



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

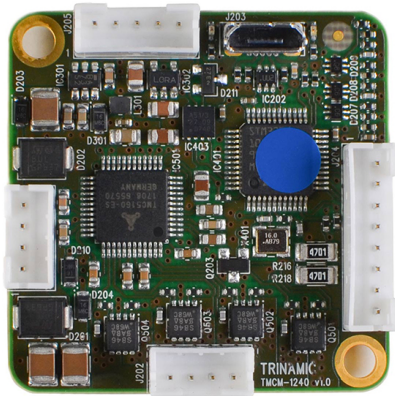
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China



# TMCM-1240 CANopen® Firmware Manual

Firmware Version V3.21 | Document Revision V1.00 • 2017-DEC-01

The TMCM-1240 is a single axis controller/driver module for 2-phase bipolar stepper motors. The TMCM-1240 CANopen® firmware allows to control the module using the CANopen® protocol, making use of the Trinamic TMC5160 motion controller and motor driver. Dynamic current control, and quiet, smooth and efficient operation are combined with stealthChop™, dcStep™, stallGuard™ and coolStep™ features.



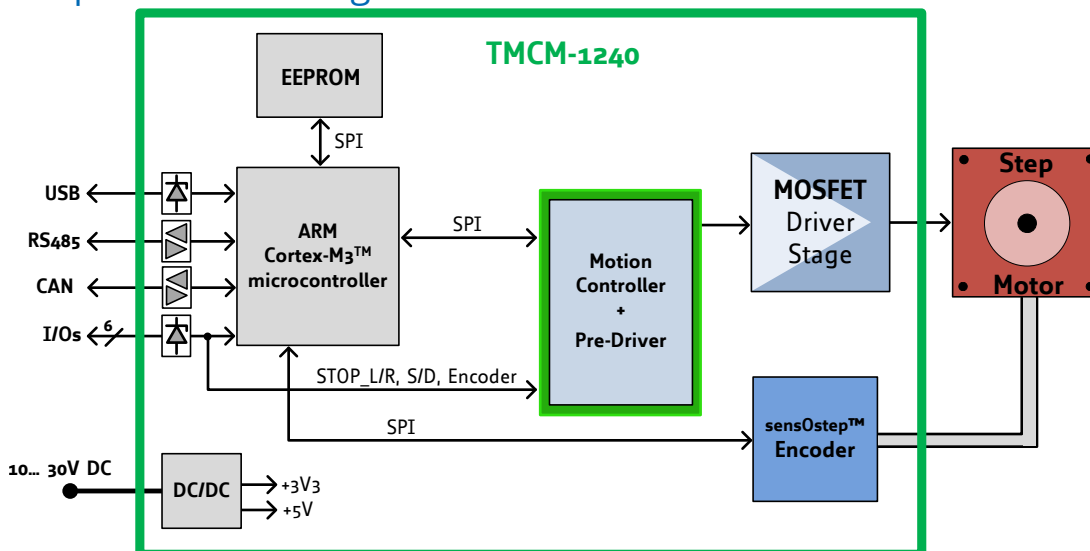
## Features

- Single axis Stepper Motor Control
- CANopen® CiA-402 Drive Profile
- coolStep™
- dcStep™
- stallGuard2™
- stealthChop™

