



Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of "Quality Parts,Customers Priority,Honest Operation,and Considerate Service",our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!



Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832

Email & Skype: info@chipsmall.com Web: www.chipsmall.com

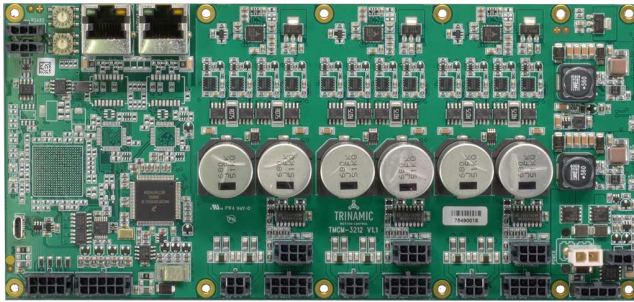
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China



TMCM-3213 CoE Firmware Manual

Firmware Version V1.01 | Document Revision V1.02 • 2016-NOV-29

The TMCM-3213 is a three axes controller/driver module for 2-phase bipolar stepper motors with separate differential encoder and separate home and stop switch inputs for each axis. Dynamic current control, and quiet, smooth and efficient operation are combined with stealthChop™, dcStep™, stallGuard™ and coolStep™ features. The module offers four analog or digital inputs as well as four digital outputs in combination with a break chopper unit.



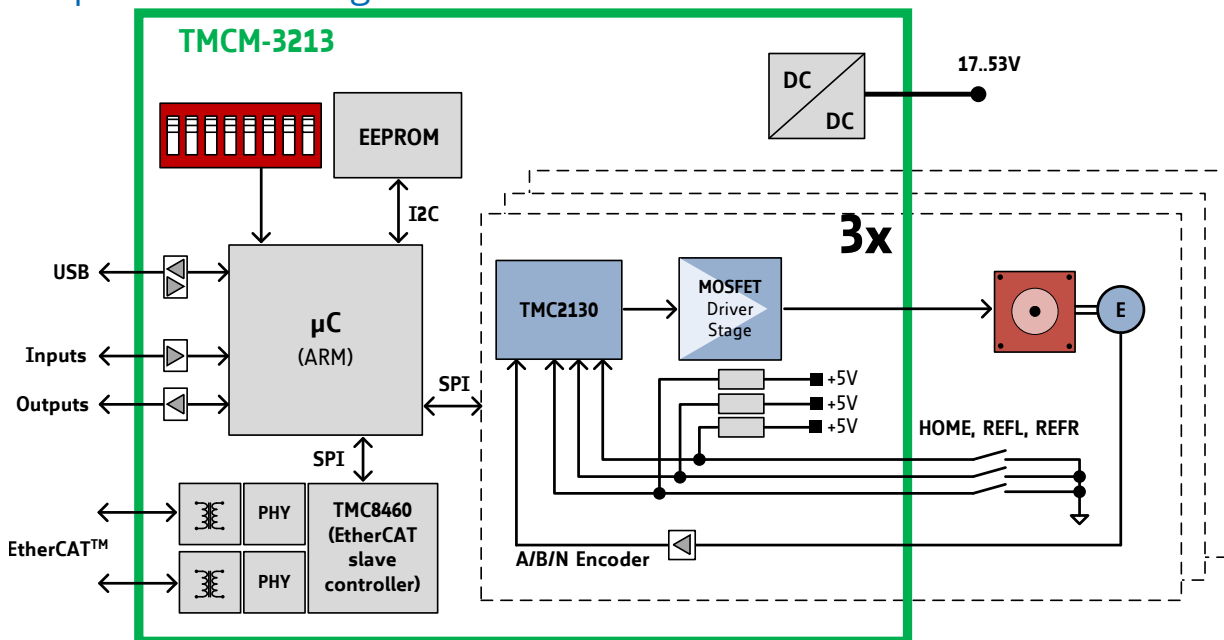
Features

- 3-Axes Stepper Motor Control
- CoE CiA-402 Drive Profile
- Encoder Support
- coolStep™
- dcStep™
- stallGuard2™
- stealthChop™

Applications

- Lab-Automation
- Semiconductor Handling
- Manufacturing
- Robotics
- Factory Automation
- CNC
- Laboratory Automation

Simplified Block Diagram



©2016 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany
 Terms of delivery and rights to technical change reserved.
 Download newest version at: www.trinamic.com



Read entire documentation.

