data Type
6083_h	Profile Acceleration	Variable	UNSIGNED32

Table 290: Object Description (6083_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	UNSIGNED32	0

Table 291: Entry Description (6083_h)

7.1.10 Object 6085_h: Quick Stop Deceleration

This object indicates the configured deceleration used to stop the motor when the quick stop function is activated and the quick stop code object $605A_h$ is set to 2 (or 6). The value is given in the same unit as profile acceleration object 6083_h.

Object Description			
Index	Name	Object Type	Data Type
6085_h	Quick Stop Deceleration	Variable	UNSIGNED32

Table 292: Object Description (6085_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	UNSIGNED32	51200

Table 293: Entry Description (6085_h)

7.1.11 Object $60FF_h$: Target Velocity

This object indicates the configured target velocity and is used as input for the trajectory generator. Object $60FF_h$ sets the target velocity when using profile velocity mode. The drive then accelerates or decelerates to that velocity using the acceleration and deceleration set by objects 6083_h and 6084_h. The values are given in units which can be selected with object $208C_h$, described in Section 4.2.19.

Object Description			
Index	Name	Object Type	Data Type
$60FF_h$	Target Velocity	Variable	SIGNED32

Table 294: Object Description ($60FF_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	SIGNED32	0

Table 295: Entry Description ($60FF_h$)

7.2 How to Move a Motor in pv Mode

Here is a small example that shows how to get a motor running in pv mode. In this example, assume that the module is reset (and then switched to pre-operational or operational) by NMT commands before.

- If there are no limit switches connected, first disable the limit switch inputs by writing 3 to object 2005_h.
- Select pv mode by writing 3 to object 6060_h.
- Write 6 to object 6040_h to switch to READY_TO_SWITCH_ON state.
- Write 7 to object 6040_h to switch to SWITCHED_ON state.
- Write 15 to object 6040_h to switch to OPERATION_ENABLED state.
- Write the desired target speed (example: 100000) to object $60FF_h$. The motor now accelerates to that speed.
- Stop the motor by writing 0 to object $60FF_h$.

8 Homing Mode

This section describes the method by which a drive seeks the home position (reference point). There are various methods of achieving this using limit switches at the ends of travel or a home switch in mid-travel. Some methods also use the index (zero) pulse train from an incremental encoder. It is possible to specify the speeds, acceleration, and the method of homing.

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

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

Figure 10 shows the defined input objects as well as the output objects. It is possible to specify the speeds, acceleration, and method of homing. The home offset object $607C_h$ allows to displace zero in the coordinate system from the home position.

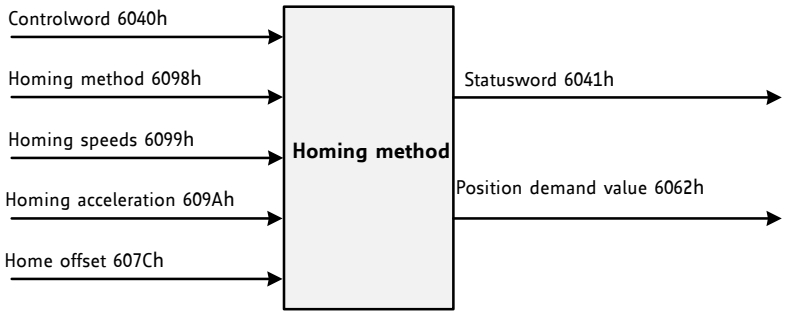


Figure 10: Homing Mode Function

Choosing a homing mode determines the following things:

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

The home position and the zero position are offset by the home offset (see object $607C_h$, Section 8.2.4).

Depending on the module, there are different sources of homing methods:

- Negative and positive limit switches
- Home switch
- Index pulse of an encoder

An exact knowledge of the absolute position is normally required to operate positioning drives. For cost reasons, drives often do not have an absolute encoder. So, a homing operation is necessary.

8.1 Homing Methods

The TMCM-3216 supports a subset of different standard CANopen homing methods. The homing method can be chosen through object 6098_h (Section 8.2.5).

Supported Homing Methods	
Method	Description
0	No homing (default value for object 6098_h).
1	Search the left end switch, then search the next encoder index pulse.
2	Search the right end switch, then search the next encoder index pulse.
3	Search the positive edge of the home switch, then search the next encoder index pulse.
5	Search the negative edge of the home switch, then search the next encoder index pulse.
17	Search the left end switch.
18	Search the right end switch.
19	Search the positive edge of the home switch.
21	Search the negative edge of the home switch.
33	Search next index pulse in negative direction.
34	Search next index pulse in positive direction.
35	The actual position is used as home position. All position values (objects 6062h, 6063h, and 6064h) are set to zero, but the motor does not move.

Table 296: Supported CANopen Homing Methods

When using homing methods that need end switch inputs or home switch inputs, take care of their configuration (object 2005_h, Section 4.2.6).

8.1.1 Homing Method 1: Homing on negative Limit Switch and Index Pulse

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

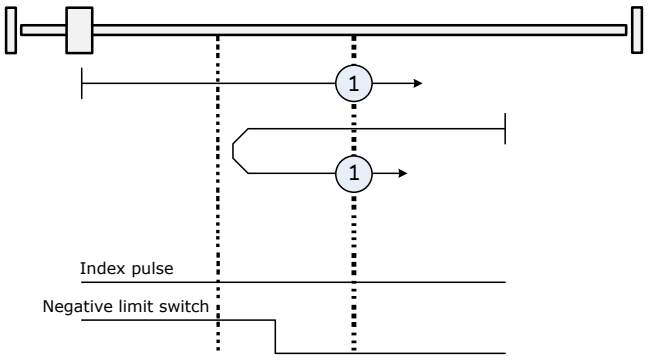


Figure 11: Homing Method 1

8.1.2 Homing Method 2: Homing on Positive Limit Switch and Index Pulse

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

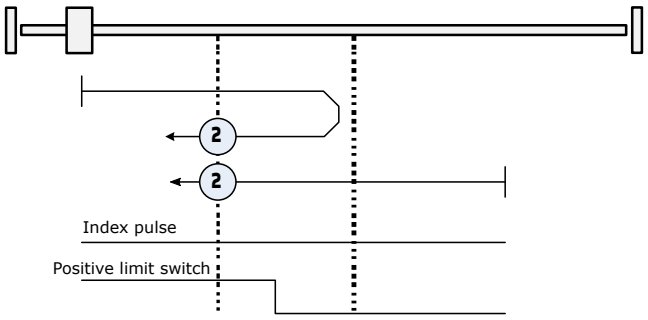


Figure 12: Homing Method 2

8.1.3 Homing Method 3: Homing on positive Home Switch and Index Pulse

Using this method, the initial direction of movement shall be dependent on the state of the home switch. The home position shall be 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 situated so that the direction of movement shall reverse during homing, the point at which the reversal takes place is anywhere after a change of state of the home switch.

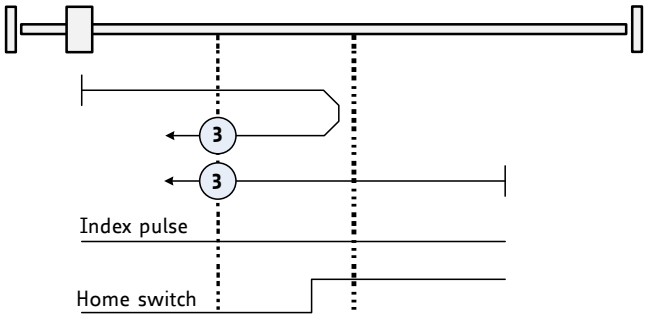


Figure 13: Homing Method 3

8.1.4 Homing Method 5: Homing on Negative Home Switch and Index Pulse

Using this method, the initial direction of movement shall be dependent on the state of the home switch. The home position shall be 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 situated so that the direction of movement shall reverse during homing, the point at which the reversal takes place is anywhere after a change of state of the home switch.

