CANopen Communication Profile Features

CANopen represents a standardized application of the CAN Application Layer (CAL) specification. Both of these specifications [1, 2] were developed by members of CAN in Automation (CiA), the international users' and manufacturers' association, and they may be implemented licence-free. The CANopen Communication profile is described in CiA Draft Standard CiA 301. The CANopen Communication profile is supplemented by a number of device profiles, for instance for I/O modules. Implementations must include all functions defined as "mandatory" and any "optional" functions implemented must be implemented as described in the CANopen specification. It is only for manufacturer-specific functions that the implementor may give free rein to his creativity. CANopen assumes that the device's hardware has a CAN transceiver and CAN protocol controller as specified in ISO 11898.

The CANopen specification describes the minimum functionality that a CANopen device must provide. Such "Minimum Capability Devices" are not particularly useful as a standard product but many functions are classified as optional in order to permit a "lean" implementation for special applications.

Index (hex)	Object
0000	not used
0001-001F	Static Data Types
0020-003F	Complex Data Types
0040-005F	Manufacturer Specific Data Types
0060-007F	Device Profile Specific Static Data Types
0080-009F	Device Profile Specific Complex Data Types
00A0-0FFF	Reserved for further use
1000-1FFF	Communication Profile Area
2000-5FFF	Manufacturer Specific Profile Area
6000-9FFF	Standardised Device Profile Area
A000-FFFF	Reserved for further use

Table 1: CANopen Object Dictionary

All CANopen devices must support the CANopen object dictionary (Table 1). This dictionary, which is very similar to the Profibus and Interbus-S object dictionaries, contains all relevant CANopen objects for this device. Read and write accesses are by means of a Service Data Object (SDO). SDOs are used for modifications to the object dictionary and for status interrogation. Each CANopen device has at least one SDO, to which two CAN identifiers are assigned. SDOs use the CAL specification's "Multiplexed Domain Transfer Protocol". This permits transfer of data of any length since the data can be split up over several CAN messages (segmented). Protocol information occupies four of the eight bytes of the SDO's first CAN message. This means that accesses to object dictionary entries with a length of up to four bytes require only a single CAN message (expedited transfer). For data lengths in excess of four bytes, a segmented transfer is performed, and all segments after the SDO's first CAN message may each contain seven bytes of useful data. The last segment contains an end indicator. An SDO is transferred in "confirmed" mode, that is, the reception of each segment is acknowledged by a corresponding CAN message.

For transferring process data, the Process Data Object (PDO) mechanism is provided. Therefore, each CANopen device that produces or consumes process data has at least one PDO. PDOs use the "Stored Event Protocol" of the CAN Application Layer. All eight data bytes of a CAN telegram are available to the user for transferring application data.. PDOs are transferred "unconfirmed", since the CAN data link layer ensures error-free transfer of a CAN frame and confirmed services are not desirable in time-critical applications because they significantly reduce the bus bandwidth. So PDOs are "straight" CAN without any protocol overhead caused by CAL/CANopen. The data contents of the PDOs are specified in the corresponding CANopen device profiles (see “CANopen Device and Application Profiles” box).

The CANopen specification defines a basic set of connections as a default, the Pre-Defined Master/Slave Connection Set. CAN identifiers are assigned automatically using the node number defined by the system integrator and the default setting for the objects defined in the CANopen specification (Table 2a and 2b). PDOs have a higher priority than SDOs. The object with the highest priority is reserved for network management (NMT) and is used to start and stop the CANopen network. The other defined objects, e.g. Emergency Message, Sync Message and Timestamp Telegram, are optional. The use of these pre-defined connections requires a master not only on the network-management side but also on the process data side, since all PDOs use different CAN identifiers and thus have no pre-defined connections. However, this is well-suited to small applications, where, for instance, network management and a control application both reside on a central PC. On the other hand, a user wanting to define more flexible connections for the process data must implement one of the optional methods for assigning identifiers, as described below.