Contents

1	Preface	6
1.1	General Features of this CoE Implementation	6
1.2	Abbreviations used in this Manual	7
1.3	Firmware Update	7
1.4	Trinamic's unique Features — easy to use with CoE	7
1.4.1	stallGuard2	8
1.4.2	coolStep	8
2	Communication	10
2.1	Reference Model	10
2.2	NMT State Machine	12
2.3	Device Model	13
2.4	Object Dictionary	14
2.4.1	Object Indices on Multi-Axis Modules	14
3	Communication area	16
3.1	Detailed object specifications	16
3.1.1	Object 1000 _h : Device Type	16
3.1.2	Object 1001 _h : Error Register	16
3.1.3	Object 1008 _h : Manufacturer Device Name	17
3.1.4	Object 1009 _h : Manufacturer Hardware Version	17
3.1.5	Object 100A _h : Manufacturer Software Version	18
3.1.6	Object 1018 _h : Identity Object	18
3.1.7	Object 1600 _h : Receive PDO Mapping Parameter	19
3.1.8	Objects 1A00 _h : Transmit PDO Mapping Parameter	20
3.1.9	Objects 1C00 _h : Sync Manager Communication Type	20
3.1.10	Objects 1C12 _h : Sync Manager 2 PDO Assignment	21
3.1.11	Objects 1C13 _h : Sync Manager 3 PDO Assignment	22
4	Manufacturer specific area	23
4.1	Objects related to coolStep	23
4.2	Detailed object specifications	26
4.2.1	Object 2000 _h : Microstep resolution	26
4.2.2	Object 2001 _h : Fullstep resolution	26
4.2.3	Object 2002 _h : Brake delay times	26
4.2.4	Object 2003 _h : Maximum current	27
4.2.5	Object 2004 _h : Standby current	28
4.2.6	Object 2005 _h : Limit switches	28
4.2.7	Object 200A _h : Enable drive delay time	29
4.2.8	Object 200B _h : Encoder parameters	29
4.2.9	Object 200C _h : Brake current feed	30
4.2.10	Object 2010 _h : Profile Start Velocity	30
4.2.11	Object 2011 _h : Profile A1	31
4.2.12	Object 2012 _h : Profile V1	31
4.2.13	Object 2013 _h : Profile D1	32
4.2.14	Object 2015 _h : Ramp Wait Time	32
4.2.15	Object 2089 _h : Setting Delay	32
4.2.16	Object 208C _h : Velocity Dimension Index	33
4.2.17	Object 208E _h : Acceleration Dimension Index	33
4.2.18	Object 2092 _h : Chopper Blank Time	34
4.2.19	Object 2093 _h : Chopper Mode	34
4.2.20	Object 2094 _h : Chopper Hysteresis Decrement	35



4.2.21	Object 2095 _h : Chopper Hysteresis End	35
4.2.22	Object 2096 _h : Chopper Hysteresis Start	36
4.2.23	Object 2097 _h : Chopper Off Time	36
4.2.24	Object 2098 _h : Smart Energy Current Minimum	36
4.2.25	Object 2099 _h : Smart Energy Current Down Step	37
4.2.26	Object 209A _h : Smart Energy Hysteresis	37
4.2.27	Object 209B _h : Smart Energy Current Up Step	38
4.2.28	Object 209C _h : Smart Energy Hysteresis Start	38
4.2.29	Object 209D _h : Smart Energy Filter Enable	39
4.2.30	Object 209E _h : stallGuard2 Threshold	39
4.2.31	Object 20A1 _h : Short Protection Disable	40
4.2.32	Object 20A3 _h : Vsense	40
4.2.33	Object 20A4 _h : Stop on Stall	41
4.2.34	Object 20A5 _h : Smart Energy Threshold Speed	41
4.2.35	Object 20B0 _h : PWM Threshold Speed	42
4.2.36	Object 20B1 _h : PWM Gradient	42
4.2.37	Object 20B2 _h : PWM Amplitude	43
4.2.38	Object 20B3 _h : dcStep Minimum Speed	43
4.2.39	Object 20B4 _h : dcStep Time	43
4.2.40	Object 20B5 _h : dcStep stallGuard	44
4.2.41	Object 20B6 _h : Fullstep Threshold Speed	44
4.2.42	Object 20B7 _h : High Speed Chopper Mode	45
4.2.43	Object 20B8 _h : High Speed Fullstep Mode	45
4.2.44	Object 20B9 _h : Power Down Ramp	45
4.2.45	Object 2100 _h : Home Offset Display	46
4.2.46	Object 2101 _h : Actual Load Value	46
4.2.47	Object 2102 _h : Driver Error Flags	47
4.2.48	Object 2107 _h : Microstep resolution display	47
4.2.49	Object 210B _h : Step Counter	48
4.2.50	Object 2121 _h : PWM Scale Value	48
4.2.51	Object 2122 _h : Measured Velocity	49
4.2.52	Object 2702 _h : Device Digital Inputs	49
4.2.53	Object 2703 _h : Device Digital Outputs	50
4.2.54	Object 270E _h : Device Analog Inputs	51