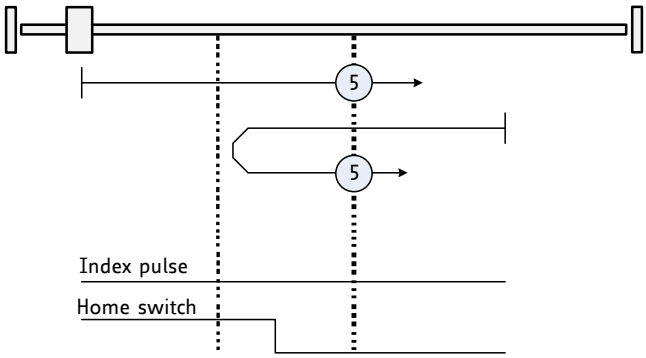


Figure 14: Homing Method 5

8.1.5 Homing Method 17: Homing on Negative Limit Switch

Using this method, the initial direction of movement shall be leftward if the negative limit switch is inactive (here: low). The home position shall be at the point where the negative limit switch becomes inactive.

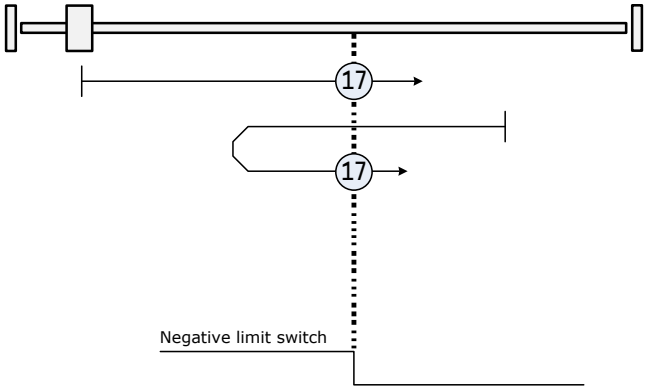


Figure 15: Homing Method 17

8.1.6 Homing Method 18: Homing on Positive Limit Switch

Using this method, the initial direction of movement shall be rightward if the positive limit switch is inactive (here: low). The home position shall be at the point where the positive limit switch becomes inactive.

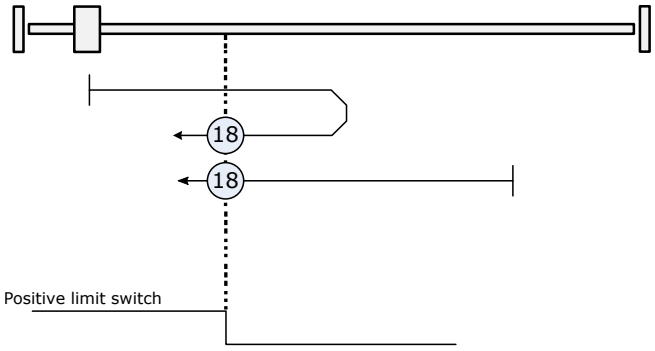


Figure 16: Homing Method 18

8.1.7 Homing Method 19: Homing on Positive Home Switch

Using this method, the initial direction of movement shall be dependent on the state of the home switch. The home position shall be at the point where the home switch changes state. If the initial direction of movement leads away from the home switch, the drive shall reverse on encountering the relevant limit switch.

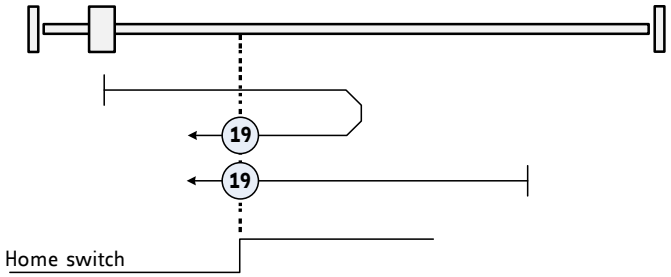


Figure 17: Homing Method 19

8.1.8 Homing Method 21: Homing on Negative Home Switch

Using this method, the initial direction of movement shall be dependent on the state of the home switch. The home position shall be at the point where the home switch changes state. If the initial direction of movement leads away from the home switch, the drive shall reverse on encountering the relevant limit switch.

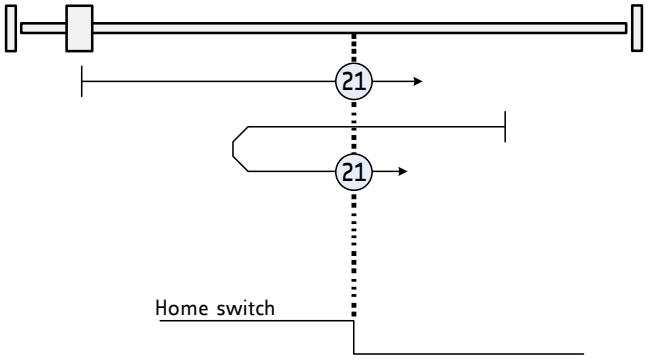


Figure 18: Homing Method 21

8.1.9 Homing Method 33 and 34: Homing on Next Index Pulse

Using these methods, the direction of homing is negative or positive respectively. The home position shall be at the index pulse found in the selected direction.

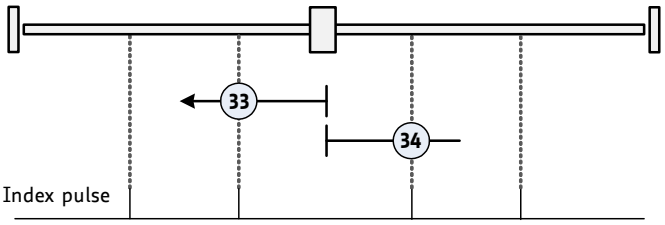


Figure 19: Homing Methods 33 and 34

8.1.10 Homing Method 35: Current Position as Home Position

In this method, the current position is considered the home position. This method does not require the drive device to be in operation enabled state.

8.2 Detailed Object Specifications

8.2.1 Object 6040_h: Controlword

This object indicates the received command controlling the power drive system finite state machine (PDS FSM). The CiA-402 state machine can be controlled using this object. See Figure 9 for detailed information.

Structure of the Controlword										
15 11	10	9	8	7	6 4	3	2	1	0	
nu	r	oms	h	fr	oms	eo	qs	ev	so	
MSB										LSB

Legend: nu = not used; r = reserved; oms = operation mode specific; h = halt; fr = fault reset; eo = enable operation; qs = quick stop; ev = enable voltage; so = switch on.

Table 297: Structure of the Controlword in hm Mode

Operation Mode Specific Bits in hm Mode		
Bit	Name	Definition
4	Homing operation start	1: start homing; 0: stop homing
8	Halt	Not supported.

Table 298: Operation Mode Specific Bits in hm Mode

Command Coding						
Command	Bits of Controlword					Transitions
	Bit 7	Bit 3	Bit 2	Bit 1	Bit 0	
Shutdown	0	x	1	1	0	2, 6, 8
Switch on	0	0	1	1	1	3
Switch on and enable operation	0	1	1	1	1	3, 4
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	0-to-1	x	x	x	x	15

Table 299: Command Coding

Object Description			
Index	Name	Object Type	Data Type
6040_h	Controlword	Variable	UNSIGNED16

Table 300: Object Description (6040_h in hm Mode)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	See command coding above.	

Table 301: Entry Description (6040_h in hm Mode)

8.2.2 Object 6041_h: Statusword

This object provides the status of the PDS FSA. It reflects the status of the CiA-402 state machine. See Figure 9 for detailed information. The object is structured as defined below.

For more information about the coding, refer to the CANopen drives and motion control device profile, part 2.