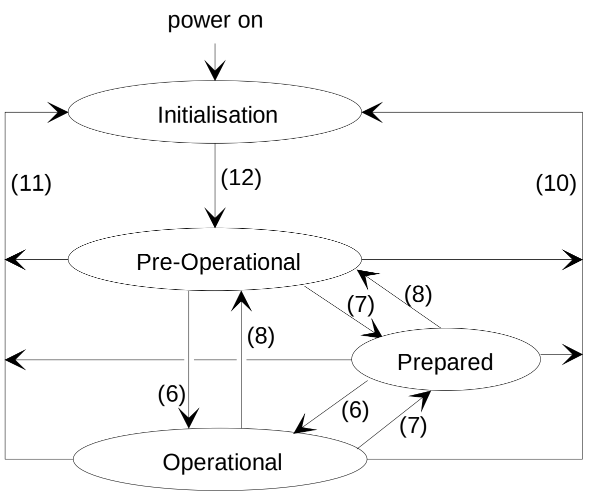


Fig. 1: State diagram of a simple CANopen boot up

object	function code (binary)	resulting COB-ID	Communication Parameters at Index	CMS priority group
NMT	0000	0	-	0
SYNC	0001	128	(1005h)	0
TIME STAMP	0010	256	-	1

Table 2a: Broadcast Objects of the Pre-defined Master/Slave Connection Set

object	function code (binary)	resulting COB-IDs	Communication Parameters at Index	CMS priority group
EMERGENCY	0001	129 - 255	-	0 , 1
PDO1 (tx)	0011	385 - 511	1800h	1 , 2
PDO1 (rx)	0100	513 - 639	1400h	2
PDO 2 (tx)	0101	641 - 767	1801h	2 , 3
PDO2 (rx)	0110	769 - 895	1401h	3 , 4
SDO (tx)	1011	1409 - 1535		6
SDO (rx)	1100	1537 - 1663		6 , 7
Nodeguard	1110	1793-1919	(100Eh)	-

Table 2b: Peer-to-Peer Objects of the Pre-defined Master/Slave Connection Set

CANopen Device and Application Profiles

- CiA DSP-401: Device profile for I/O modules
- CiA DSP-402: Device profile for drives and motion control
- CiA WDP-403: Device profile for human machine interfaces
- CiA WD-404: Device profile for measuring devices and closed-loop controllers
- CiA WD-405: Device profile for IEC-1131 Interfaces
- CiA DSP: Device profile for encoders
- CiA WDP: Application profile for public transport
- CiA WDP: Application profile for fork-lifts

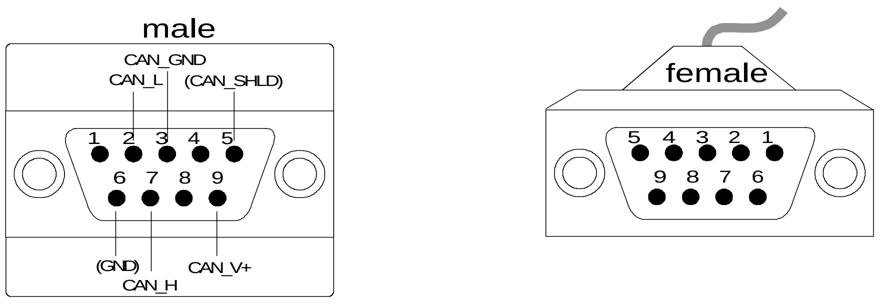
After the supply voltage is switched on, minimum capability devices automatically enter the "initialization" state, followed by the "pre-operational" state. In this state the device can communicate with the CANopen device responsible for network management (NMT Master) only by means of SDOs. Only when the NMT Master has transmitted an appropriate CAN message (NMT Start Remote Node) does the minimum capability device (NMT slave) enter the "operational" state and actual communication, that is transfer of PDOs, can begin. Every CANopen device must support all states shown in Fig. 1 and the corresponding services to achieve these states. In the "prepared" state the device no longer takes part in PDO communication. This state can be used for application-specific purposes. The specific behaviour in this state is defined in the corresponding CANopen device profiles.

The physical bus interface of CANopen devices is as specified in ISO 11898. In addition, several baud rates and corresponding bit-timing settings are recommended (Table 3). However, every CANopen device must support 20-kBit/s transfer. The pin assignments and designations for different connectors are also defined (Fig. 2). The

CANopen specification allows the system integrator a great deal of freedom in defining the mechanical and electromechanical components, without endangering interoperability.

