

MicroControl
Systemhaus für Automatisierung

AN1201

Introduction to CANopen

This introduction to CANopen describes the basic communications mechanisms and the usage of identifiers.

Author: Uwe Koppe
MicroControl GmbH & Co. KG

Contents

1 What is CANopen?

The CANopen protocol is a standardized layer-7 protocol for the CAN bus. The protocol CANopen, on the one hand, stipulates "how" the participants communicate with each other, i.e. which messages (or identifiers) can be used to address the devices. In CANopen, mechanisms have been implemented which define the exchange of process data in real time as well as high-volume data transmission or sending of alarm messages. On the other hand, CANopen defines "what" is communicated, i.e. a certain parameter for setting a device is addressed via a defined interface (device profile or application profile).

The CANopen profiles are organized in a table (object dictionary). All device profiles share a common "communication profile" which is used to set or query the basic device data. The data contains e.g. the device name, hardware and software version, error state, applied CAN identifiers and various other parameters. Device profiles describe the special features or parameters of a certain "type" of devices. So far, device profiles have been defined for digital or analogue I/O devices, motors, sensors and actuators, programmable controls, encoders, medical technology, public transport, batteries and extruder systems to name just a few.

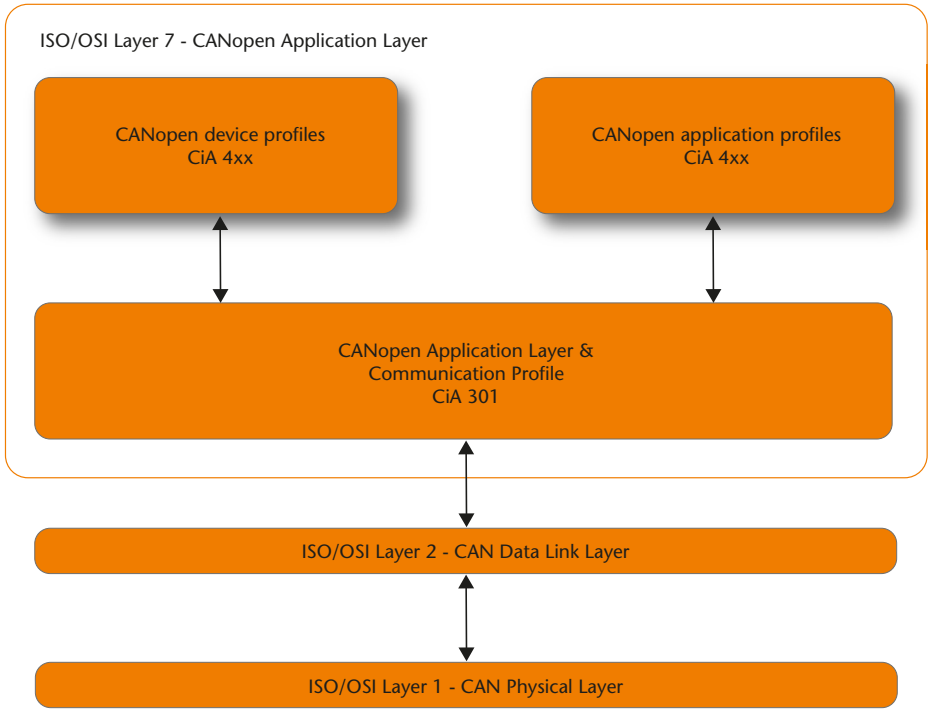

Fig. 1: Structure of CANopen device profiles

The device profiles are placed above the communication profile /1/ (see figure 1). The modular structure and the standardized format have two advantages: first, when designing a system the user only has to handle those profiles which are relevant for the intended application (e.g. digital I/O components). Second, there is a big choice of devices available in the market which are all based on this common concept and provide identical process information.