5	Profile specific area	52
5.1	Detailed object specifications	52
5.1.1	Object 605A _h : Quick stop option code	52
5.1.2	Object 605B _h : Shutdown option code	53
5.1.3	Object 605C _h : Disable operation option code	54
5.1.4	Object 605D _h : Halt option code	54
5.1.5	Object 605E _h : Fault reaction option code	55
5.1.6	Object 6060 _h : Modes of operation	55
5.1.7	Object 6061 _h : Modes of operation	56
5.1.8	Object 606A _h : Sensor selection code	57
5.1.9	Object 608F _h : Position Encoder Resolution	57
5.1.10	Object 60FD _h : Digital Inputs	58
5.1.11	Object 6502 _h : Supported Drive Modes	58



6	Profile position mode	60
6.1	Detailed Object Specifications	60
6.1.1	Object 6040 _h : Control Word	61
6.1.2	Object 6041 _h : Status Word	62
6.1.3	Object 6062 _h : Position Demand Value	63
6.1.4	Object 6063 _h : Position Actual Internal Value	64
6.1.5	Object 6064 _h : Position Actual Value	64
6.1.6	Object 6065 _h : Following Error Window	65
6.1.7	Object 6067 _h : Position Window	65
6.1.8	Object 6068 _h : Position Window Time	66
6.1.9	Object 606C _h : Velocity Actual Value	66
6.1.10	Object 607A _h : Target Position	67
6.1.11	Object 607D _h : Software Position Limit	67
6.1.12	Object 6081 _h : Profile Velocity	68
6.1.13	Object 6082 _h : End Velocity	68
6.1.14	Object 6083 _h : Profile Acceleration	69
6.1.15	Object 6084 _h : Profile Deceleration	69
6.1.16	Object 6085 _h : Quick Stop Deceleration	69
6.1.17	Object 60F2 _h : Positioning Option Code	70
6.2	How to move a Motor in pp Mode	71
7	Profile velocity mode	72
7.1	Detailed Object Specifications	72
7.1.1	Object 6040 _h : Control Word	72
7.1.2	Object 6041 _h : Status Word	73
7.1.3	Object 6062 _h : Position Demand Value	75
7.1.4	Object 6063 _h : Position Actual Internal Value	75
7.1.5	Object 6064 _h : Position Actual Value	76
7.1.6	Object 6065 _h : Following Error Window	76
7.1.7	Object 606C _h : Velocity Actual Value	77
7.1.8	Object 607D _h : Software Position Limit	77
7.1.9	Object 6083 _h : Profile Acceleration	78
7.1.10	Object 6085 _h : Quick Stop Deceleration	78
7.1.11	Object 60FF _h : Target Velocity	78
7.2	How to move a Motor in pv Mode	79
8	Homing mode	80
8.1	Homing Methods	81
8.1.1	Homing Method 1: Homing on negative Limit Switch and Index Pulse	81
8.1.2	Homing Method 2: Homing on positive Limit Switch and Index Pulse	82
8.1.3	Homing Method 3: Homing on positive Home Switch and Index Pulse	82
8.1.4	Homing Method 5: Homing on negative Home Switch and Index Pulse	82
8.1.5	Homing Method 17, 18, 19, and 21: Homing without Index Pulse	83
8.1.6	Homing Method 33 and 34: Homing on next Index Pulse	83
8.1.7	Homing Method 35: Current position as home position	84
8.2	Detailed Object Specifications	85
8.2.1	Object 6040 _h : Control Word	85
8.2.2	Object 6041 _h : Status Word	86
8.2.3	Object 606C _h : Velocity Actual Value	87
8.2.4	Object 607C _h : Home Offset	88
8.2.5	Object 6098 _h : Homing Method	89
8.2.6	Object 6099 _h : Homing Speeds	89
8.2.7	Object 609A _h : Homing Acceleration	89
8.2.8	Object 2100 _h : Home Offset Display	90



8.3 How to start a Homing in hm Mode	90
9 Emergency Messages (EMCY)	92
10 Figures Index	94
11 Tables Index	95
12 Supplemental Directives	98
12.1 Producer Information	98
12.2 Copyright	98
12.3 Trademark Designations and Symbols	98
12.4 Target User	98
12.5 Disclaimer: Life Support Systems	98
12.6 Disclaimer: Intended Use	98
12.7 Collateral Documents & Tools	99
13 Revision History	100
13.1 Firmware Revision	100
13.2 Document Revision	100



1 Preface

This document specifies objects and modes of operation of the Trinamic TMCM-3213 stepper motor control module with CANopen-over-EtherCAT (CoE) firmware. The CoE firmware is designed to fulfill the EtherCAT version of the CANopen DS402 standards. The EtherCAT conformance has also been tested. This manual assumes that the reader is already familiar with the basics of EtherCAT and the CoE protocol (especially DS402).