Structure of the Statusword															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
dir	mot	oms		ila	tr	rm	ms	w	sod	qs	ve	f	oe	so	rtso
MSB															LSB

Legend: nu = not used; r = reserved; oms = operation mode specific; h = halt; fr = fault reset; oe = operation enable; qs = quick stop; ve = voltage enable; s o =switch on.

Table 302: Structure of the Statusword in hm Mode

Trinamic Specific Bits		
Bit	Name	Definition
14	Motor activity	0: Motor stands still. 1: Motor rotates.
15	Direction of rotation	This bit shows the direction of rotation.

Table 303: Trinamic Specific Bits

Operation Mode Specific Bits in hm Mode		
Bit	Name	Definition
10	Target reached	Set when the zero position is found or homing is stopped by setting controlword bit 4 to zero.
12	Home attained	Set when zero position is found.
13	Homing error	Not supported.

Table 304: Operation Mode Specific Bits in hm Mode

State Coding	
Statusword	FSA State
xxxx xxxx x0xx 0000_h	Not ready to switch on
xxxx xxxx x1xx 0000_h	Switch on disabled
xxxx xxxx x01x 0001_h	Ready to switch on
xxxx xxxx x01x 0011_h	Switched on
xxxx xxxx x01x 0111_h	Operation enabled
xxxx xxxx x00x 0111_h	Quick stop active
xxxx xxxx x0xx 1111_h	Fault reaction active
xxxx xxxx x0xx 1000_h	Fault

Table 305: State Coding

Object Description			
Index	Name	Object Type	Data Type
6041_h	Controlword	Variable	UNSIGNED16

Table 306: Object Description (6041_h in hm Mode)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	See state coding above.	

Table 307: Entry Description (6041_h in hm Mode)

8.2.3 Object $606C_h$: Velocity Actual Value

This object shows the actual velocity value of the motor. The value is given in internal or user-defined velocity units (depending on object $208C_h$, described in Section 4.2.19).

Object Description			
Index	Name	Object Type	Data Type
$606C_h$	Velocity Actual Value	Variable	SIGNED32

Table 308: Object Description ($606C_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	yes	SIGNED32	no

Table 309: Entry Description ($606C_h$)

8.2.4 Object $607C_h$: Home Offset

This object indicates the configured difference between the zero position for the application and the machine home position/home switch (found during homing). While homing, the machine 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. The effect of setting the home position to a non-zero value depends on the selected homing method. The value of this object is given in microsteps. Negative values indicate the opposite direction.

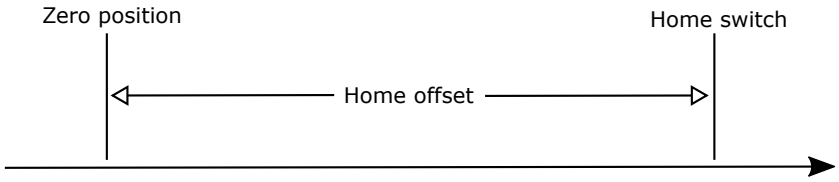


Figure 20: Home Offset

Object Description			
Index	Name	Object Type	Data Type
$607C_h$	Home Offset	Variable	SIGNED32

Table 310: Object Description ($607C_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	SIGNED32	0

Table 311: Entry Description ($607C_h$)

8.2.5 Object 6098_h: Homing Method

The homing method to be used can be selected by writing to this object. See Table 296 for a list of homing methods supported by the current version of the TMCM-3216 CANopen firmware.

Object Description			
Index	Name	Object Type	Data Type
6098_h	Homing Method	Variable	SIGNED8

Table 312: Object Description (6098_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	SIGNED8	0

Table 313: Entry Description (6098_h)

8.2.6 Object 6099_h: Homing Speeds

This object indicates the configured speeds used during homing procedure. The values are given in pps units or internal units selectable with object $208C_h$ (Section 4.2.19). Using object 6099_h, a fast and a slow homing speed can be set. In most homing modes, the home switch is searched with the fast speed first. When the home switch is found, the motor is decelerated to the slow speed (using the homing acceleration, object $609A_h$) to search for the exact switch point. When the switch point is found, the motor is stopped at that point.

Object Description			
Index	Name	Object Type	Data Type
6099_h	Homing Speeds	Array	UNSIGNED32

Table 314: Object Description (6099_h)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
1	Fast homing speed	rw	no	UNSIGNED32	0
2	Slow homing speed	rw	no	UNSIGNED32	0

Table 315: Entry Description (6099_h)

8.2.7 Object $609A_h$: Homing Acceleration

This object indicates the configured acceleration and deceleration to be used during homing operation. The value is given in units selected by object $208E_h$ (Section 4.2.20).

Object Description			
Index	Name	Object Type	Data Type
$609A_h$	Homing Acceleration	Variable	UNSIGNED32

Table 316: Object Description ($609A_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	UNSIGNED32	0

Table 317: Entry Description ($609A_h$)

8.3 How to Start a Homing in hm Mode

Here is a small example that shows how to home the motor in hm mode. In this example, assume that the module is reset (and then switched to pre-operational or operational) by NMT commands before. The home switch must be connected to the home switch input. It can be operated manually.

- Select hm mode by writing 6 to object 6060_h.
- Write 6 to object 6040_h to switch to READY_TO_SWITCH_ON state.
- Write 7 to object 6040_h to switch to SWITCHED_ON state.
- Write 15 to object 6040_h to switch to OPERATION_ENABLED state.
- Select homing method 19 by writing 19 to object 6098_h.
- Set the homing speeds by writing, example, 50000 to object 6099_h sub index 1 and, example, 10000 to object 6099_h sub index 2.
- Write 31 to object 6040_h to start the homing process.
- Press and release the home switch.
- When homing has finished, write 15 to object 6040_h again.

9 Cyclic Synchronous Position Mode

The cyclic synchronous position mode is used to directly control the position of the motor. It contains limit functions, but not a trajectory generator. The trajectory generator is located in the control device (the manager), not in the drive device. In cyclic synchronous manner, the control device provides a target position to the drive device, which performs position control, velocity control, and torque control.

The main control parameters are the target position (object $607A_h$, see Section 9.1.7) and the interpolation time period (object $60C2_h$, see Section 9.1.10). The drive automatically sets the velocity in such a manner that the next target position is reached within the interpolation time period. Acceleration and deceleration ramps are not used in this mode.

The cyclic synchronous position mode covers the following sub-functions:

- Position demand value input directly through an object.
- Monitoring of the position.
- Limiting the position using the software limits or the hardware limit switches.

9.1 Detailed Object Specifications

9.1.1 Object 6040_h: Controlword

This object indicates the received command controlling the power drive system finite state machine (PDS FSM). The CiA-402 state machine can be controlled using this object. See Figure 9 for detailed information. The cyclic synchronous position mode does not use any mode specific bits of the Controlword.

Structure of the Controlword									
15	9	8	7	6	4	3	2	1	0
nu		h	fr	nu		eo	qs	ev	so
MSB									LSB

Legend: nu = not used; h = halt; fr = fault reset; eo = enable operation; qs = quick stop; ev = enable voltage; so = switch on.

Table 318: Structure of the Controlword in csp Mode

Command Coding						
Command	Bits of Controlword					Transitions
	Bit 7	Bit 3	Bit 2	Bit 1	Bit 0	
Shutdown	0	x	1	1	0	2, 6, 8
Switch on	0	0	1	1	1	3
Switch on and enable operation	0	1	1	1	1	3, 4
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	0-to-1	x	x	x	x	15

Table 319: Command Coding

Object Description			
Index	Name	Object Type	Data Type
6040_h	Controlword	Variable	UNSIGNED16

Table 320: Object Description (6040_h in csp Mode)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	See command coding above.	