1.1 Object Dictionary

The object dictionary describes the complete functional range (parameters) of a CANopen device and it is organized in tabular format. The object dictionary contains the standardized data types, the objects of the CANopen communication profile as well as the device profiles and, as the case may be, manufacturer-specific objects and data types. The entries are addressed by means of 16-bit indices (row in the table, max. 65536 entries) and an 8-bit sub index (column in the table, max. 256 entries), thus facilitating grouping of associated objects. The structure of the CANopen object dictionary is shown in the following table:

Index (hex)	Object
0000	not used
0001 - 001F	static data types
0020 - 003F	complex data types
0040 - 005F	manufacturer-specific data types
0060 - 007F	profile-specific static data types
0080 - 009F	profile-specific complex data types
00A0 - 0FFF	reserved
1000 - 1FFF	communication profile (CiA 301 & 302)
2000 - 5FFF	manufacturer-specific parameters
6000 - 9FFF	parameters from standardized profiles (CiA 4xx)
A000 - AFFF	network variables
B000 - FFFF	reserved

Table 1: Structure of object dictionary

2 Communication Mechanisms

Communication between participants corresponds broadly with the client-server model. The process data is transmitted according to the producer-consumer model.

2.1 Service Data Objects

Service Data Objects (SDOs) are used for changing data elements inside the object dictionary and for querying the device status. Each CANopen device is equipped with at least one SDO channel and two CAN identifiers assigned to it. This protocol can transmit data sets of any length, if necessary, the data will be divided (segmented) into several CAN messages. In the first CAN message of a SDO four out of the eight bytes available hold protocol information. To access object dictionary entries of up to 4 byte (e.g. Integer16, Integer32, Float) one single CAN message will be sufficient (Expedited Transfer). Data sets longer than 4 bytes are sent in segmented transmission where all segments of the SDO following the first CAN message can contain 7 bytes of user data each. The last segment contains an end identifier. Only confirmed SDOs are transmitted, i.e. the recipient will acknowledge reception of each message. The introduction of the CANopen specification 4.0 also allows faster SDO transfer (block transfer). Block transfer implies that the system will no longer acknowledge every single segment, but will acknowledge reception of segment groups, thus, considerably increasing the band width and facilitating the transmission of high data volumes.

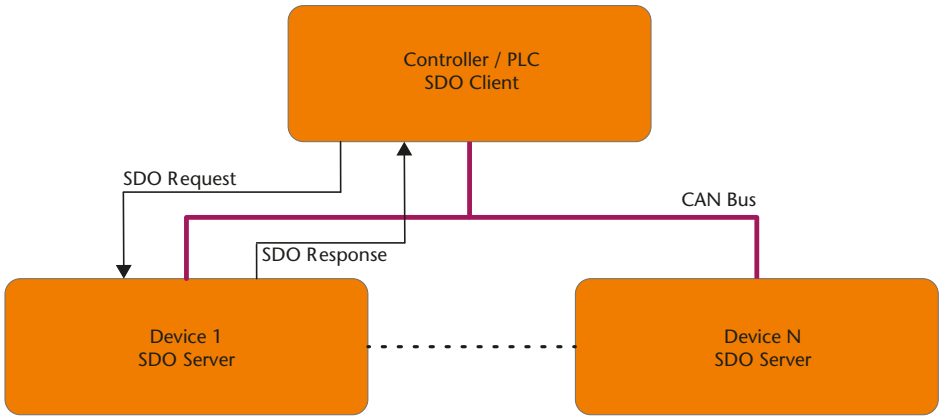

Fig. 2: SDO client-server structure

2.2 Process Data Objects

Transmission of process data follows the Process Data Object (PDO) mechanism. Each CANopen device which produces and / or consumes process data has at least one PDO. In a PDO, all 8 data bytes of a CAN message can be used individually. The transmission of a PDO will not be acknowledged as, in the end, the CAN data link layer will ensure accurate transmission of a message. Also, acknowledged messages are not desired in time-critical applications as they significantly reduce the band width of the bus, i.e. PDOs are "pure CAN" without any protocol overhead by CANopen!

Process data can be transmitted in different ways:

Event

Sending of a PDO is triggered by an external or internal event. This event e.g. can be a change in the potential of a digital input or the expiration of a timer in the device.