If necessary it is always possible to turn the module into a TMCL module by loading the TMCM-3213 TMCL firmware again through the USB interface, with the help of the firmware update function of the TMCL-IDE 3.0.

1.1 General Features of this CoE Implementation

Main Characteristics

- Communication according to EtherCAT standards
- Protocols: CoE, FoE

SDO Communication

- 1 server
- Expedited transfer
- Segmented transfer
- No block transfer

PDO Communication

- Producer
- Consumer
- RPDOs
 - Dynamic mapping with max. 9 mapping entries.
 - Default mappings: manufacturer specific.
- TPDOs
 - Dynamic mapping with max. 9 mapping entries.
 - Default mappings: manufacturer specific.

Sync managers

- Sync manager 0: receive mailbox used for SDO communication
- Sync manager 1: send mailbox used for SDO communication
- Sync manager 2: process data output (used for TPDO)
- Sync manager 3: process data input (used for RPDO)

Further Characteristics

- Emergency: producer



1.2 Abbreviations used in this Manual

Abbreviations	
CAN	Controller area network
CoE	CANopen over EtherCAT
CHGND	chassis ground / earth ground
COB	Communication object
FoE	File transfer over EtherCAT
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 boot loader and the CANopen firmware itself. Whereas the boot loader is installed during production and testing at TRINAMIC and remains untouched throughout the whole lifetime, the CANopen firmware can easily be updated by the user. The new firmware can be loaded into the module via the firmware update function of the TMCL-IDE, using the USB interface of the module.

1.4 Trinamic's unique Features — easy to use with CoE



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.

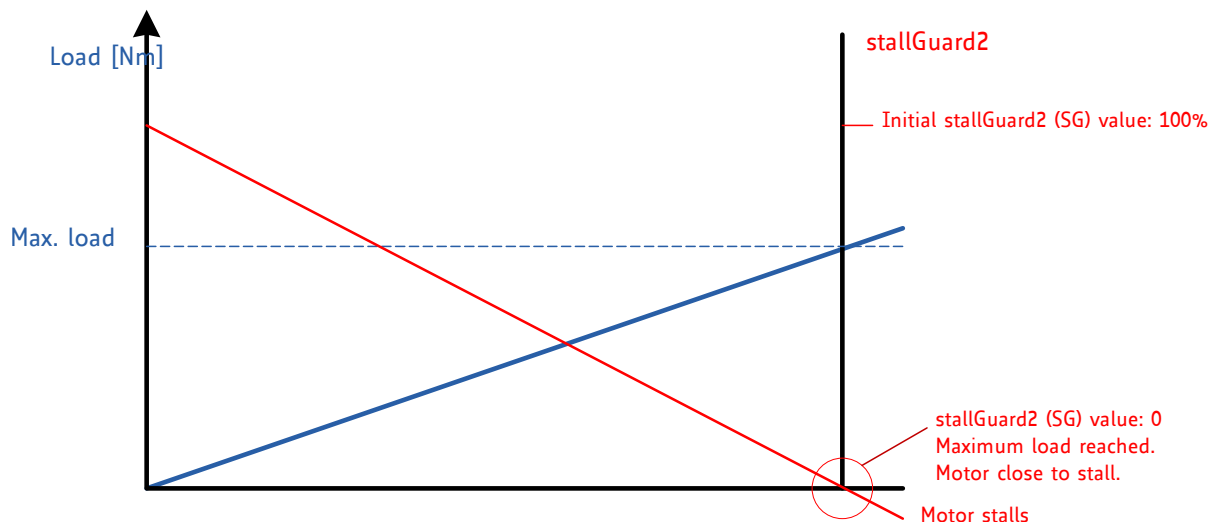


Figure 1: stallGuard2 Load Measurement as a Function of Load

1.4.2 coolStep

coolStep is a load-adaptive automatic current scaling based on the load measurement via 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. Because 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.