Table 321: Entry Description (6040_h in csp Mode)

9.1.2 Object 6041_h: Statusword

This object provides the status of the PDS FSA. It reflects the status of the CiA-402 state machine. See Figure 9 for detailed information. The object is structured as defined below.
For more information about the coding, refer to the CANopen drives and motion control device profile, part 2.

Structure of the Statusword															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
dir	mot	oms		ila	r	rm	ms	w	sod	qs	ve	f	oe	so	rtso
MSB															LSB

Legend: nu = not used; r = reserved; oms = operation mode specific; h = halt; fr = fault reset; oe = operation enable; qs = quick stop; ve = voltage enable; so = switch on.

Table 322: Structure of the Statusword in csp Mode

Trinamic Specific Bits		
Bit	Name	Definition
14	Motor activity	0: Motor stands still. 1: Motor rotates.
15	Direction of rotation	This bit shows the direction of rotation.

Table 323: Trinamic Specific Bits

Operation Mode Specific Bits in csp Mode		
Bit	Name	Definition
10	Reserved	Not used.
12	Target position ignored	0: Target position ignored. 1: Target position used as input to position controller.
13	Following error	0: No following error. 1: Following error.

Table 324: Operation Mode Specific Bits in csp Mode

State Coding	
Statusword	FSA State
xxxx xxxx x0xx 0000_h	Not ready to switch on
xxxx xxxx x1xx 0000_h	Switch on disabled
xxxx xxxx x01x 0001_h	Ready to switch on
xxxx xxxx x01x 0011_h	Switched on
xxxx xxxx x01x 0111_h	Operation enabled
xxxx xxxx x00x 0111_h	Quick stop active
xxxx xxxx x0xx 1111_h	Fault reaction active
xxxx xxxx x0xx 1000_h	Fault

Table 325: State Coding

Object Description			
Index	Name	Object Type	Data Type
6041_h	Controlword	Variable	UNSIGNED16

Table 326: Object Description (6041_h in csp Mode)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	See state coding above	

Table 327: Entry Description (6041_h in csp Mode)

9.1.3 Object 6062_h: Position Demand Value

This object provides the demanded position value. The value is given in microsteps. Object 6062_h indicates the actual position of the motor. Do not confuse it with objects 6063_h and 6064_h.

Object Description			
Index	Name	Object Type	Data Type
6062_h	Position Demand Value	Variable	SIGNED32

Table 328: Object Description (6062_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	yes	SIGNED32	no

Table 329: Entry Description (6062_h)

9.1.4 Object 6063_h: Position Actual Internal Value

This object provides the actual value of the encoder or the motor. Use the sensor selection object $606A_h$ (see Section 5.1.8) for selecting the motor or the encoder first. Object 6063_h indicates the actual position of the encoder or the motor, rescaled to the microstep resolution. The value is given in microsteps.

Object Description			
Index	Name	Object Type	Data Type
6063_h	Position Actual Internal Value	Variable	SIGNED32

Table 330: Object Description (6063_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	yes	SIGNED32	no

Table 331: Entry Description (6063_h)

9.1.5 Object 6064_h: Position Actual Value

This object provides the actual value of the position measurement device. It always contains the same value as object 6063_h.

Object Description			
Index	Name	Object Type	Data Type
6064_h	Position Actual Value	Variable	SIGNED32

Table 332: Object Description (6064_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	yes	SIGNED32	no

Table 333: Entry Description (6064_h)

9.1.6 Object $606C_h$: Velocity Actual Value

This object shows the actual velocity value of the motor. The value is given in internal or user-defined velocity units (depending on object $208C_h$, described in Section 4.2.19).

Object Description			
Index	Name	Object Type	Data Type
$606C_h$	Velocity Actual Value	Variable	SIGNED32

Table 334: Object Description ($606C_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	yes	SIGNED32	no

Table 335: Entry Description ($606C_h$)

9.1.7 Object $607A_h$: Target Position

The target position is the position the drive should move to in cyclic synchronous position mode using the current interpolation time period. In csp mode, this value is always interpreted as an absolute value.

Object Description			
Index	Name	Object Type	Data Type
$607A_h$	Target Position	Variable	SIGNED32

Table 336: Object Description ($607A_h$ in csp Mode)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	SIGNED32	0

Table 337: Entry Description ($607A_h$ in csp Mode)

9.1.8 Object $607D_h$: Software Position Limit

This object indicates the configured maximal and minimal software position limits. These parameters define the absolute position limits for the position demand value and position actual value. Every new target position is checked against these limits. The limit positions are always relative to the machine home position. Before being compared with the target position, they are corrected internally by the home offset as follows:

$$Corrected\_min\_position\_limit = min\_position\_limit - home\_offset$$
$$Corrected\_max\_position\_limit = max\_position\_limit - home\_offset$$

Object Description			
Index	Name	Object Type	Data Type
$607D_h$	Software Position Limit	Array	SIGNED32

Table 338: Object Description ($607D_h$)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
1	Minimum Position Limit	rw	no	SIGNED32	-2147483648
2	Maximum Position Limit	rw	no	SIGNED32	2147483647

Table 339: Entry Description ($607D_h$)

9.1.9 Object $60B0_h$: Position Offset

This object provides an offset to the target position (object $607A_h$, see Section 9.1.7). The value is given in microsteps and is added to the target position.

Object Description			
Index	Name	Object Type	Data Type
$60B0_h$	Position Offset	Variable	SIGNED32

Table 340: Object Description ($60B0_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	-2147483648...2147483647	0

Table 341: Entry Description ($60B0_h$)

9.1.10 Object $60C2_h$: Interpolation Time Period

This object indicates the interpolation cycle time. The interpolation time period (sub-index 01_h) is given in $10^{interpolation\_time\_index}$ s. The interpolation time index (sub-index 02_h) is dimensionless.

Object Description			
Index	Name	Object Type	Data Type
$60C2_h$	Interpolation Time Period	Record	Interpolation time period record (0080_h)

Table 342: Object Description ($60C2_h$)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
0	Highest sub-index supported	ro	no	UNSIGNED8	2
1	Interpolation Time Period Value	rw	no	UNSIGNED8	1
2	Interpolation Time Index	rw	no	-3...3	-3

Table 343: Entry Description ($60C2_h$)

10 Cyclic Synchronous Velocity Mode

The cyclic synchronous velocity mode is used to directly control the velocity of the motor. It contains limit functions, but not a trajectory generator. The trajectory generator is located in the control device (the manager), not in the drive device. In cyclic synchronous manner, the control device provides a target velocity to the drive device, which performs position control, velocity control, and torque control.

The main control parameters are the target velocity (object $60FF_h$, see Section 10.1.4) and the interpolation time period (object $60C2_h$, see Section 10.1.7). The drive automatically sets the acceleration in such a manner that the next target velocity is reached within the interpolation time period. Acceleration and deceleration ramps are not used in this mode.

The cyclic synchronous velocity mode covers the following sub-functions:

- Velocity demand value input directly through an object.
- Monitoring of the position.
- Limiting the position using the software limits or hardware limit switches.

10.1 Detailed Object Specifications

10.1.1 Object 6040_h: Controlword

This object indicates the received command controlling the power drive system finite state machine (PDS FSM). The CiA-402 state machine can be controlled using this object. See Figure 9 for detailed information. The cyclic synchronous velocity mode does not use any mode specific bits of the Controlword.

Structure of the Controlword									
15	9	8	7	6	4	3	2	1	0
nu		h	fr	nu		eo	qs	ev	so
MSB									LSB

Legend: nu = not used; h = halt; fr = fault reset; eo = enable operation; qs = quick stop; ev = enable voltage; so = switch on.

Table 344: Structure of the Controlword in csv Mode

Command Coding						
Command	Bits of Controlword					Transitions
	Bit 7	Bit 3	Bit 2	Bit 1	Bit 0	
Shutdown	0	x	1	1	0	2, 6, 8
Switch on	0	0	1	1	1	3
Switch on and enable operation	0	1	1	1	1	3, 4
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	0-to-1	x	x	x	x	15

Table 345: Command Coding

Object Description			
Index	Name	Object Type	Data Type
6040_h	Controlword	Variable	UNSIGNED16

Table 346: Object Description (6040_h in csv Mode)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	See command coding above.	