## Applications

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

## Simplified Block Diagram



# Contents

<b>1</b>	<b>Preface</b>	<b>6</b>
1.1	General Features of this CANopen 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 CANopen	8
1.4.1	stallGuard2	8
1.4.2	coolStep	8
<b>2</b>	<b>Communication</b>	<b>10</b>
2.1	Reference Model	10
2.2	NMT State Machine	12
2.3	Device Model	13
2.4	Object Dictionary	14
<b>3</b>	<b>Communication Area</b>	<b>16</b>
3.1	Detailed Object Specifications	16
3.1.1	Object 1000 <sub>h</sub> : Device Type	16
3.1.2	Object 1001 <sub>h</sub> : Error Register	16
3.1.3	Object 1005 <sub>h</sub> : COB-ID SYNC Message	17
3.1.4	Object 1008 <sub>h</sub> : Manufacturer Device Name	18
3.1.5	Object 1009 <sub>h</sub> : Manufacturer Hardware Version	18
3.1.6	Object 100A <sub>h</sub> : Manufacturer Software Version	18
3.1.7	Object 100C <sub>h</sub> : Guard Time	19
3.1.8	Object 100D <sub>h</sub> : Life Time Factor	19
3.1.9	Object 1010 <sub>h</sub> : Store Parameters	19
3.1.10	Object 1011 <sub>h</sub> : Restore Parameters	21
3.1.11	Object 1014 <sub>h</sub> : COB-ID Emergency Object	22
3.1.12	Object 1015 <sub>h</sub> : Inhibit Time EMCY	22
3.1.13	Object 1016 <sub>h</sub> : Consumer Heartbeat Time	23
3.1.14	Object 1017 <sub>h</sub> : Producer Heartbeat Time	23
3.1.15	Object 1018 <sub>h</sub> : Identity Object	24
3.1.16	Object 1029 <sub>h</sub> : Error Behaviour	24
3.1.17	Objects 1400 <sub>h</sub> – 1403 <sub>h</sub> : Receive PDO Communication Parameter	25
3.1.18	Objects 1600 <sub>h</sub> – 1603 <sub>h</sub> : Receive PDO Mapping Parameter	26
3.1.19	Objects 1800 <sub>h</sub> – 1803 <sub>h</sub> : Transmit PDO Communication Parameter	27
3.1.20	Objects 1A00 <sub>h</sub> – 1A03 <sub>h</sub> : Transmit PDO Mapping Parameter	28
<b>4</b>	<b>Manufacturer specific Area</b>	<b>30</b>
4.1	Objects related to coolStep	30
4.2	Detailed Object Specifications	32
4.2.1	Object 2000 <sub>h</sub> : Microstep Resolution	32
4.2.2	Object 2001 <sub>h</sub> : Fullstep Resolution	32
4.2.3	Object 2002 <sub>h</sub> : Brake Delay Times	32
4.2.4	Object 2003 <sub>h</sub> : Maximum Current	33
4.2.5	Object 2004 <sub>h</sub> : Standby Current	34
4.2.6	Object 2005 <sub>h</sub> : Limit Switches	34
4.2.7	Object 200A <sub>h</sub> : Enable Drive Delay Time	35
4.2.8	Object 200B <sub>h</sub> : Encoder Parameters	35
4.2.9	Object 200C <sub>h</sub> : Brake Current Feed	36
4.2.10	Object 2010 <sub>h</sub> : Profile Start Velocity	36
4.2.11	Object 2011 <sub>h</sub> : Profile A1	37
4.2.12	Object 2012 <sub>h</sub> : Profile V1	37



4.2.13 Object 2013 <sub>h</sub> : Profile D1	38
4.2.14 Object 2015 <sub>h</sub> : Ramp Wait Time	38
4.2.15 Object 2089 <sub>h</sub> : Setting Delay	38
4.2.16 Object 208C <sub>h</sub> : Velocity Dimension Index	39
4.2.17 Object 208E <sub>h</sub> : Acceleration Dimension Index	39
4.2.18 Object 2092 <sub>h</sub> : Chopper Blank Time	40
4.2.19 Object 2093 <sub>h</sub> : Chopper Mode	40
4.2.20 Object 2094 <sub>h</sub> : Chopper Hysteresis Decrement	41
4.2.21 Object 2095 <sub>h</sub> : Chopper Hysteresis End	41
4.2.22 Object 2096 <sub>h</sub> : Chopper Hysteresis Start	42
4.2.23 Object 2097 <sub>h</sub> : Chopper Off Time	42
4.2.24 Object 2098 <sub>h</sub> : Smart Energy Current Minimum	42
4.2.25 Object 2099 <sub>h</sub> : Smart Energy Current Down Step	43
4.2.26 Object 209A <sub>h</sub> : Smart Energy Hysteresis	43
4.2.27 Object 209B <sub>h</sub> : Smart Energy Current Up Step	44
4.2.28 Object 209C <sub>h</sub> : Smart Energy Hysteresis Start	44
4.2.29 Object 209D <sub>h</sub> : Smart Energy Filter Enable	45
4.2.30 Object 209E <sub>h</sub> : stallGuard2 Threshold	45
4.2.31 Object 20A1 <sub>h</sub> : Short Protection Disable	46
4.2.32 Object 20A4 <sub>h</sub> : Stop on Stall	46
4.2.33 Object 20A5 <sub>h</sub> : Smart Energy Threshold Speed	47
4.2.34 Object 20B0 <sub>h</sub> : PWM Threshold Speed	47
4.2.35 Object 20B1 <sub>h</sub> : PWM Gradient	48
4.2.36 Object 20B2 <sub>h</sub> : PWM Amplitude	48
4.2.37 Object 20B3 <sub>h</sub> : dcStep Minimum Speed	48
4.2.38 Object 20B4 <sub>h</sub> : dcStep Time	49
4.2.39 Object 20B5 <sub>h</sub> : dcStep stallGuard	49
4.2.40 Object 20B6 <sub>h</sub> : Fullstep Threshold Speed	50
4.2.41 Object 20B7 <sub>h</sub> : High Speed Chopper Mode	50
4.2.42 Object 20B8 <sub>h</sub> : High Speed Fullstep Mode	50
4.2.43 Object 20B9 <sub>h</sub> : Power Down Ramp	51
4.2.44 Object 2100 <sub>h</sub> : Home Offset Display	51
4.2.45 Object 2101 <sub>h</sub> : Actual Load Value	52
4.2.46 Object 2102 <sub>h</sub> : Driver Error Flags	52
4.2.47 Object 2107 <sub>h</sub> : Microstep Resolution Display	53
4.2.48 Object 210B <sub>h</sub> : Step Counter	53
4.2.49 Object 2121 <sub>h</sub> : PWM Scale Value	54
4.2.50 Object 2122 <sub>h</sub> : Measured Velocity	54
4.2.51 Object 2700 <sub>h</sub> : TMCL Direct Communication	55
4.2.52 Object 2701 <sub>h</sub> : Manufacturer Specific Mode	55
4.2.53 Object 2702 <sub>h</sub> : Device Digital Inputs	56
4.2.54 Object 2703 <sub>h</sub> : Device Digital Outputs	56
4.2.55 Object 2704 <sub>h</sub> : CAN Bit Rate	57
4.2.56 Object 2705 <sub>h</sub> : Node ID	57
4.2.57 Object 2706 <sub>h</sub> : Store	58
4.2.58 Object 2707 <sub>h</sub> : CAN Bit Rate Load	58
4.2.59 Object 2708 <sub>h</sub> : Node ID Load	59
4.2.60 Object 270E <sub>h</sub> : Device Analog Inputs	59
4.2.61 Object 2710 <sub>h</sub> : Pull-up Resistors	59