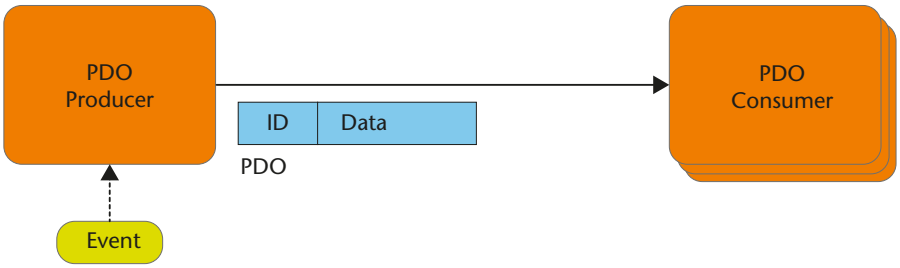

Fig. 3: Event-driven PDO

Synchronous

In synchronous transmission a bus device sends synchronizations messages (message without data content) which causes a PDO producer to transmit process data.

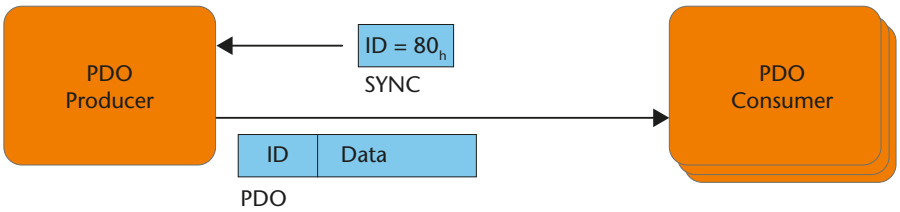

Fig. 4: Synchronous PDO transmission

2.3 PDO-Mapping

As described above, when transmitting process data all 8 data bytes can be used. As there is not any protocol information available, the format of the communication between producer and consumer has to be defined through PDO mapping. The system can only transmit the data which is available in the object dictionary of the device!

Static mapping means that the process data in the PDO message is arranged in a pre-defined sequence. The user cannot change the sequence which is defined by the manufacturer of the device.

Variable mapping means that the process data can be arranged at random within the PDO message. To do so, the entries in the object dictionary regarding address (i.e. index and sub-index) as well as the size (number of bits) are entered in the mapping object.

The following example shows how to enter the objects "Error Register" (1001_h:00_h) and "Digital input 1..8" (6000_h:01_h) in the mapping table. As soon as a PDO has to be sent, the data of the objects is copied into the PDO message.