Table 347: Entry Description (6040_h in csv Mode)

10.1.2 Object 6041_h: Statusword

This object provides the status of the PDS FSA. It reflects the status of the CiA-402 state machine. See Figure 9 for detailed information. The object is structured as defined below.

For more information about the coding, refer to the CANopen drives and motion control device profile, part 2.

Structure of the Statusword															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
dir	mot	oms		ila	r	rm	ms	w	sod	qs	ve	f	oe	so	rtso
MSB															LSB

Legend: nu = not used; r = reserved; oms = operation mode specific; h = halt; fr = fault reset; oe = operation enable; qs = quick stop; ve = voltage enable; so = switch on.

Table 348: Structure of the Statusword in csv Mode

Trinamic Specific Bits		
Bit	Name	Definition
14	Motor activity	0: Motor stands still. 1: Motor rotates.
15	Direction of rotation	This bit shows the direction of rotation.

Table 349: Trinamic Specific Bits

Operation Mode Specific Bits in csv Mode		
Bit	Name	Definition
10	Reserved	Not used.
12	Target position ignored	0: Target velocity ignored. 1: Target velocity used as input to velocity controller.
13	Reserved	Not used.

Table 350: Operation Mode Specific Bits in csv Mode

State Coding	
Statusword	FSA State
xxxx xxxx x0xx 0000_h	Not ready to switch on
xxxx xxxx x1xx 0000_h	Switch on disabled
xxxx xxxx x01x 0001_h	Ready to switch on
xxxx xxxx x01x 0011_h	Switched on
xxxx xxxx x01x 0111_h	Operation enabled
xxxx xxxx x00x 0111_h	Quick stop active
xxxx xxxx x0xx 1111_h	Fault reaction active
xxxx xxxx x0xx 1000_h	Fault

Table 351: State Coding

Object Description			
Index	Name	Object Type	Data Type
6041_h	Controlword	Variable	UNSIGNED16

Table 352: Object Description (6041_h in csv Mode)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	See state coding above	

Table 353: Entry Description (6041_h in csv Mode)

10.1.3 Object $606C_h$: Velocity Actual Value

This object shows the actual velocity value of the motor. The value is given in internal or user-defined velocity units (depending on object $208C_h$, described in Section 4.2.19).

Object Description			
Index	Name	Object Type	Data Type
$606C_h$	Velocity Actual Value	Variable	SIGNED32

Table 354: Object Description ($606C_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	yes	SIGNED32	no

Table 355: Entry Description ($606C_h$)

10.1.4 Object $60FF_h$: Target Velocity

In csv mode, the target velocity specifies the velocity to be reached within the interpolation time period. The values are given in units which can be selected with object $208C_h$, described in Section 4.2.19.

Object Description			
Index	Name	Object Type	Data Type
$60FF_h$	Target Velocity	Variable	SIGNED32

Table 356: Object Description ($60FF_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	SIGNED32	0

Table 357: Entry Description ($60FF_h$)

10.1.5 Object $607D_h$: Software Position Limit

This object indicates the configured maximal and minimal software position limits. These parameters define the absolute position limits for the position demand value and position actual value. Every new target position is checked against these limits. The limit positions are always relative to the machine home position. Before being compared with the target position, they are corrected internally by the home offset as follows:

$$Corrected\_min\_position\_limit = min\_position\_limit - home\_offset$$
$$Corrected\_max\_position\_limit = max\_position\_limit - home\_offset$$

Object Description			
Index	Name	Object Type	Data Type
$607D_h$	Software Position Limit	Array	SIGNED32

Table 358: Object Description ($607D_h$)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
1	Minimum Position Limit	rw	no	SIGNED32	-2147483648
2	Maximum Position Limit	rw	no	SIGNED32	2147483647

Table 359: Entry Description ($607D_h$)

10.1.6 Object $60B1_h$: Velocity Offset

This object provides an offset to the target velocity (object $60FF_h$, see Section 10.1.4)). The value is added to the target velocity.

Object Description			
Index	Name	Object Type	Data Type
$60B1_h$	Velocity Offset	Variable	SIGNED32

Table 360: Object Description ($60B1_h$)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	yes	-2147483648...2147483647	0

Table 361: Entry Description ($60B1_h$)

10.1.7 Object $60C2_h$: Interpolation Time Period

This object indicates the interpolation cycle time. The interpolation time period (sub-index 01_h) is given in $10^{interpolation\_time\_index}$ s. The interpolation time index (sub-index 02_h) is dimensionless.

Object Description			
Index	Name	Object Type	Data Type
$60C2_h$	Interpolation Time Period	Record	Interpolation time period record (0080_h)

Table 362: Object Description ($60C2_h$)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
0	Highest sub-index supported	ro	no	UNSIGNED8	2
1	Interpolation Time Period Value	rw	no	UNSIGNED8	1
2	Interpolation Time Index	rw	no	-3...3	-3

Table 363: Entry Description ($60C2_h$)

11 Emergency Messages

The module sends an emergency (EMCY) message if an error occurs. The message contains information about the error type. The module can map internal errors and object 1001_h (error register) is part of every emergency object.

11.1 Format of Emergency Messages

The COB-ID of an emergency (EMCY) message is 80_h+ID of the node. So, for example, it is 81_h for node #1. The first two bytes contain the error code with LSB first. The third data byte contains the content of the error register (object 1001_h). The other five bytes can contain additional information, depending on the error code.

EMCY								
COB-ID	1	2	3	4	5	6	7	8
$0x80_h$+Node ID	Error code		Error register	Additional bytes				
	LSB	MSB	(1001_h)	#1…#5				

Table 364: Format of EMCY Messages

11.2 Error Codes Used by the TMCM-3216 Module

The following table shows all EMCY error codes used by the TMCM-3216 module.
Note that the additional byte #2 shows which motor is affected.

Emergency Messages (EMCY) of the TMCM-3216						
	Additional byte					
Error code	1	2	3	4	5	Description
0000_h	0	0…2	0	0	0	**Fault reset** The fault reset command has been executed.
1000_h	1	0…2	0	0	0	**Generic error: open load bridge A** The motor driver indicates open load on bridge A. It is possible that the motor cable is broken or that there is an error in the power amplifier itself.
1000_h	2	0…2	0	0	0	**Generic error: open load bridge B** The motor driver indicates open load on bridge B. It is possible that the motor cable is broken or that there is an error in the power amplifier itself.
2310_h	0	0…2	0	0	0	**Overcurrent high side** The motor driver indicates an overcurrent on the high side. This can be caused by a short circuit in the driver stage.
2311_h	0	0…2	0	0	0	**Overcurrent bridge B** The motor driver indicates that there is overcurrent on bridge B. This can be caused by a short circuit in the motor itself or in the motor driver stage.