<b>5</b>	<b>Profile specific Area</b>	<b>61</b>
5.1	Detailed Object Specifications	61
5.1.1	Object 605A <sub>h</sub> : Quick Stop Option Code	61
5.1.2	Object 605B <sub>h</sub> : Shutdown Option Code	62
5.1.3	Object 605C <sub>h</sub> : Disable Operation Option Code	62
5.1.4	Object 605D <sub>h</sub> : Halt Option Code	63
5.1.5	Object 605E <sub>h</sub> : Fault Reaction Option Code	63
5.1.6	Object 6060 <sub>h</sub> : Modes of Operation	64
5.1.7	Object 6061 <sub>h</sub> : Modes of Operation Display	65
5.1.8	Object 606A <sub>h</sub> : Sensor Selection Code	65
5.1.9	Object 608F <sub>h</sub> : Position Encoder Resolution	66
5.1.10	Object 60FD <sub>h</sub> : Digital Inputs	66
5.1.11	Object 6502 <sub>h</sub> : Supported Drive Modes	67
<b>6</b>	<b>Profile Position Mode</b>	<b>69</b>
6.1	Detailed Object Specifications	69
6.1.1	Object 6040 <sub>h</sub> : Control Word	70
6.1.2	Object 6041 <sub>h</sub> : Status Word	71
6.1.3	Object 6062 <sub>h</sub> : Position Demand Value	72
6.1.4	Object 6063 <sub>h</sub> : Position Actual Internal Value	73
6.1.5	Object 6064 <sub>h</sub> : Position Actual Value	73
6.1.6	Object 6065 <sub>h</sub> : Following Error Window	74
6.1.7	Object 6067 <sub>h</sub> : Position Window	74
6.1.8	Object 6068 <sub>h</sub> : Position Window Time	75
6.1.9	Object 606C <sub>h</sub> : Velocity Actual Value	75
6.1.10	Object 607A <sub>h</sub> : Target Position	76
6.1.11	Object 607D <sub>h</sub> : Software Position Limit	76
6.1.12	Object 6081 <sub>h</sub> : Profile Velocity	77
6.1.13	Object 6082 <sub>h</sub> : End Velocity	77
6.1.14	Object 6083 <sub>h</sub> : Profile Acceleration	78
6.1.15	Object 6084 <sub>h</sub> : Profile Deceleration	78
6.1.16	Object 6085 <sub>h</sub> : Quick Stop Deceleration	78
6.1.17	Object 60F2 <sub>h</sub> : Positioning Option Code	79
6.2	How to move a Motor in pp Mode	80
<b>7</b>	<b>Profile Velocity Mode</b>	<b>81</b>
7.1	Detailed Object Specifications	81
7.1.1	Object 6040 <sub>h</sub> : Control Word	81
7.1.2	Object 6041 <sub>h</sub> : Status Word	82
7.1.3	Object 6062 <sub>h</sub> : Position Demand Value	84
7.1.4	Object 6063 <sub>h</sub> : Position Actual Internal Value	84
7.1.5	Object 6064 <sub>h</sub> : Position Actual Value	85
7.1.6	Object 6065 <sub>h</sub> : Following Error Window	85
7.1.7	Object 606C <sub>h</sub> : Velocity Actual Value	86
7.1.8	Object 607D <sub>h</sub> : Software Position Limit	86
7.1.9	Object 6083 <sub>h</sub> : Profile Acceleration	87
7.1.10	Object 6085 <sub>h</sub> : Quick Stop Deceleration	87
7.1.11	Object 60FF <sub>h</sub> : Target Velocity	87
7.2	How to move a Motor in pv Mode	88