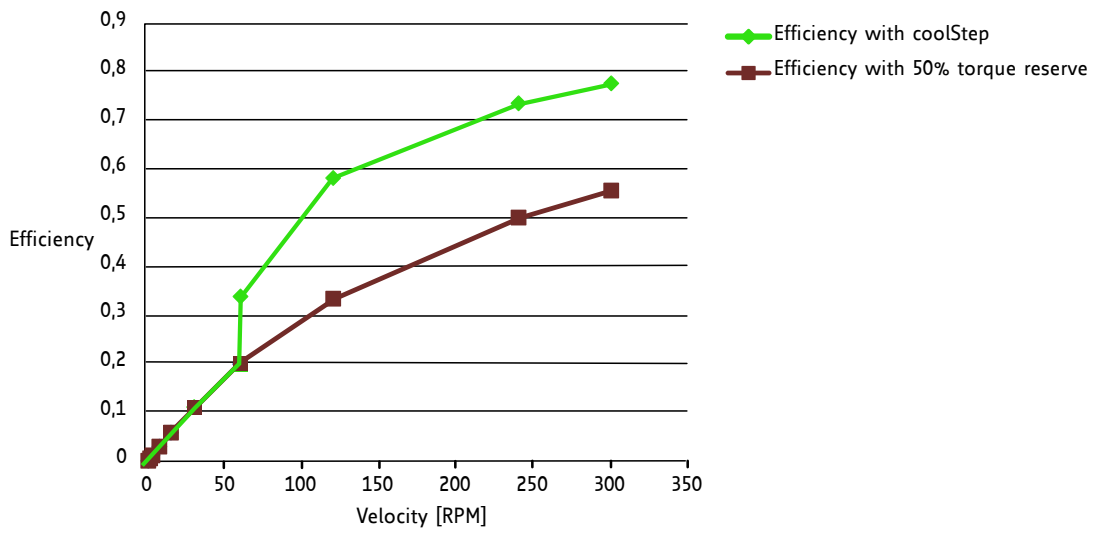


Figure 2: Energy Efficiency Example with coolStep



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 which 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 via the EtherCAT with peer service object(s) using a protocol.

The application and the 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 2: Service Primitives

A service type defines the primitives that are 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 3: 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 via the state machine. The NMT state machine consists of the states shown in figure 3.

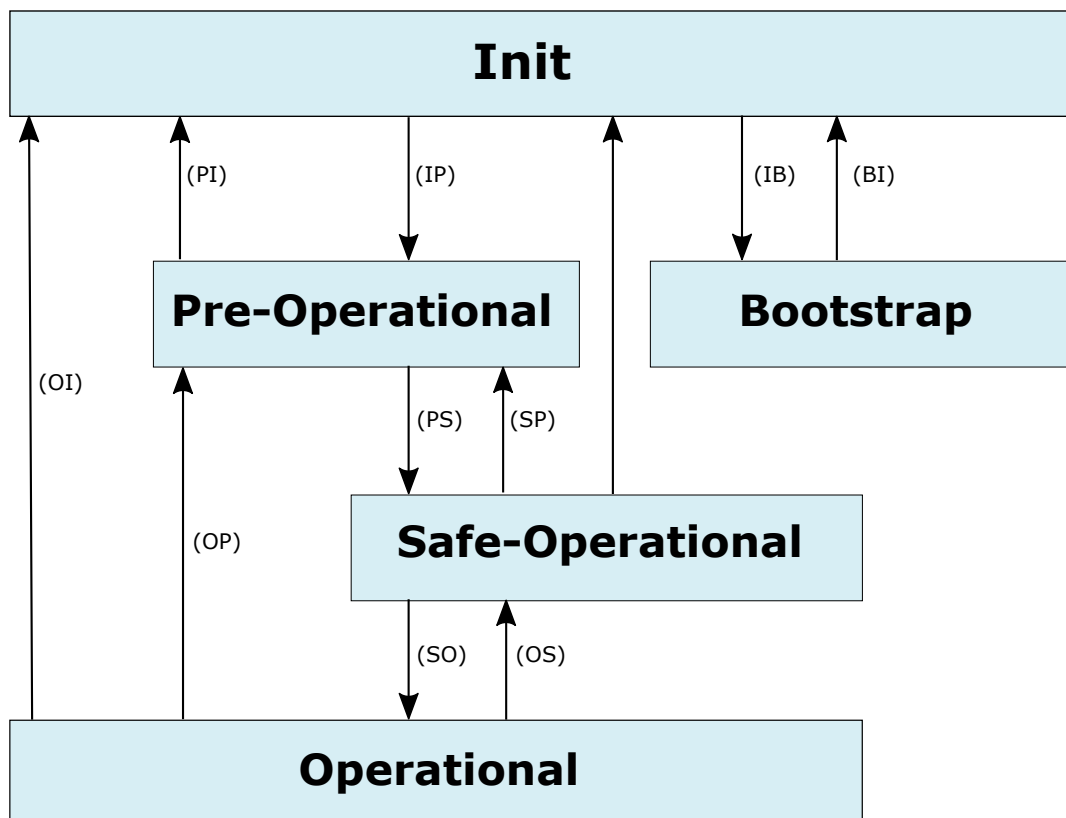


Figure 3: NMT State Machine

After power-on or reset the device enters the Initialization (**INIT**) state.

The master can then switch the device to Pre-Operational (**PRE-OP**) state. In this state, only SDO communication is possible. PDO communication is not possible.