	Additional byte					
Error code	1	2	3	4	5	Description
2312_h	0	0…2	0	0	0	**Overcurrent bridge A** The motor driver indicates that there is overcurrent on bridge A. This can be caused by a short circuit in the motor itself or in the motor driver stage.
3230_h	0	0…2	0	0	0	**stallGuard2 error** The actual load value exceeds the stallGuard2 limit.
4310_h	1	0…2	0	0	0	**Overtemperature pre-warning** The temperature in the motor driver exceeds the pre-warning limit.
4310_h	2	0…2	0	0	0	**Overtemperature error** The motor driver has been switched off because the temperature limit has been exceeded.
5441_h	0	255	0	0	0	**Shutdown switch active** The enable signal is missing (due to the shutdown switch) and the motor driver has been switched off.
6320_h	0	255	0	0	0	**Parameter error** The data in the received PDO is either wrong or cannot be accepted due to the internal state of the drive.
8100_h	0	255	0	0	0	**Communication error** General CAN bus communication error.
8110_h	1	255	0	0	0	**CAN controller overflow** The receive message buffer of the CAN controller hardware is full and some CAN messages are lost.
8110_h	2	255	0	0	0	**CAN Tx buffer overflow** The software CAN transmit buffer is full and thus some CAN messages are lost.
8110_h	3	255	0	0	0	**CAN Rx buffer overflow** The software CAN receive buffer is full and so some CAN messages are lost.
8120_h	0	255	0	0	0	**CAN error passive** The CAN controller has detected communication errors and has entered the CAN Error passive state.
8130_h	0	255	0	0	0	**Heartbeat or lifeguard error** The module did not receive a heartbeat or lifeguard message in time.
8140_h	0	255	0	0	0	**CAN controller recovered from bus-off state** The CAN controller has detected too many errors and has changed into the bus-off state. The drive has been stopped and disabled. This message is sent after the CAN controller has recovered from bus-off state and is bus-on again.
8210_h	0	255	0	0	0	**PDO not processed due to length error** A PDO sent to the module cannot be processed because too few bytes are supplied.

	Additional byte					
Error code	1	2	3	4	5	Description
8220_h	0	255	0	0	0	**PDO length exceeded** A PDO sent to the module cannot be processed because too many bytes are supplied.
8611_h	0	0…2	0	0	0	**Following error** The deviation between motor position counter and encoder position counter has exceeded the following error window.
$ff00_h$	0	0…2	0	0	0	**Undervoltage** The supply voltage is too low to drive a motor.
$ff01_h$	1	0…2	0	0	0	**Positive software limit** The actual position is outside the range defined by object $607d_h$.
$ff01_h$	2	0…2	0	0	0	**Negative software limit** The actual position is outside the range defined by object $607d_h$.
$ff01_h$	3	0…2	0	0	0	**Positive limit switch** The positive limit switch has been touched outside of the homing function.
$ff01_h$	4	0…2	0	0	0	**Negative limit switch** The negative limit switch has been touched outside of the homing function.

Table 365: Emergency Messages (EMCY)

12 SDO Abort Codes

Trying to access an object through SDO read or SDO write may result in an error. In such a case, an SDO abort transfer message containing an abort code is sent. The following table lists all SDO abort codes defined by the CiA-301 standard. Not all of these are used by the TMCM-3216 module.

SDO Abort Codes	
Abort code	Description
05030000_h	Toggle bit not alternated.
05040000_h	SDO protocol timed out.
05040001_h	Client/server command specifier not valid or unknown.
05040002_h	Invalid block size.
05040003_h	Invalid sequence number.
05040004_h	CRC error.
05040005_h	Out of memory.
06010000_h	Unsupported access to an object.
06010001_h	Attempt to read a write only object.
06010002_h	Attempt to write a read only object.
06020000_h	Object does not exist in object dictionary.
06040041_h	Object cannot be mapped to the PDO.
06040042_h	The number and length of the objects to be mapped exceed the PDO length.
06040043_h	General parameter incompatibility reason.
06040047_h	General internal incompatibility in the device.
06060000_h	Access failed due to a hardware error.
06070010_h	Data type does not match, length of service parameter does not match.
06070012_h	Data type does not match, length of service parameter too high.
06070013_h	Data type does not match, length of service parameter too low.
06090011_h	Sub-index does not exist.
06090030_h	Invalid value for parameter.
06090031_h	Value of parameter too high.
06090032_h	Value of parameter too low.
06090036_h	Maximum value is less than minimum value.
$060A0023_h$	Resource not available.
08000000_h	General error.
08000020_h	Data cannot be transferred to or stored in the application.
08000021_h	Data cannot be transferred to or stored in the application because of local control.

Abort code	Description
08000022_h	Data cannot be transferred to or stored in the application because of the present device state.
08000023_h	Object dictionary dynamic generation failed or no object dictionary is present.
08000024_h	No data available.

Table 366: SDO Abort Codes

13 Figures Index

1 stallGuard2 Load Measurement as a Function of Load 9
2 Energy Efficiency Example with CoolStep 10
3 Typical Motion Profile with TRINAMIC EightPoint Motion Controller 11
4 NMT State Machine 15
5 Communication Architecture 16
6 Device Model 17
7 CoolStep Adjustment Points and Thresholds 35
8 Brake Output Timing 38
9 DS402 Finite State Machine 79
10 Homing Mode Function 99
11 Homing Method 1 100
12 Homing Method 2 101
13 Homing Method 3 101
14 Homing Method 5 102
15 Homing Method 17 102
16 Homing Method 18 102
17 Homing Method 19 103
18 Homing Method 21 103
19 Homing Methods 33 and 34 104
20 Home Offset 108

14 Tables Index

1 Abbreviations Used in this Manual . . 8
2 SixPoint Ramp Parameters 11
3 Service Primitives 13
4 Service Types 14
5 Object Dictionary 18
6 Multi-Axis Object Indices 19
7 Object Description (1000_h) 20
8 Entry Description (1000_h) 20
9 Object Description (1001_h) 20
10 Entry Description (1001_h) 20
11 Error Register Bits 21
12 Value Definition (1005_h) 21
13 Object Description (1005_h) 21
14 Entry Description (1005_h) 21
15 Object Description (1008_h) 22
16 Entry Description (1008_h) 22
17 Object Description (1009_h) 22
18 Entry Description (1009_h) 22
19 Object Description ($100A_h$) 22
20 Entry Description ($100A_h$) 23
21 Save Signature 23
22 Object Description (1010_h) 23
23 Entry Description (1010_h) 24
24 Load Signature 24
25 Object Description (1011_h) 25
26 Entry Description (1011_h) 25
27 Object Description (1014_h) 25
28 Entry Description (1014_h) 25
29 Object Description (1015_h) 26
30 Entry Description (1015_h) 26
31 Value Definition (1016_h) 26
32 Object Description (1016_h) 26
33 Entry Description (1016_h) 27
34 Object Description (1017_h) 27
35 Entry Description (1017_h) 27
36 Object Description (1018_h) 27
37 Entry Description (1018_h) 28
38 Object Description (1029_h) 28
39 Entry Description (1029_h) 28
40 Object Description (1400_h) 29
41 Entry Description (1400_h) 29
42 Object Description (1600_h) 30
43 Entry Description (1600_h) 30
44 Object Description (1800_h) 31
45 Entry Description (1800_h) 32
46 Object Description ($1A00_h$) 32
47 Entry Description ($1A00_h$) 33
48 Multi-Axis Object Indices (Manufacturer Specific Area) 34
49 CoolStep-Related Objects 36
50 Object Description (2000_h) 37
51 Entry Description (2000_h) 37
52 Object Description (2001_h) 37
53 Entry Description (2001_h) 37
54 Object Description (2002_h) 38
55 Entry Description (2002_h) 38
56 Object Description (2003_h) 38
57 Entry Description (2003_h) 38
58 Object Description (2004_h) 39
59 Entry Description (2004_h) 39
60 Object Description (2005_h) 39
61 Entry Description (2005_h) 39
62 Bit Definitions (2005_h) 40
63 Object Description ($200A_h$) 40
64 Entry Description $(200A)_h$ 40
65 Object Description ($200B_h$) 40
66 Entry Description ($200B_h$) 41
67 Object Description ($200C_h$) 41
68 Entry Description ($200C_h$) 41
69 Object Description (2010_h) 41
70 Entry Description (2010_h) 42
71 Object Description (2011_h) 42
72 Entry Description (2011_h) 42
73 Object Description (2012_h) 42
74 Entry Description (2012_h) 42
75 Object Description (2013_h) 43
76 Entry Description (2013_h) 43
77 Object Description (2015_h) 43
78 Entry Description (2015_h) 43
79 Object Description (2016_h) 44
80 Entry Description (2016_h) 44
81 Object Description (2017_h) 44
82 Entry Description (2017_h) 44
83 Object Description (2018_h) 44
84 Entry Description (2018_h) 45
85 Object Description (2089_h) 45
86 Entry Description (2089_h) 45
87 Object Description ($208C_h$) 45
88 Entry Description ($208C_h$) 46
89 Object Description ($208E_h$) 46
90 Entry Description ($208E_h$) 46
91 Object Description (2092_h) 46
92 Entry Description (2092_h) 46
93 Object Description (2093_h) 47
94 Entry Description (2093_h) 47
95 Object Description (2094_h) 47
96 Entry Description (2094_h) 47
97 Object Description (2095_h) 48
98 Entry Description (2095_h) 48
99 Object Description (2096_h) 48
100 Entry Description (2096_h) 48
101 Object Description (2097_h) 49
102 Entry Description (2097_h) 49
103 Object Description (2098_h) 49