<b>8 Homing Mode</b>	<b>89</b>
8.1 Homing Methods	90
8.1.1 Homing Method 1: Homing on negative Limit Switch and Index Pulse	90
8.1.2 Homing Method 2: Homing on positive Limit Switch and Index Pulse	91
8.1.3 Homing Method 3: Homing on positive Home Switch and Index Pulse	91
8.1.4 Homing Method 5: Homing on negative Home Switch and Index Pulse	91
8.1.5 Homing Method 17, 18, 19, and 21: Homing without Index Pulse	92
8.1.6 Homing Method 33 and 34: Homing on next Index Pulse	92
8.1.7 Homing Method 35: Current Position as Home Position	93
8.2 Detailed Object Specifications	94
8.2.1 Object 6040 <sub>h</sub> : Control Word	94
8.2.2 Object 6041 <sub>h</sub> : Status Word	95
8.2.3 Object 606C <sub>h</sub> : Velocity Actual Value	96
8.2.4 Object 607C <sub>h</sub> : Home Offset	97
8.2.5 Object 6098 <sub>h</sub> : Homing Method	98
8.2.6 Object 6099 <sub>h</sub> : Homing Speeds	98
8.2.7 Object 609A <sub>h</sub> : Homing Acceleration	98
8.2.8 Object 2100 <sub>h</sub> : Home Offset Display	99
8.3 How to start a Homing in hm Mode	99
<b>9 Cyclic synchronous Position Mode</b>	<b>101</b>
9.1 Detailed Object Specifications	101
9.1.1 Object 6040 <sub>h</sub> : Control Word	101
9.1.2 Object 6041 <sub>h</sub> : Status Word	102
9.1.3 Object 6062 <sub>h</sub> : Position Demand Value	104
9.1.4 Object 6063 <sub>h</sub> : Position Actual Internal Value	104
9.1.5 Object 6064 <sub>h</sub> : Position Actual Value	105
9.1.6 Object 606C <sub>h</sub> : Velocity Actual Value	105
9.1.7 Object 607A <sub>h</sub> : Target Position	105
9.1.8 Object 607D <sub>h</sub> : Software Position Limit	106
9.1.9 Object 60B0 <sub>h</sub> : Position Offset	106
9.1.10 Object 60C2 <sub>h</sub> : Interpolation Time Period	107
<b>10 Emergency Messages (EMCY)</b>	<b>108</b>
<b>11 Figures Index</b>	<b>110</b>
<b>12 Tables Index</b>	<b>111</b>
<b>13 Supplemental Directives</b>	<b>115</b>
13.1 Producer Information	115
13.2 Copyright	115
13.3 Trademark Designations and Symbols	115
13.4 Target User	115
13.5 Disclaimer: Life Support Systems	115
13.6 Disclaimer: Intended Use	115
13.7 Collateral Documents & Tools	116
<b>14 Revision History</b>	<b>117</b>
14.1 Firmware Revision	117
14.2 Document Revision	117



# 1 Preface

This document specifies objects and modes of operation of the Trinamic TMCM-1240 stepper motor control module with CANopen firmware. The CANopen firmware is designed to fulfill the CANopen DS402 and DS301 standards. This manual assumes that the reader is already familiar with the basics of the CANopen protocol, defined by the DS301 and DS402 standards of the CAN-CiA. If necessary it is always possible to turn the module into a TMCL module by loading the TMCM-1240 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 CANopen Implementation

### Main Characteristics

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

### SDO Communication

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

### PDO Communication

- Producer
- Consumer
- RPDOs
  - Axis 0: 1, 2, 3, 4
  - Transmission modes: asynchronous.
  - Dynamic mapping with max. 3 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
  - Transmission modes: asynchronous, asynchronous with event timer, synchronous.
  - Dynamic mapping with max. 3 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 3 are synchronous PDOs)
- Emergency: producer
- RTR: supported only for node guarding/life guarding
- 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 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 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.