In Safe-Operational (**SAFE-OP**) state, also PDO communication is possible. Inputs can be read, but outputs cannot be switched and the motor cannot be run.

In Operational (**OP**) state, all features of the module can be used. PDO communication is possible, outputs can be switched and the motor can be used. During Operational state the device can use all supported communication objects.

When switching from Operational to Safe-Operational state the motor will be stopped if it has been running. When the EtherCAT connection is lost during Operational state the device will also automatically switch to



Safe-Operational state.

The Bootstrap (**BOOT**) state is used for firmware updates via FoE. Before FoE can be used the device has to be switched to this state.

2.3 Device Model

A CoE device mainly consists of the following parts:

- *Communication*: This function unit provides the communication objects and the appropriate functionality to transport data items via the underlying network structure.
- *Object dictionary*: The object dictionary is a collection of all the data items which 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.

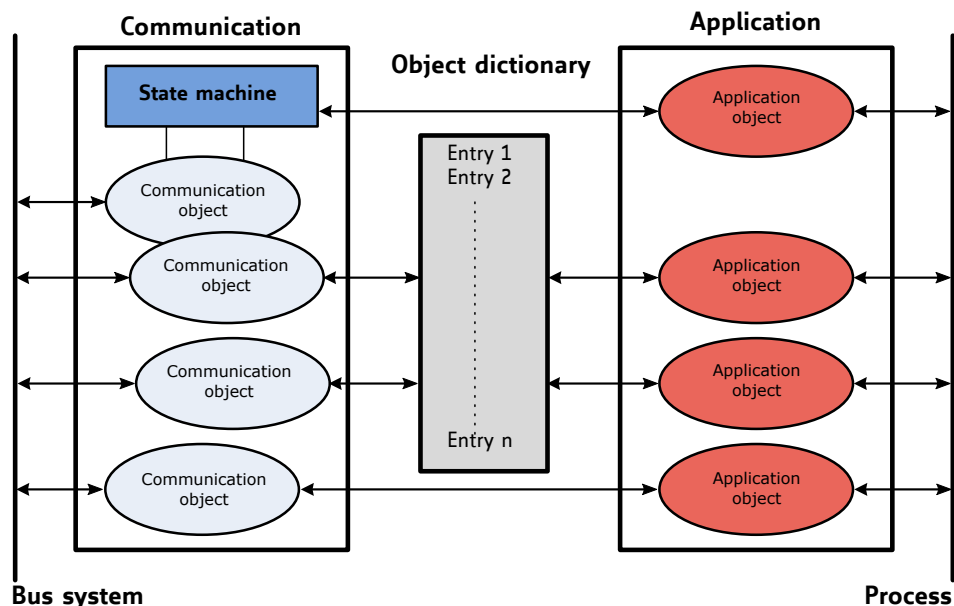


Figure 4: Device Model



2.4 Object Dictionary

The most important part of a device profile is the object dictionary description. The object dictionary is essentially a grouping of objects accessible via the network in an ordered pre-defined fashion. Each object within the dictionary is addressed using a 16-bit index. The overall layout of the standard object dictionary is shown in table 4:

Object Dictionary	
Index	Object
0000 _h	Not used.
0001 _h – 001F _h	Static data types.
0020 _h – 003F _h	Complex data types.
0040 _h – 005F _h	Manufacturer specific complex data types.
0060 _h – 007F _h	Device profile specific static data types.
0080 _h – 009F _h	Device profile specific complex data types.
00A0 _h – 0FFF _h	Reserved for further use.
1000 _h – 1FFF _h	Communication profile area.
2000 _h – 5FFF _h	Manufacturer specific profile area.
6000 _h – 9FFF _h	Standardized device profile area.
A000 _h – BFFF _h	Standardized interface profile area.
C000 _h – FFFF _h	Reserved for further use.

Table 4: 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-3213 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 via the network. They describe the device parameters and the device functionality of the device profile.

2.4.1 Object Indices on Multi-Axis Modules

On a multi-axis module like the TMCM-3213 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 $motor_number \cdot 200_h$ to the index of a manufacturer specific object to get its index for other motors.
- Add an offset of $motor_number \cdot 800_h$ 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 5: Multi-Axis Object Indices



3 Communication area

The communication area contains all objects that define the communication parameters of the CoE device according to the EtherCAT 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 which describes the device profile that is used and a second 16-bit field which provides additional information about optional functionality of the device.

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

Table 6: Object Description (1000_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	UNSIGNED32	FFFC0192 _h

Table 7: Entry Description (1000_h)

3.1.2 Object 1001_h: Error Register

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 which describes the device profile that is used and a second 16-bit field which provides additional information about optional functionality of the device.

