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!

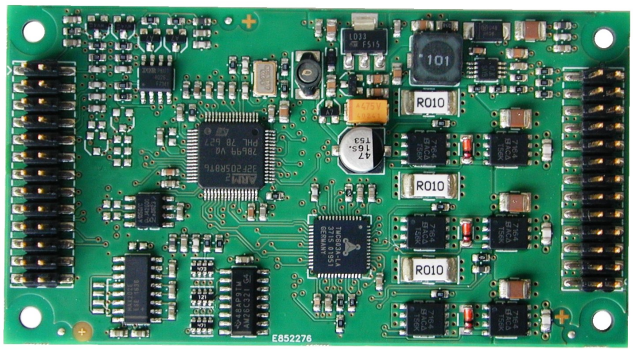
Contact us

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-1633 CANopen Firmware Manual

Firmware Version V2.10 | Document Revision V1.01 • 2018-Apr-19

The TMCM-1633 is a single axis controller module for brushless DC (BLDC) and PMSM motors. It offers field oriented control (FOC) with up-to 10A RMS phase currents at +48V DC supply. Besides hall sensor and incremental ABN encoder interfaces for connection to the motor, digital inputs and outputs can be used. A CAN interface allows communication with a CANopen master.



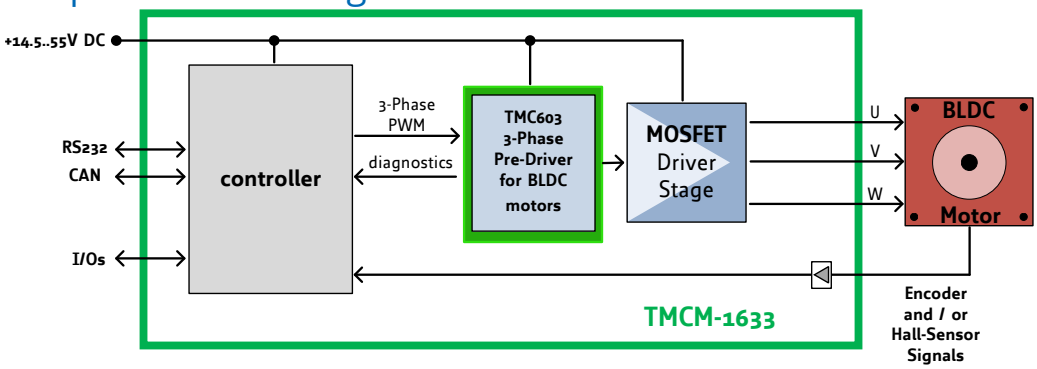
Features

- Single axis field oriented control for BLDC/PMSM motor
- Hall and ABN encoder support
- +14,5..48V DC supply voltage
- Up to 10A RMS peak motor current
- RS232 & CAN interface
- CANopen CiA 402 drive profile
- Torque, Velocity, and Position control

Applications

- Life Sciences
- Test & Measurement
- Robotics / Automation

Simplified Block Diagram



Read entire documentation.

TRINAMIC
MOTION CONTROL

Contents

1 Preface

This document specifies objects and modes of operation of the Trinamic TMCM-1633 BLDC/PMSM 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-1633 TMCL firmware again with the help of the firmware update function of the TMCL-IDE 3.0 and the UART interface.

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 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 by the user. The new firmware can be loaded into the module via the firmware update function of the TMCL-IDE, using the UART interface of the module.

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 1.

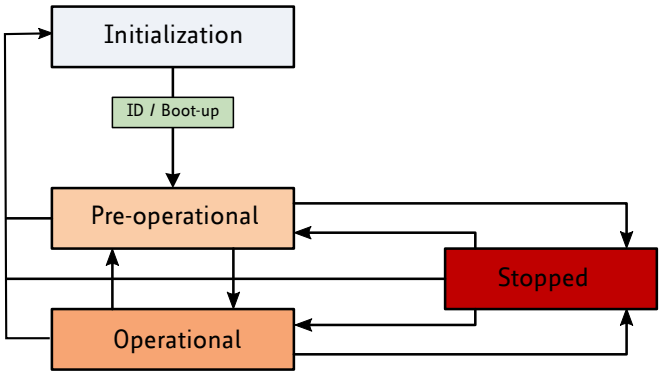


Figure 1: NMT State Machine

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.