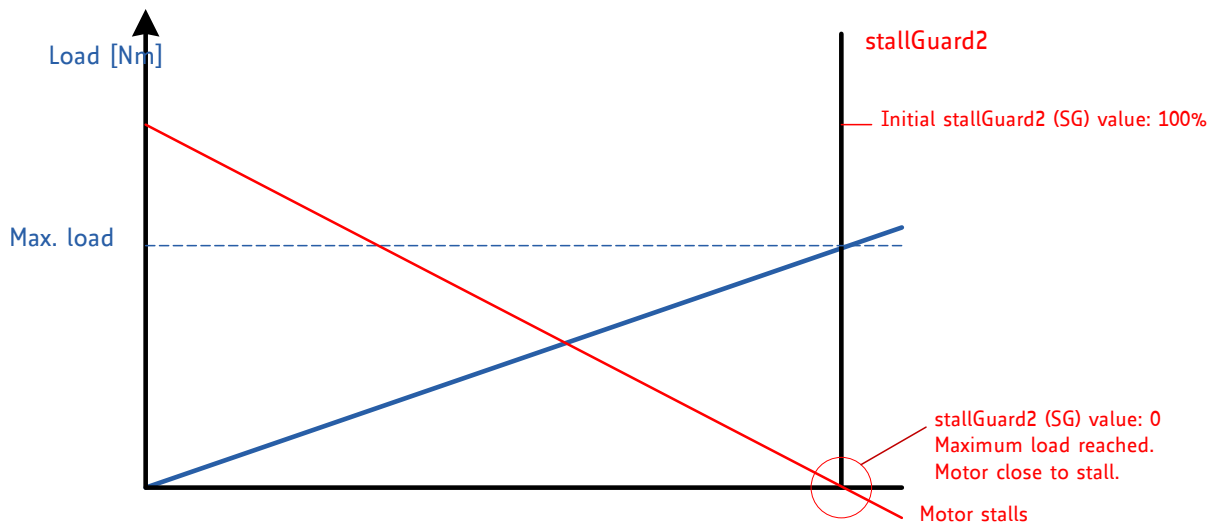


Figure 1: stallGuard2 Load Measurement as a Function of Load

### 1.4.2 coolStep

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



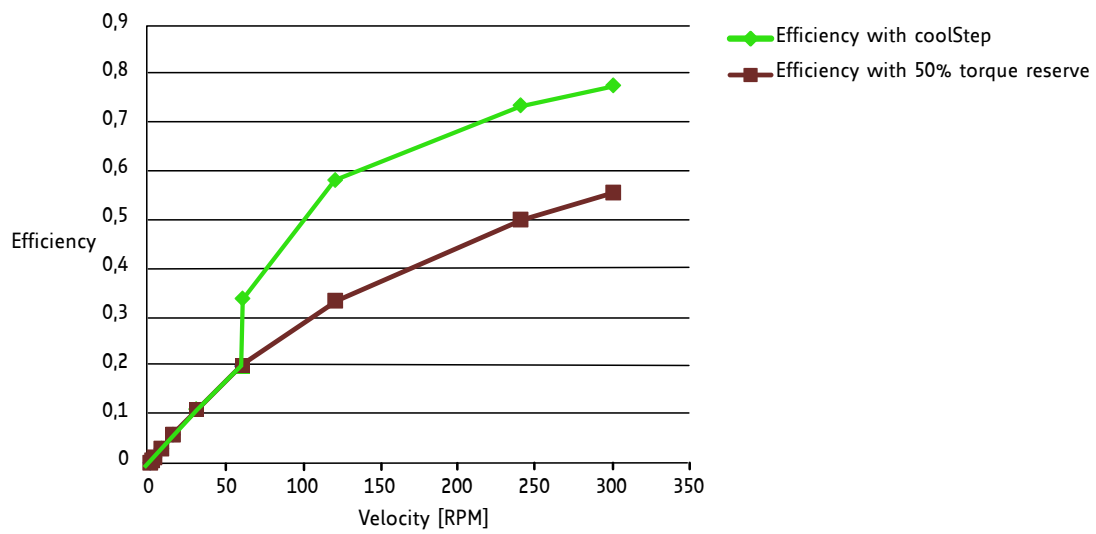


Figure 2: Energy Efficiency Example with coolStep



## 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 CAN Network 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.

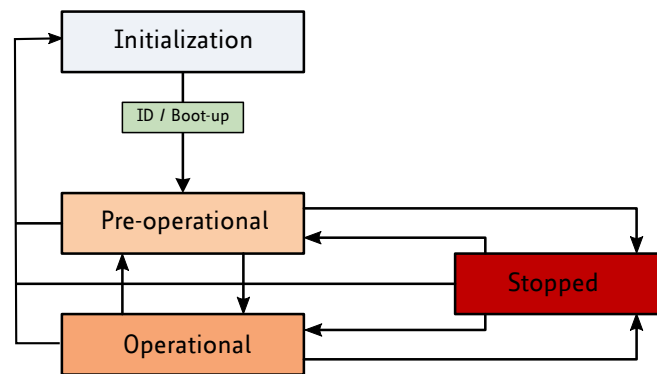


Figure 3: NMT State Machine

After power-on or reset the device enters the Initialization 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 that it is ready to work. A device that stays in Pre-operational state may start to transmit SYNC-, time stamp- or heartbeat message. In contrast to the PDO communication that is disabled in this state, the device can communicate via SDO.

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

A device that was 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 (e.g. to start and stop the device), 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 which 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 will be switch to FAULT state.