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

Table 8: Object Description (1001_h)

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

Table 9: 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 10: Error Register Bits

3.1.3 Object 1008_h: Manufacturer Device Name

This object contains the manufacturer device name.

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

Table 11: Object Description (1008_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	—	TMCM-3213

Table 12: Entry Description (1008_h)

3.1.4 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 13: Object Description (1009_h)



Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	—	Depends on device, e.g. 1.0.

Table 14: Entry Description (1009_h)

3.1.5 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 15: Object Description (100A_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	—	Depends on device, e.g. 1.0.

Table 16: Entry Description (100A_h)

3.1.6 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 17: Object Description (1018_h)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
00 _h	Number of entries	ro	no	0...3	3
01 _h	Vendor ID	ro	no	UNSIGNED32	0286 _h
02 _h	Product code	ro	no	UNSIGNED32	3213
03 _h	Revision number	ro	no	UNSIGNED32	e.g. 20003 _h for version 2.3

Table 18: Entry Description (1018_h)

3.1.7 Object 1600_h: Receive PDO Mapping Parameter

This object contains the mapping parameters for the RPDO the device is able to receive. The sub-index 00_h contains the number of valid entries within the mapping record. This number of entries is also the number of the application variables which shall be received with the corresponding RPDO. The sub-indices from 01_h 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.

Object Description			
Index	Name	Object Type	Data Type
1600 _h	Receive PDO mapping parameter	RECORD	PDO Mapping

Table 19: Object Description (1600_h)

Entry Description				
Sub-index	Description	Access	Value Range	Default Value
00 _h	Number of mapped application objects in PDO	rw	0...9	Index 1600 _h : 4
01 _h	Mapping entry 1	rw	UNSIGNED32	60400010 _h
02 _h	Mapping entry 3	rw	UNSIGNED32	607A0020 _h
03 _h	Mapping entry 4	rw	UNSIGNED32	60710010 _h
04 _h	Mapping entry 5	rw	UNSIGNED32	60FF0020 _h
05 _h	Mapping entry 2	rw	UNSIGNED32	0 _h
06 _h	Mapping entry 6	rw	UNSIGNED32	0 _h
07 _h	Mapping entry 7	rw	UNSIGNED32	0 _h
08 _h	Mapping entry 8	rw	UNSIGNED32	0 _h
09 _h	Mapping entry 9	rw	UNSIGNED32	0 _h

Table 20: Entry Description (1600_h)



3.1.8 Objects 1A00_h: Transmit PDO Mapping Parameter

This object contains the mapping parameters for the TPDO the device is able to transmit. The sub-index 00_h contains the number of valid entries within the mapping record. This number of entries is also the number of the application variables which shall be transmitted with the corresponding TPDO. The sub-indices from 01_h 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.

Object Description			
Index	Name	Object Type	Data Type
1A00 _h	Transmit PDO mapping parameter	RECORD	PDO Mapping

Table 21: Object Description (1A00_h)

Entry Description				
Sub-index	Description	Access	Value Range	Default Value
00 _h	Number of mapped application objects in PDO	rw	0...9	6
01 _h	Mapping entry 1	rw	UNSIGNED32	60410010 _h
02 _h	Mapping entry 2	rw	UNSIGNED32	60610008 _h
03 _h	Mapping entry 3	rw	UNSIGNED32	60640020 _h
04 _h	Mapping entry 4	rw	UNSIGNED32	60770010 _h
05 _h	Mapping entry 5	rw	UNSIGNED32	606C0020 _h
06 _h	Mapping entry 6	rw	UNSIGNED32	60FD0020 _h
07 _h	Mapping entry 7	rw	UNSIGNED32	0 _h
08 _h	Mapping entry 8	rw	UNSIGNED32	0 _h
09 _h	Mapping entry 9	rw	UNSIGNED32	0 _h

Table 22: Entry Description (1A00_h)

3.1.9 Objects 1C00_h: Sync Manager Communication Type

This object describes the communication types of the EtherCAT sync managers. The types of the first four sync managers are normally fixed and should not be changed. Sync managers can have the following for communication types:



Sync Manager Communication Types	
Type	Description
1	Mailbox receive
2	Mailbox send
3	Process data input
4	Process data output

Table 23: Sync Manager Communication Types

Object Description			
Index	Name	Object Type	Data Type
1C00 _h	Sync manager communication type	RECORD	UNSIGNED8

Table 24: Object Description (1C00_h)

Entry Description				
Sub-index	Description	Access	Value Range	Default Value
00 _h	Number of entries	rw	0...3	4
01 _h	Communication type sync manager 1	rw	UNSIGNED8	1
02 _h	Communication type sync manager 2	rw	UNSIGNED8	2
03 _h	Communication type sync manager 3	rw	UNSIGNED8	3
04 _h	Communication type sync manager 4	rw	UNSIGNED8	4

Table 25: Entry Description (1C00_h)

3.1.10 Objects 1C12_h: Sync Manager 2 PDO Assignment

This object contains the index of the PDO definition object that is assigned to sync manager 2. Normally, the RPDO objects are assigned to sync manager 2. Under most circumstances there is no need to change this setting.