Bit rate Bus length [1]	Nominal bit time t_b	Number of time quanta per bit	Length of time quantum t_q	Location of sample point
1 Mbit/s 25 m	1 µs	8	125 ns	6 t_q (750 ns)
800 kbit/s 50 m	1.25 µs	10	125 ns	8 t_q (1 µs)
500 kbit/s 100 m	2 µs	16	125 ns	14 t_q (1.75 µs)
250 kbit/s 250 m [2]	4 µs	16	250 ns	14 t_q (3.5 µs)
125 kbit/s 500 m [2]	8 µs	16	500 ns	14 t_q (7 µs)
50 kbit/s 1000 m [3]	20 µs	16	1.25 µs	14 t_q (17.5 µs)
20 kbit/s 2500 m [3]	50 µs	16	3.125 µs	14 t_q (43.75 µs)
10 kbit/s 5000 m [3]	100 µs	16	6.25 µs	14 t_q (87.5 µs)

Table 3: Recommended baud rates and bit timings



Pin	Signal	Description
1	-	Reserved
2	CAN_L	CAN_L bus line (dominant low)
3	CAN_GND	CAN Ground
4	-	Reserved
5	(CAN_SHLD)	Optional CAN Shield
6	(GND)	Optional Ground
7	CAN_H	CAN_H bus line (dominant high)
8	-	Reserved
9	(CAN_V+)	Optional CAN external positive supply (dedicated for supply of transceiver and optocouplers, if galvanic isolation of the bus node applies)

Fig. 2: Pin assignments for D-Sub connectors

Pre-defined Objects

One of the pre-defined CANopen objects is the Sync object, which can be used only for synchronizing the network devices. This message, implemented as a “CMS Basic Variable” in accordance with CAL standards, is particularly relevant for drive applications, and it should therefore be supported by all CANopen devices that must be synchronized with drives. The cyclically transmitted Sync telegram causes the CANopen devices that support this function to store the actual values of the inputs quasi-simultaneously. They are then transferred to all interested devices in the following time frame (Fig. 3). In the following time frame, the CANopen devices transmit the output values to the actuators. However, these values are not valid until the next Sync message is received. In the two specified time frames and any remaining time frame within the communication cycle, lower-priority asynchronous PDOs and SDOs can be transferred.

The “Time Stamp Object” (CMS Stored Event according to CAL), which is also optional, refers to a general time base and is particularly suitable for timing tasks in data capture systems. The third pre-defined message is the “Emergency Object”. Implementation of this optional object is strongly recommended for all devices with a major function in the application. Also, some

CANopen device profiles support this object in order to provide high-priority notification of fatal device errors to other devices. The Emergency message is used by the device to signal an internal error.

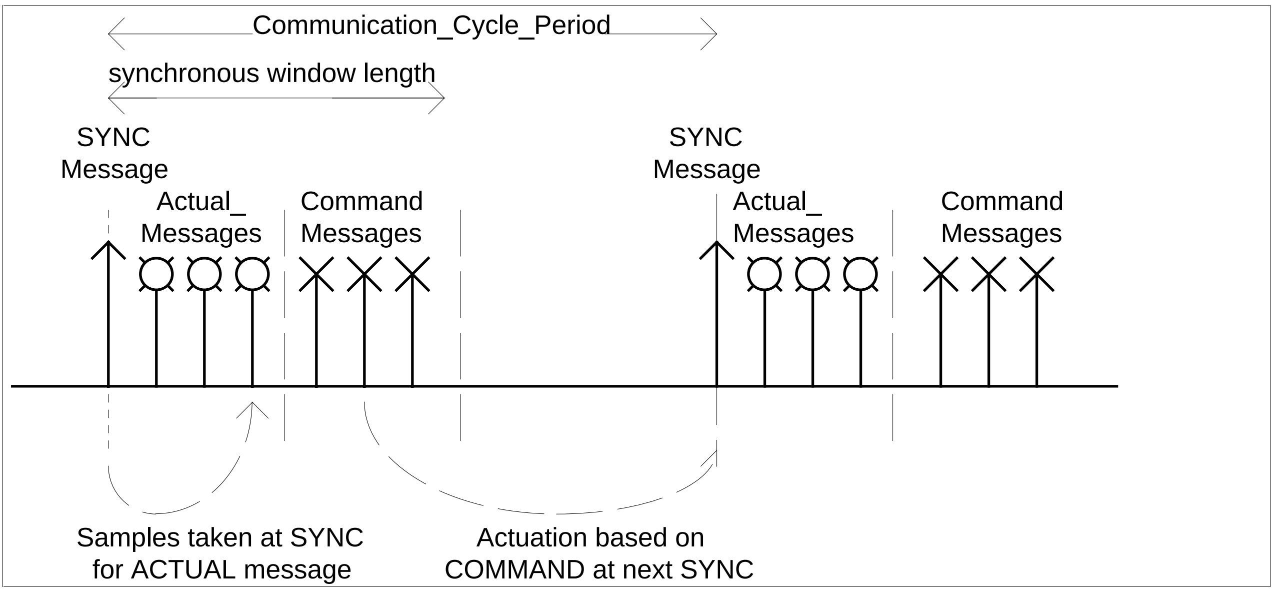

Communication_Cycle_Period
synchronous window length
SYNC Message
Actual_ Messages
Command Messages
SYNC Message
Actual_ Messages
Command Messages
Samples taken at SYNC for ACTUAL message
Actuation based on COMMAND at next SYNC