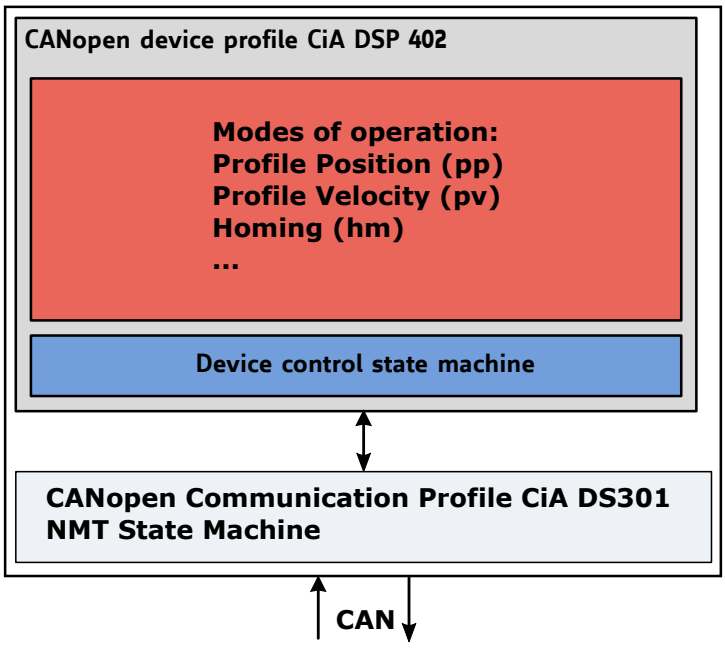


Figure 2: Communication Architecture

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.

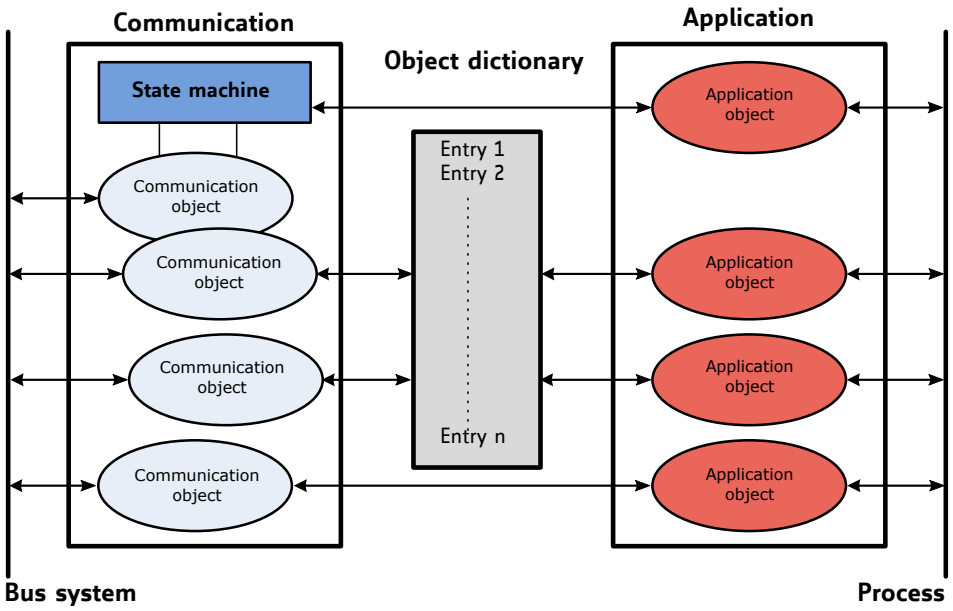


Figure 3: Device Model

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-1633 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.

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 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 5: Object Description (1000_h)

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

Table 6: 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 7: Object Description (1001_h)

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

Table 8: 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 9: Error Register Bits

3.1.3 Object 1005_h: 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_h)

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

Table 11: Object Description (1005_h)

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

Table 12: 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 13: Object Description (1008_h)

Entry Description				
Sub-index	Access	PDO Mapping	Value Range	Default Value
0	ro	no	—	TMCM-1633

Table 14: 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 15: 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 16: 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 17: 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 18: Entry Description ($100A_h$)

3.1.7 Object $100C_h$: Guard Time

The objects at index $100C_h$ and $100D_h$ 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
$100C_h$	Guard Time	Variable	UNSIGNED16

Table 19: Object Description ($100C_h$)

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

Table 20: Entry Description ($100C_h$)

3.1.8 Object $100D_h$: 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
$100D_h$	Life Time Factor	Variable	UNSIGNED8

Table 21: Object Description ($100D_h$)

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

Table 22: Entry Description ($100D_h$)

3.1.9 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.

There are several parameter groups:

- Sub-index 0_h: contains the largest sub-index that is supported.
- Sub-index 1_h: saves all parameters.
- Sub-index 2_h: saves communication parameters $100C_h$, $100D_h$, 1015_h, 1017_h, and 1029_h.
- Sub-index 3_h: saves device profile parameters.
- Sub-index 4_h: 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_h, see also table 23).