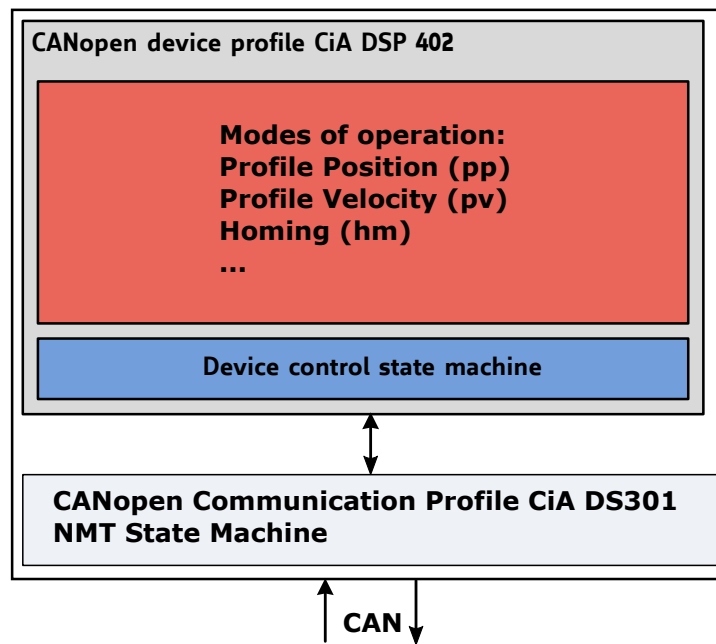


Figure 4: Communication Architecture

## 2.3 Device Model

A CANopen 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.



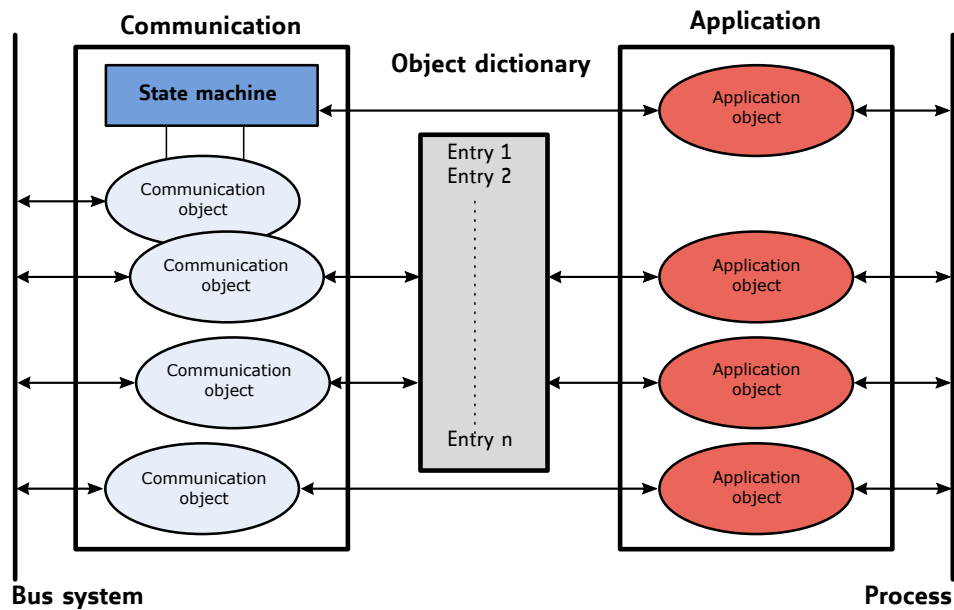


Figure 5: Device Model

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

Table 4: Object Dictionary



The communication profile area at indices 1000<sub>h</sub> through 1FFF<sub>h</sub> contains the communication specific parameters for the CAN network. These entries are common to all devices.

The manufacturer segment at indices 2000<sub>h</sub> through 5FFF<sub>h</sub> contains manufacturer specific objects. These objects control the special features of the Trinamic TMCM-1240 motion control device.

The standardized device profile area at indices 6000<sub>h</sub> through 9FFF<sub>h</sub> 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.





## 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<sub>h</sub>: Device Type

This object contains information about the device type. The object 1000<sub>h</sub> 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 <sub>h</sub>	Device type	Variable	UNSIGNED32

Table 5: Object Description (1000<sub>h</sub>)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	UNSIGNED32	FFFC0192 <sub>h</sub>

Table 6: Entry Description (1000<sub>h</sub>)

#### 3.1.2 Object 1001<sub>h</sub>: Error Register

This object contains error information. The CANopen device maps internal errors into object 1001<sub>h</sub>. It is part of an emergency object.