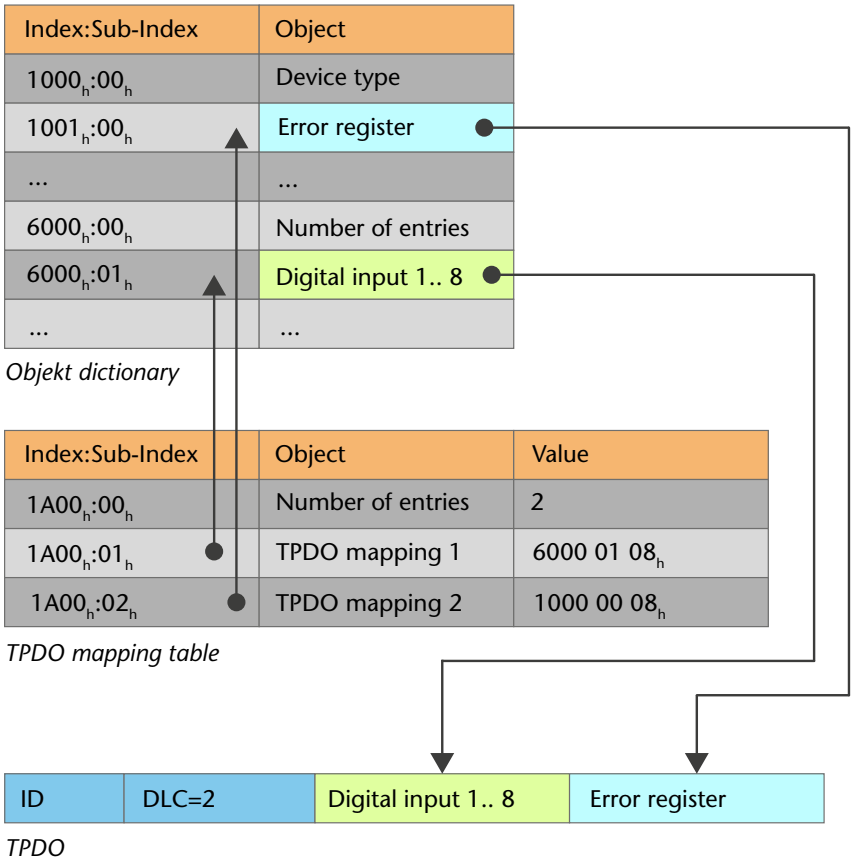

Fig. 5: How to use the PDO mapping table

2.4 Network Management

In a CANopen network there is only one NMT manger (NMT = network management), all other devices are NMT server. The NMT manager controls all other devices and can change their state to enable communication.

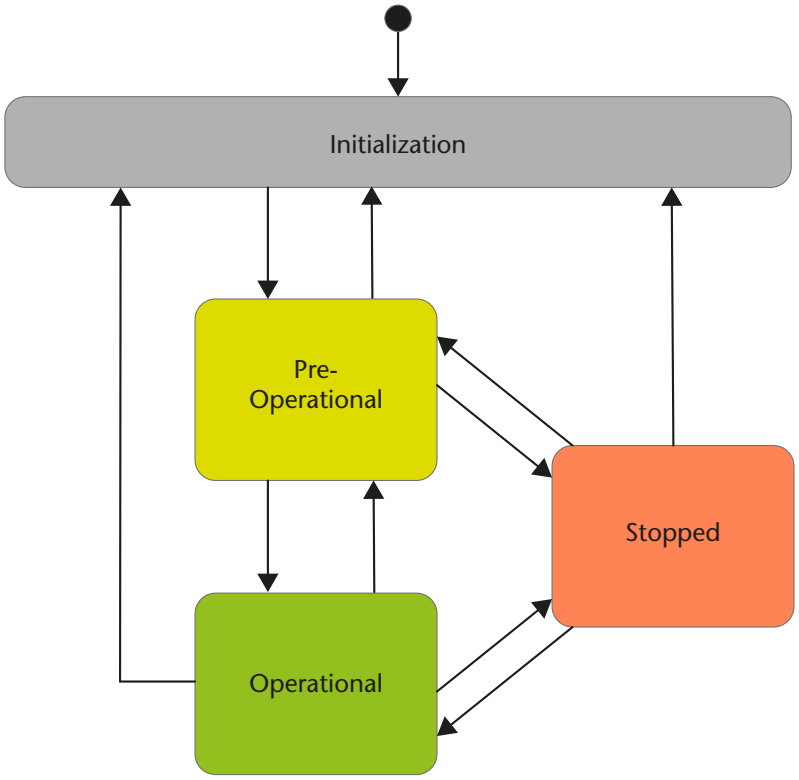

Fig. 6: NMT state diagram

With introduction of inclusive language into the CANopen standards the term *Master* has been replaced by *Manager* as well as the term *Slave* is now replaced by *Server*.

Initialization — This is the state of the node during the start-up process. In this process the device application as well as the device communication (bit rate and node address) are initialized. After start-up the node switches to pre-operational state automatically.

Pre-Operational — In "Pre-Operational" state the node can communicate via SDOs. However, the node is not able to communicate via PDOs.

Operational — In "Operational" state the node is fully ready to operate and can send and receive process data autonomously.

Stopped — In the "Stopped" state the node is completely disconnected from the network and can neither communicate via SDOs nor PDOs. The node can only be switched to a different network state through a corresponding network command (e.g. Start-Node).

To indicate that the device is ready for operation after start-up a "Boot-Up Message" will be sent. This message uses the identifier of the NMT-Error-Control protocol (Table 3 page 10) and is closely linked to the set device address (700_h + device address).