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

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_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 24: Object Description (1010_h)

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_h)

3.1.10 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.

There are several parameter groups:

- Sub-index 0_h: contains the largest sub-index that is supported.
- Sub-index 1_h: restores all parameters.
- Sub-index 2_h: restores communication parameters $100C_h$, $100D_h$, 1015_h, 1017_h, and 1029_h.
- Sub-index 3_h: restores device profile parameters.
- Sub-index 4_h: 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" ($64616F6C_h$, see also table 26).

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

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_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 have been restored they will become active after the next rest or power cycle of the TMCM-1633.

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

Table 27: Object Description (1011_h)

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_h)

3.1.11 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 29: Object Description (1014_h)

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

Table 30: Entry Description (1014_h)

3.1.12 Object 1015_h: Inhibit Time EMCY

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

Object Description			
Index	Name	Object Type	Data Type
1015_h	COB-ID emergency object	Variable	UNSIGNED16

Table 31: Object Description (1015_h)

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

Table 32: Entry Description (1015_h)

3.1.13 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 33: Value Definition (1016_h)

Object Description			
Index	Name	Object Type	Data Type
1016_h	Consumer heartbeat time	Variable	UNSIGNED16

Table 34: Object Description (1016_h)

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

Table 35: Entry Description (1016_h)

3.1.14 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 36: Object Description (1017_h)

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

Table 37: Entry Description (1017_h)

3.1.15 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 38: 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	1633
03_h	Revision number	ro	no	UNSIGNED32	e.g. 20003_h for version 2.3

Table 39: Entry Description (1018_h)

3.1.16 Object 1029_h: 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_h	Error behaviour	Array	UNSIGNED8

Table 40: Object Description (1029_h)

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

Table 41: Entry Description (1029_h)

3.1.17 Objects 1400_h – 1403_h: Receive PDO Communication Parameter

This object contains the communication parameters for the RPDOs which the device is able to receive. The sub-index 00_h 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_h 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_h contains the transmission type of the RPDO. This can be FF_h or FE_h for event-driven, or 00_h for synchronous.

Object Description			
Index	Name	Object Type	Data Type
1400_h – 1403_h	Receive PDO 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 42: Object Description (1400_h)

Entry Description				
Sub-index	Description	Access	Value Range	Default Value
00_h	Largest sub-index supported	ro	2	2
01_h	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
02_h	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 43: Entry Description (1400_h)

3.1.18 Objects 1600_h – 1603_h: Receive PDO Mapping Parameter

These objects contain the mapping parameters for the RPDOs 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 – 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 44: 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...3	Index 1600_h: 1 Index 1601_h: 2 Index 1602_h: 2 Index 1603_h: 2
01_h	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
02_h	Mapping entry 2	rw	UNSIGNED32	Index 1600_h: 0 Index 1601_h: 60600008_h Index 1602_h: $607A0020_h$ Index 1603_h: $60FF0020_h$
03_h	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 45: 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 3.1.17). Then, set its number of mapped PDO entries to zero (sub-index 0 of the appropriate PDO mapping object). Now, the mapppings themself 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.

3.1.19 Objects 1800_h – 1803_h: Transmit PDO Communication Parameter

This object contains the communication parameters for the TPDOs which the device is able to transmit. The sub-index 00_h 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 01_h 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_h contains the transmission type of the RPDO. This can be FF_h or FE_h for event-driven, or 00_h or 01_h for synchronous.

Sub-index 03_h contains the inhibit time, given in milliseconds. After a TPDO has been sent, it will not be sent again before the inhibit time has elapsed.

Sub-index 04_h is not used.

Sub-index 05_h contains the event timer value in milliseconds. When this is set to a value greater than 0 the TPDO will be sent repeatedly each time the event timer has elapsed. For example, when this value is set to 250, the TPDO will be sent every 250ms.

Object Description			
Index	Name	Object Type	Data Type
1800_h – 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 46: Object Description (1800_h)

Entry Description				
Sub-index	Description	Access	Value Range	Default Value
00_h	Largest sub-index supported	ro	5	5
01_h	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
02_h	Transmission type	rw	UNSIGNED8	Index 1800_h: FF_h Index 1801_h: FF_h Index 1802_h: 01_h Index 1803_h: 01_h
03_h	Inhibit time	rw	UNSIGNED16	0
04_h	Compatibility entry	ro	UNSIGNED8	0
05_h	Event timer	rw	UNSIGNED16	0

Table 47: Entry Description (1800_h)

3.1.20 Objects $1A00_h$ – $1A03_h$: Transmit PDO Mapping Parameter

These objects contain the mapping parameters for the TPDOs 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.