Object Description			
Index	Name	Object Type	Data Type
1001 <sub>h</sub>	Error register	Variable	UNSIGNED8

Table 7: Object Description (1001<sub>h</sub>)

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

Table 8: Entry Description (1001<sub>h</sub>)



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 9: Error Register Bits

### 3.1.3 Object 1005<sub>h</sub>: COB-ID SYNC Message

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 10: Value Definition (1005<sub>h</sub>)

Object Description			
Index	Name	Object Type	Data Type
1005 <sub>h</sub>	COB-ID SYNC message	Variable	UNSIGNED32

Table 11: Object Description (1005<sub>h</sub>)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	UNSIGNED32	80 <sub>h</sub>

Table 12: Entry Description (1005<sub>h</sub>)

### 3.1.4 Object 1008<sub>h</sub>: 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 <sub>h</sub>	Manufacturer Device Name	Variable	Visible String

Table 13: Object Description (1008<sub>h</sub>)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	—	TMCM-1240

Table 14: Entry Description (1008<sub>h</sub>)

### 3.1.5 Object 1009<sub>h</sub>: Manufacturer Hardware Version

This object contains the hardware version description.

Object Description			
Index	Name	Object Type	Data Type
1009 <sub>h</sub>	Manufacturer Hardware Version	Variable	Visible String

Table 15: Object Description (1009<sub>h</sub>)

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 (1009<sub>h</sub>)

### 3.1.6 Object 100A<sub>h</sub>: Manufacturer Software Version

This object contains the software version description.

Object Description			
Index	Name	Object Type	Data Type
100A <sub>h</sub>	Manufacturer Software Version	Variable	Visible String

Table 17: Object Description (100A<sub>h</sub>)



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

Table 18: Entry Description (100A<sub>h</sub>)

### 3.1.7 Object 100C<sub>h</sub>: Guard Time

The objects at index 100C<sub>h</sub> and 100D<sub>h</sub> shall indicate the configured guard time respectively the life time factor. The life time factor multiplied with the guard time gives the life time for the life guarding protocol.

Object Description			
Index	Name	Object Type	Data Type
100C <sub>h</sub>	Guard Time	Variable	UNSIGNED16

Table 19: Object Description (100C<sub>h</sub>)

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

Table 20: Entry Description (100C<sub>h</sub>)

### 3.1.8 Object 100D<sub>h</sub>: Life Time Factor

The life time factor multiplied with the guard time gives the life time for the life guarding protocol.

Object Description			
Index	Name	Object Type	Data Type
100D <sub>h</sub>	Life Time Factor	Variable	UNSIGNED8

Table 21: Object Description (100D<sub>h</sub>)

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

Table 22: Entry Description (100D<sub>h</sub>)

### 3.1.9 Object 1010<sub>h</sub>: Store Parameters

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



There are several parameter groups:

- Sub-index 0<sub>h</sub>: contains the largest sub-index that is supported.
- Sub-index 1<sub>h</sub>: saves all parameters.
- Sub-index 2<sub>h</sub>: saves communication parameters 100C<sub>h</sub>, 100D<sub>h</sub>, 1015<sub>h</sub>, 1017<sub>h</sub>, and 1029<sub>h</sub>.
- Sub-index 3<sub>h</sub>: saves device profile parameters.
- Sub-index 4<sub>h</sub>: saves motor 0 parameters.

**Note** In order 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<sub>h</sub>, see also table 23).

Save Signature			
e	v	a	s
65 <sub>h</sub>	76 <sub>h</sub>	61 <sub>h</sub>	73 <sub>h</sub>

Table 23: 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<sub>h</sub>). If a wrong signature is written, the device refuses to store and responds with abort SDO transfer (abort code: 0800002x<sub>h</sub>).

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 <sub>h</sub>	Store Parameters	Array	UNSIGNED32

Table 24: Object Description (1010<sub>h</sub>)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
01h	Save all parameters	rw	no	UNSIGNED32	—
02h	Save communication parameters	rw	no	UNSIGNED32	—
03h	Save device profile parameters	rw	no	UNSIGNED32	—
04h	Save motor axis 0 parameters	rw	no	UNSIGNED32	—

Table 25: Entry Description (1010<sub>h</sub>)



### 3.1.10 Object 1011<sub>h</sub>: 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.

There are several parameter groups:

- Sub-index 0<sub>h</sub>: contains the largest sub-index that is supported.
- Sub-index 1<sub>h</sub>: restores all parameters.
- Sub-index 2<sub>h</sub>: restores communication parameters 100C<sub>h</sub>, 100D<sub>h</sub>, 1015<sub>h</sub>, 1017<sub>h</sub>, and 1029<sub>h</sub>.
- Sub-index 3<sub>h</sub>: restores device profile parameters.
- Sub-index 4<sub>h</sub>: restores motor 0 parameters.

---

**Note** In order 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<sub>h</sub>, see also table 26).

---

Load Signature			
d	a	o	l
64 <sub>h</sub>	61 <sub>h</sub>	6F <sub>h</sub>	6C <sub>h</sub>

Table 26: 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<sub>h</sub>). If a wrong signature is written, the device refuses to restore and responds with abort SDO transfer (abort code: 0800002x<sub>h</sub>).