2.5 Node-Guarding and Heartbeat

Checking of a CAN node is of special importance where the node does not send continuous messages (cyclic PDOs). To monitor a CANopen node there are two mechanisms which can be used alternatively. In the node-guarding protocol the NMT manager sends messages to the existing CANopen devices which these have to respond to within a certain time period. A possible black-out of a node will, therefore, only be registered by the NMT manager. Also, the node-guarding is based on CAN remote frames and the user is recommended to refrain from using it /3/. We recommend to use the Heartbeat protocol which has been published in the CANopen version 4.0. In the heartbeat mechanism each node autonomously sends a message in cyclic intervals. The message can be monitored by any device in the network.

2.6 Emergency Messages

Messages of the "Emergency" type are used to indicate errors in the operation of the device. With the emergency message a code is transmitted which allows clear error identification (defined according to the communication profile CiA 301/1/ as well as the respective device profiles CiA 40x).

Code (hex)	Identification
00xx	no error
10xx	undefined error type
20xx	current error
30xx	voltage error
40xx	temperature error
50xx	Error in hardware
60xx	Error in software
70xx	additional devices
80xx	communication
90xx	external error
FF00	device specific

Table 2: Code of emergency messages

The table shows an excerpt of available error codes. Each CANopen device sends the emergency message autonomously. In the current version of the CANopen communication profile the emergency message can also be switched off.

3 Distribution of Identifiers

Basically, in communication via CANopen identifiers with a length of 11 bits (standard frames) are used. The available number of possible identifiers is separated into different sections by the Pre-defined Connection Set. The distribution of identifiers is designed in such a way that a CANopen network can comprise a maximum of 127 devices.

Identifier	Service	Direction	COB-ID calculation	Note
000_h	NMT	Receive	-	static
080_h	SYNC	Rcv. / Trm	-	variable, index 1005_h
081_h .. $0FF_h$	EMCY	Transmit	080_h + Node-ID	variable, index 1014_h
100_h	TIME	Rcv. / Trm.	-	variable, index 1012_h
181_h .. $1FF_h$	TPDO1	Transmit	180_h + Node-ID	variable, index 1800_h
201_h .. $27F_h$	RPDO1	Receive	200_h + Node-ID	variable, index 1400_h
281_h .. $2FF_h$	TPDO2	Transmit	280_h + Node-ID	variable, index 1801_h
301_h .. $37F_h$	RPDO2	Receive	300_h + Node-ID	variable, index 1401_h
381_h .. $3FF_h$	TPDO3	Transmit	380_h + Node-ID	variable, index 1802_h
401_h .. $47F_h$	RPDO3	Receive	400_h + Node-ID	variable, index 1402_h
481_h .. $4FF_h$	TPDO4	Transmit	480_h + Node-ID	variable, index 1803_h
501_h .. $57F_h$	RPDO4	Receive	500_h + Node-ID	variable, index 1403_h
581_h .. $5FF_h$	SDO	Transmit	580_h + Node-ID	static
601_h .. $67F_h$	SDO	Receive	600_h + Node-ID	static
701_h .. $77F_h$	NMT-EC	Transmit	700_h + Node-ID	static
7E4h .. 7E5h	LSS	Rcv. / Trm.	-	static

Table 3: Pre-defined Connection Set

Typically, a CANopen manager is realised through a PLC or a PC. The CANopen devices can have an address from 1 to 127. The device address (which in many devices is set via DIP switches) automatically stipulates the number of identifiers which are occupied by the device. A device with the module address 3 would occupy the following identifiers (refer to the calculation formula in Table 3 page 10):

Identifier	Function	Acronym
000_h	Network Management	NMT
080_h	Synchronization Message	SYNC
083_h	Emergency Message	EMCY
183_h	Transmit PDO 1	TPDO1
203_h	Receive PDO 1	RPDO1
283_h	Transmit PDO 2	TPDO2
303_h	Receive PDO 2	RPDO2
383_h	Transmit PDO 3	TPDO3
403_h	Receive PDO 3	RPDO3
483_h	Transmit PDO 4	TPDO4
503_h	Receive PDO 4	RPDO4
583_h	Transmit SDO	SDO Response
603_h	Receive SDO	SDO Request
703_h	Node-Guarding / Heartbeat	NMT-EC

Table 4: How to use CAN identifiers in a CANopen device address 3

This setting may be changed if required. For the user, the Pre-defined Connection Set facilitates setting up a CANopen network and it is impossible to assign two identical identifiers within the network.