104 Entry Description (2098_h) 49
105 Object Description (2099_h) 50
106 Entry Description (2099_h) 50
107 Object Description ($209A_h$) 50
108 Entry Description ($209A_h$) 50
109 Object Description ($209B_h$) 51
110 Entry Description ($209B_h$) 51
111 Object Description ($209C_h$) 51
112 Entry Description ($209C_h$) 51
113 Object Description ($209D_h$) 52
114 Entry Description ($209D_h$) 52
115 Object Description ($209E_h$) 52
116 Entry Description ($209E_h$) 52
117 Object Description ($20A1_h$) 52
118 Entry Description ($20A1_h$) 53
119 Object Description ($20A2_h$) 53
120 Entry Description ($20A2_h$) 53
121 Object Description ($20A3_h$) 53
122 Entry Description ($20A3_h$) 54
123 Object Description ($20A4_h$) 54
124 Entry Description ($20A4_h$) 54
125 Object Description ($20A5_h$) 54
126 Entry Description ($20A5_h$) 54
127 Object Description ($20A7_h$) 55
128 Entry Description ($20A7_h$) 55
129 Object Description ($20A8_h$) 55
130 Entry Description ($20A8_h$) 55
131 Object Description ($20A9_h$) 56
132 Entry Description ($20A9_h$) 56
133 Object Description ($20AA_h$) 56
134 Entry Description ($20AA_h$) 56
135 Object Description ($20B0_h$) 56
136 Entry Description ($20B0_h$) 57
137 Object Description ($20B1_h$) 57
138 Entry Description ($20B1_h$) 57
139 Object Description ($20B2_h$) 57
140 Entry Description ($20B2_h$) 57
141 Object Description ($20B3_h$) 58
142 Entry Description ($20B3_h$) 58
143 Object Description ($20B4_h$) 58
144 Entry Description ($20B4_h$) 58
145 Object Description ($20B5_h$) 59
146 Entry Description ($20B5_h$) 59
147 Object Description ($20B6_h$) 59
148 Entry Description ($20B6_h$) 59
149 Object Description ($20B7_h$) 59
150 Entry Description ($20B7_h$) 60
151 Object Description ($20B8_h$) 60
152 Entry Description ($20B8_h$) 60
153 Object Description ($20B9_h$) 60
154 Entry Description ($20B9_h$) 60
155 Object Description (2100_h) 61
156 Entry Description (2100_h) 61
157 Object Description (2101_h) 61
158 Entry Description (2101_h) 61
159 Driver Error Flags (2102_h) 62
160 Object Description (2102_h) 62
161 Entry Description (2102_h) 62
162 Object Description (2107_h) 62
163 Entry Description (2107_h) 63
164 Object Description ($210A_h$) 63
165 Entry Description ($210A_h$) 63
166 Object Description (2121_h) 63
167 Entry Description (2121_h) 63
168 Object Description (2122_h) 64
169 Entry Description (2122_h) 64
170 Bit Definitions (2702_h) 64
171 Object Description (2702_h) 64
172 Entry Description (2702_h) 64
173 Bit Definitions (2703_h) 65
174 Object Description (2703_h) 65
175 Entry Description (2703_h) 65
176 Object Description (2704_h) 66
177 Entry Description $(2704)_h$ 66
178 Object Description (2705_h) 66
179 Entry Description (2705_h) 66
180 Object Description (2706_h) 67
181 Entry Description (2706_h) 67
182 Object Description (2707_h) 67
183 Entry Description $(2707)_h$ 67
184 Object Description (2708_h) 68
185 Entry Description (2708_h) 68
186 Object Description ($270E_h$) 68
187 Entry Description ($270E_h$) 68
188 Value Description (2711_h) 69
189 Object Description (2711_h) 69
190 Entry Description (2711_h) 69
191 Multi-Axis Object Indices (Profile Specific Area) 70
192 Value Description ($605A_h$) 71
193 Object Description ($605A_h$) 71
194 Entry Description ($605A_h$) 71
195 Value Description ($605B_h$) 71
196 Object Description ($605B_h$) 71
197 Entry Description ($605B_h$) 72
198 Value Description ($605C_h$) 72
199 Object Description ($605C_h$) 72
200 Entry Description ($605C_h$) 72
201 Value Description ($605D_h$) 72
202 Object Description ($605D_h$) 73
203 Entry Description ($605D_h$) 73
204 Value Description ($605E_h$) 73
205 Object Description ($605E_h$) 73
206 Entry Description ($605E_h$) 73
207 Value Description (6060_h) 74
208 Object Description (6060_h) 74
209 Entry Description (6060_h) 74
210 Value Description (6061_h) 74

211 Object Description (6061_h) 75
212 Entry Description (6061_h) 75
213 Value Description ($606A_h$) 75
214 Object Description ($606A_h$) 75
215 Entry Description ($606A_h$) 75
216 Object Description ($608F_h$) 76
217 Entry Description ($608F_h$) 76
218 Object Description ($60FD_h$) 76
219 Entry Description ($60FD_h$) 76
220 Value Definition (6502_h) 77
221 Object Description (6502_h) 77
222 Entry Description (6502_h) 77
223 Value Definition ($67FF_h$) 78
224 Object Description ($67FF_h$) 78
225 Entry Description ($67FF_h$) 78
226 Structure of the Controlword in pp Mode 80
227 Operation Mode Specific Bits in pp Mode 80
228 Command Coding 80
229 Object Description (6040_h in pp Mode) 81
230 Entry Description (6040_h in pp Mode) 81
231 Structure of the Statusword in pp Mode 81
232 Trinamic Specific Bits 81
233 Operation Mode Specific Bits in pp Mode 82
234 State Coding 82
235 Object Description (6041_h in pp Mode) 82
236 Entry Description (6041_h in pp Mode) 82
237 Object Description (6062_h) 83
238 Entry Description (6062_h) 83
239 Object Description (6063_h) 83
240 Entry Description (6063_h) 83
241 Object Description (6064_h) 83
242 Entry Description (6064_h) 84
243 Object Description (6065_h) 84
244 Entry Description (6065_h) 84
245 Object Description (6067_h) 85
246 Entry Description (6067_h) 85
247 Object Description (6068_h) 85
248 Entry Description (6068_h) 85
249 Object Description ($606C_h$) 85
250 Entry Description ($606C_h$) 86
251 Object Description ($607A_h$ in pp Mode) 86
252 Entry Description ($607A_h$ in pp Mode) 86
253 Object Description ($607D_h$) 86
254 Entry Description ($607D_h$) 87
255 Object Description (6081_h) 87
256 Entry Description (6081_h) 87
257 Object Description (6082_h) 87
258 Entry Description (6082_h) 87
259 Object Description (6083_h) 88
260 Entry Description (6083_h) 88
261 Object Description (6084_h) 88
262 Entry Description (6084_h) 88
263 Object Description (6085_h) 89
264 Entry Description (6085_h) 89
265 Bit Definitions of Object $60F2_h$ 89
266 Object Description ($60F2_h$) 89
267 Entry Description ($60F2_h$) 89
268 Structure of the Controlword in pv Mode 91
269 Command Coding 92
270 Object Description (6040_h in pv Mode) 92
271 Entry Description (6040_h in pv Mode) 92
272 Structure of the Statusword in pv Mode 93
273 Trinamic Specific Bits 93
274 Operation Mode Specific Bits in pv Mode 93
275 State Coding 93
276 Object Description (6041_h in pv Mode) 94
277 Entry Description (6041_h in pv Mode) 94
278 Object Description (6062_h) 94
279 Entry Description (6062_h) 94
280 Object Description (6063_h) 94
281 Entry Description (6063_h) 95
282 Object Description (6064_h) 95
283 Entry Description (6064_h) 95
284 Object Description (6065_h) 95
285 Entry Description (6065_h) 96
286 Object Description ($606C_h$) 96
287 Entry Description ($606C_h$) 96
288 Object Description ($607D_h$) 96
289 Entry Description ($607D_h$) 97
290 Object Description (6083_h) 97
291 Entry Description (6083_h) 97
292 Object Description (6085_h) 97
293 Entry Description (6085_h) 97
294 Object Description ($60FF_h$) 98
295 Entry Description ($60FF_h$) 98
296 Supported CANopen Homing Methods 100
297 Structure of the Controlword in hm Mode 105
298 Operation Mode Specific Bits in hm Mode 105
299 Command Coding 105
300 Object Description (6040_h in hm Mode) 106
301 Entry Description (6040_h in hm Mode) 106
302 Structure of the Statusword in hm Mode 106
303 Trinamic Specific Bits 106
304 Operation Mode Specific Bits in hm Mode 107
305 State Coding 107
306 Object Description (6041_h in hm Mode) 107
307 Entry Description (6041_h in hm Mode) 107
308 Object Description ($606C_h$) 108
309 Entry Description ($606C_h$) 108
310 Object Description ($607C_h$) 108