Object Description			
Index	Name	Object Type	Data Type
1C12 _h	Sync manager 2 PDO assignment	RECORD	PDO assignment

Table 26: Object Description (1C12_h)



Entry Description				
Sub-index	Description	Access	Value Range	Default Value
00 _h	Number of assigned PDOs	rw	0...1	1
01 _h	PDO mapping index of assigned RPDO	rw	UNSIGNED16	1600 _h

Table 27: Entry Description (1C12_h)

3.1.11 Objects 1C13_h: Sync Manager 3 PDO Assignment

This object contains the index of the PDO definition object that is assigned to sync manager 3. Normally, the TPDO objects are assigned to sync manager 3. Under most circumstances there is no need to change this setting.

Object Description			
Index	Name	Object Type	Data Type
1C13 _h	Sync manager 3 PDO assignment	RECORD	PDO assignment

Table 28: Object Description (1C13_h)

Entry Description				
Sub-index	Description	Access	Value Range	Default Value
00 _h	Number of assigned PDOs	rw	0...1	1
01 _h	PDO mapping index of assigned TPDO	rw	UNSIGNED16	1A00 _h

Table 29: Entry Description (1C13_h)

4 Manufacturer specific area

The manufacturer segment contains manufacturer specific objects. These objects control the special features of the Trinamic Motion Control device TMC3213.

i 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 $motor_number \cdot 200_h$ 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. Please see also section 2.4.1.

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 30: Multi-axis Object Indices (Manufacturer specific Area)

4.1 Objects related to coolStep

Figure 5 shows an overview of the coolStep related objects for motor #0. Please 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

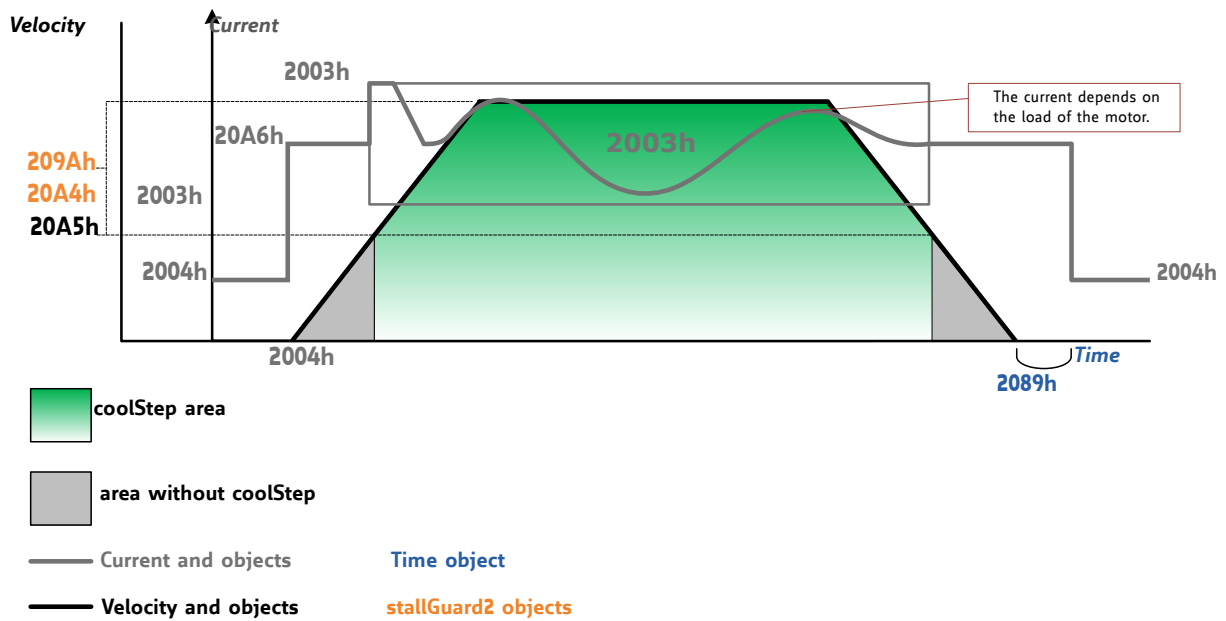


Figure 5: coolStep Adjustment Points and Thresholds



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 stallGuard 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 stallGuard 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 will not be stopped. Above this speed the motor will stop 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 31: coolStep related Objects