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 have been restored they will become active after the next rest or power cycle of the TMCM-1240.

Object Description			
Index	Name	Object Type	Data Type
1011 <sub>h</sub>	Restore parameters	Array	UNSIGNED32

Table 27: Object Description (1011<sub>h</sub>)



Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
01h	Restore all parameters	rw	no	UNSIGNED32	—
02h	Restore communication parameters	rw	no	UNSIGNED32	—
03h	Restore device profile parameters	rw	no	UNSIGNED32	—
04h	Restore motor axis 0 parameters	rw	no	UNSIGNED32	—

Table 28: Entry Description (1011<sub>h</sub>)

### 3.1.11 Object 1014<sub>h</sub>: COB-ID Emergency Object

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

Object Description			
Index	Name	Object Type	Data Type
1014 <sub>h</sub>	COB-ID emergency object	Variable	UNSIGNED32

Table 29: Object Description (1014<sub>h</sub>)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	rw	no	UNSIGNED32	80 <sub>h</sub> + Node ID

Table 30: Entry Description (1014<sub>h</sub>)

### 3.1.12 Object 1015<sub>h</sub>: Inhibit Time EMCY

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

Object Description			
Index	Name	Object Type	Data Type
1015 <sub>h</sub>	COB-ID emergency object	Variable	UNSIGNED16

Table 31: Object Description (1015<sub>h</sub>)

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

Table 32: Entry Description (1015<sub>h</sub>)



### 3.1.13 Object 1016<sub>h</sub>: 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 33: Value Definition (1016<sub>h</sub>)

Object Description			
Index	Name	Object Type	Data Type
1016 <sub>h</sub>	Consumer heartbeat time	Variable	UNSIGNED16

Table 34: Object Description (1016<sub>h</sub>)

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

Table 35: Entry Description (1016<sub>h</sub>)

### 3.1.14 Object 1017<sub>h</sub>: 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 <sub>h</sub>	Producer heartbeat time	Variable	UNSIGNED16

Table 36: Object Description (1017<sub>h</sub>)





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

Table 37: Entry Description (1017<sub>h</sub>)

### 3.1.15 Object 1018<sub>h</sub>: Identity Object

The object 1018<sub>h</sub> contains general information about the device:

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

Object Description			
Index	Name	Object Type	Data Type
1018 <sub>h</sub>	Identity object	Record	Identity

Table 38: Object Description (1018<sub>h</sub>)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
00 <sub>h</sub>	Number of entries	ro	no	0...3	3
01 <sub>h</sub>	Vendor ID	ro	no	UNSIGNED32	0286 <sub>h</sub>
02 <sub>h</sub>	Product code	ro	no	UNSIGNED32	1240
03 <sub>h</sub>	Revision number	ro	no	UNSIGNED32	e.g. 20003 <sub>h</sub> for version 2.3

Table 39: Entry Description (1018<sub>h</sub>)

### 3.1.16 Object 1029<sub>h</sub>: Error Behaviour

If a device failure is detected in operational state, the device can be configured to enter alternatively 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 <sub>h</sub>	Error behaviour	Array	UNSIGNED8

Table 40: Object Description (1029<sub>h</sub>)

Entry Description					
Sub-index	Description	Access	PDO Mapping	Value Range	Default Value
00 <sub>h</sub>	Number of error classes	ro	no	—	2
01 <sub>h</sub>	Communication error	rw	no	UNSIGNED8	0 (enter stopped state)
02 <sub>h</sub>	Application error	rw	no	UNSIGNED8	1 (remain in current state)

Table 41: Entry Description (1029<sub>h</sub>)

### 3.1.17 Objects 1400<sub>h</sub> – 1403<sub>h</sub>: Receive PDO Communication Parameter

This object contains the communication parameters for the RPDOs which the device is able to receive. The sub-index 00<sub>h</sub> 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 01<sub>h</sub> 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 in order 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 02<sub>h</sub> contains the transmission type of the RPDO. This can be FF<sub>h</sub> or FE<sub>h</sub> for event-driven, or 00<sub>h</sub> for synchronous.

Object Description			
Index	Name	Object Type	Data Type
1400 <sub>h</sub> – 1403 <sub>h</sub>	Receive PDO parameter	RECORD	RPDO CommPar
1400 <sub>h</sub>	RPDO 1	RECORD	RPDO CommPar
1401 <sub>h</sub>	RPDO 2	RECORD	RPDO CommPar
1402 <sub>h</sub>	RPDO 3	RECORD	RPDO CommPar
1403 <sub>h</sub>	RPDO 4	RECORD	RPDO CommPar

Table 42: Object Description (1400<sub>h</sub>)