4 Device Description - EDS and DCF

The "Electronic Data Sheet" (EDS) /2/ describes the functions of a CANopen device in machine-readable format. The EDS contains all objects, the supported bit rates, the manufacturer and other specifications. The EDS, however, is only a template for the device as it does not contain the values of an object.

The "Device Configuration File" (DCF) is identical to the EDS but it also contains the values of each object.

The format of both description files is similar to the *.ini format of Windows. For the user, the integration of a new device into the network is fairly easy: Connect hardware and read disk with the pertaining EDS or DCF file into the manager. Now the device is ready for operation.

```
.....

[MandatoryObjects]
SupportedObjects=3
1=0x1000
2=0x1001
3=0x1018

[1000]
ParameterName=DeviceType
ObjectType=0x7
DataType=0x7
AccessType=ro
DefaultValue=0x00020194
PDOMapping=0

[1001]
ParameterName=ErrorRegister
ObjectType=0x7
DataType=0x5
LowLimit=0x0
HighLimit=0xff
AccessType=ro
DefaultValue=0x0
PDOMapping=0

[1018]
SubNumber=5
ParameterName=Identity Object
ObjectType=0x9

[1018sub0]
ParameterName=Number of entries
ObjectType=0x7
DataType=0x5
.......
```

Fig. 7: Excerpt from an EDS file

5 Conclusion

CANopen is constantly advanced by the members of the international user and manufacturer association CAN in Automation e.V. (https://www.can-cia.org). The CANopen application layer defines different communication services and protocols (e.g. process and service data) as well as the network management.

CANopen is mainly used as an embedded network in machinery, but also as a general industrial communications system. It is based on the international standards ISO 11898-1 (CAN protocol) and ISO 11898-2 (fast physical layer). The CANopen application layer has been standardized by the CENELEC and is defined in the European standard EN 50325-4.

If you would like to get further information about CANopen (or participate in the development of profiles), you will be welcome to download for free all public specifications from the homepage of the CAN in Automation association /4/.

6 References

/1/ CiA 301: CANopen application layer and communication profile, Version 4.2.0, CAN in Automation
https://www.can-cia.org

/2/ CiA 306: CANopen electronic data sheet specification, Version 1.3.0, CAN in Automation
https://www.can-cia.org

/3/ CiA 802: CANopen application note – CAN remote frames: Avoiding of usage, Version 1.1.0, CAN in Automation
https://www.can-cia.org

/4/ CAN in Automation e.V.: CiA specifications
https://www.can-cia.org/cia-groups/technical-documents

7 Document Versions

Revision	*Date*	*Description*
01	01.03.2000	First version
02	18.08.2014	Revision according to CiA 301, version 4.2.0
03	24.05.2024	update figures, inclusive language

Disclaimers

Life support — Products and software described in this application note are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury. MicroControl customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify MicroControl for any damages resulting from such application.

Right to make changes — MicroControl reserves the right to make changes in the products - including circuits and/or software - described or contained herein in order to improve design and/or performance. MicroControl assumes no responsibility or liability for the use of any of these products, conveys no licence or title under any patent, copyright, or mask work right to these products, and makes no representations or warranties that these products are free from patent, copyright, or mask work right infringement, unless otherwise specified.

Copyright

MicroControl
Systemhaus für Automatisierung

MicroControl GmbH & Co. KG
Junkersring 23
53844 Troisdorf
Germany
Fon: +49 / 2241 / 25 65 9 - 0
Fax: +49 / 2241 / 25 65 9 - 11
http://www.microcontrol.net