311 Entry Description ($607C_h$) 108
312 Object Description (6098_h) 109
313 Entry Description (6098_h) 109
314 Object Description (6099_h) 109
315 Entry Description (6099_h) 109
316 Object Description ($609A_h$) 110
317 Entry Description ($609A_h$) 110
318 Structure of the Controlword in csp Mode 111
319 Command Coding 112
320 Object Description (6040_h in csp Mode) 112
321 Entry Description (6040_h in csp Mode) 112
322 Structure of the Statusword in csp Mode 112
323 Trinamic Specific Bits 113
324 Operation Mode Specific Bits in csp Mode 113
325 State Coding 113
326 Object Description (6041_h in csp Mode) 113
327 Entry Description (6041_h in csp Mode) 114
328 Object Description (6062_h) 114
329 Entry Description (6062_h) 114
330 Object Description (6063_h) 114
331 Entry Description (6063_h) 114
332 Object Description (6064_h) 115
333 Entry Description (6064_h) 115
334 Object Description ($606C_h$) 115
335 Entry Description ($606C_h$) 115
336 Object Description ($607A_h$ in csp Mode) 115
337 Entry Description ($607A_h$ in csp Mode) 116
338 Object Description ($607D_h$) 116
339 Entry Description ($607D_h$) 116
340 Object Description ($60B0_h$) 116
341 Entry Description ($60B0_h$) 117
342 Object Description ($60C2_h$) 117
343 Entry Description ($60C2_h$) 117
344 Structure of the Controlword in csv Mode 118
345 Command Coding 119
346 Object Description (6040_h in csv Mode) 119
347 Entry Description (6040_h in csv Mode) 119
348 Structure of the Statusword in csv Mode 120
349 Trinamic Specific Bits 120
350 Operation Mode Specific Bits in csv Mode 120
351 State Coding 120
352 Object Description (6041_h in csv Mode) 121
353 Entry Description (6041_h in csv Mode) 121
354 Object Description ($606C_h$) 121
355 Entry Description ($606C_h$) 121
356 Object Description ($60FF_h$) 121
357 Entry Description ($60FF_h$) 122
358 Object Description ($607D_h$) 122
359 Entry Description ($607D_h$) 122
360 Object Description ($60B1_h$) 122
361 Entry Description ($60B1_h$) 123
362 Object Description ($60C2_h$) 123
363 Entry Description ($60C2_h$) 123
364 Format of EMCY Messages 124
365 Emergency Messages (EMCY) 126
366 SDO Abort Codes 128
367 Firmware Revision 136
368 Document Revision 136

15 Supplemental Directives

15.1 Producer Information

15.2 Copyright

15.3 Trademark Designations and Symbols

Trademark designations and symbols used in this documentation indicate that a product or feature is owned and registered as trademark and/or patent either by ADI Trinamic or by other manufacturers, whose products are used or referred to in combination with ADI Trinamic's products and ADI Trinamic's product documentation.

This CANopen Firmware Manual is a non-commercial publication that seeks to provide concise scientific and technical user information to the target user. Thus, trademark designations and symbols are only entered in the Short Spec of this document that introduces the product at a quick glance. The trademark designation /symbol is also entered when the product or feature name occurs for the first time in the document. All trademarks and brand names used are property of their respective owners.

15.4 Target User

The documentation provided here, is for programmers and engineers only, who are equipped with the necessary skills and have been trained to work with this type of product.

The Target User knows how to responsibly make use of this product without causing harm to himself or others, and without causing damage to systems or devices, in which the user incorporates the product.

15.5 Disclaimer: Life Support Systems

ADI Trinamic/Trinamic Motion Control GmbH & Co. KG does not authorize or warrant any of its products for use in life support systems, without the specific written consent of ADI Trinamic/Trinamic Motion Control GmbH & Co. KG.

Life support systems are equipment intended to support or sustain life, and whose failure to perform, when properly used in accordance with instructions provided, can be reasonably expected to result in personal injury or death.

Information given in this document is believed to be accurate and reliable. However, no responsibility is assumed for the consequences of its use nor for any infringement of patents or other rights of third parties which may result from its use. Specifications are subject to change without notice.

15.6 Disclaimer: Intended Use

The data specified in this user manual is intended solely for the purpose of product description. No representations or warranties, either express or implied, of merchantability, fitness for a particular purpose

or of any other nature are made hereunder with respect to information/specification or the products to which information refers and no guarantee with respect to compliance to the intended use is given.

In particular, this also applies to the stated possible applications or areas of applications of the product. TRINAMIC products are not designed for and must not be used in connection with any applications where the failure of such products would reasonably be expected to result in significant personal injury or death (safety-Critical Applications) without ADI Trinamic's/Trinamic Motion Control GmbH & Co. KG specific written consent.

ADI Trinamic/Trinamic Motion Control GmbH & Co. KG products are not designed nor intended for use in military or aerospace applications or environments or in automotive applications unless specifically designated for such use by ADI Trinamic/Trinamic Motion Control GmbH & Co. KG.
ADI Trinamic/Trinamic Motion Control GmbH & Co. KG conveys no patent, copyright, mask work right or other trademark right to this product. ADI Trinamic/Trinamic Motion Control GmbH & Co. KG assumes no liability for any patent and/or other trademark rights of a third party resulting from processing or handling of the product and/or any other use of the product.

15.7 Collateral Documents and Tools

This product documentation is related and/or associated with additional tool kits, firmware and other items, as provided on the product page at: www.analog.com.

16 Revision History

16.1 Firmware Revision

Version	Date	Description
1.04	01/25	Initial release

Table 367: Firmware Revision

16.2 Document Revision

Version	Date	Description
0	01/25	Initial release

Table 368: Document Revision