Fig. 3: Synchronous data transfer

Optional Node Guarding

The CANopen communication interface normally assumes that any connected device is functioning correctly and is able to exchange messages with other devices. However, a situation can arise where a device is no longer functioning correctly and enters the “disconnected” state. In order to detect this the NMT Master in the CANopen network should maintain a data base containing entries for all NMT slaves that support what is known as “Node/Life Guarding”. This monitoring, which is very strongly recommended for asynchronous message exchange, starts with the first Remote Transmit Request after the Guarding Identifier is transferred by the NMT master.
With Cyclic Node Guarding the NMT master regularly polls its NMT slave to request its status. The NMT slaves participating in the guarding process test internally whether the “Node Guarding” is taking place in the defined time interval (Life Guarding). This is necessary in order to determine whether the NMT Master is still alive.

Optional Variable PDO Mapping

The PDOs defined in the various standardized or manufacturer-specific CANopen device profiles usually contain several application objects, e.g. an analog input value, 8 digital input values and 8 digital output values. This assignment of standardized objects to the PDOs is known as Default PDO Mapping. To permit PDOs to be modified by the system integrator or allow the definition of special PDOs optimized for the specific application the device must support Variable PDO Mapping. This is not necessary for simple devices. In this case the PDO Mapping entries in the object dictionary are read-only.

Optional CAL Boot-Up

The “natural” form of boot-up in a CANopen network is doubtless the minimum boot-up. The more complicated CAL boot-up has its origins in a time when the powerful instrument of the object dictionary was not yet available to provide for parametrization. In some applications, however, an extended boot-up in accordance with the CAL specification is required. This is the case in all CAL systems that do not have a CANopen profile extension. It is also the case for CANopen applications that are going to use the DBT function. When the network is initialised, the Distributor (DBT) dynamically assigns the CAN identifiers. A detailed description of the DBTs is included in [2].

Identifier assignment via SDO

The assignment of CAN identifiers via SDO messages is also optional. This assignment is performed in the “pre-operational” state. At this time, the Configuration Master has access to the object dictionaries of all CANopen devices. This optional function is required, for instance, for Variable PDO Mapping. There are now a number of configuration tools on the market for setting PDO mappings and PDO identifiers.

Electronic Data Sheet

The “Electronic Data Sheet (EDS)” described in the appendix of the CANopen specification should be supported by the manufacturer so that CANopen configuration tools can read all information concerning a device without unnecessary effort. An EDS is only a template. If all CANopen information (in other words, the object dictionary) is entered in the EDS, we call it the Device Description File (DCF). The DCF includes not only the objects but also their values. So it is the incarnation of the EDS. The DCF also contains the values for baud rates and device identification. If a manufacturer does not provide an EDS, the system integrator can use a default EDS, which he must then, of course, fill out himself.

References

[1] CiA DS-301, V3.0: CANopen Communication Profile. October 1996.
[2] CiA DS-201 to DS-207, V1.1: CAN Application Layer. February 1996.

Framework for Programmable CANopen Devices

The CANopen Communication Profile (CiA DS-301) defines the basic communication mechanisms for exchanging data. In general the mechanisms which are specified in the communication profile are sufficient for the definition of profiles for devices which, on the application level, provide some kind of I/O functionality.
For the description and operation of programmable devices further mechanisms are necessary which will be specified in CiA DS-302. This specification has to be regarded as a framework for the definition of device profiles for intelligent or programmable devices in form of an extension to the communication profile DS-301. The additional mechanisms specified in DS-302 are useful especially for intelligent devices like PLCs, MMIs or CANopen tools.
The framework comprises the following mechanisms and definitions:

- The dynamic establishment of SDO connections between devices. Dynamic SDO connections are handled by the SDO Manager.
- The term CANopen Manager is introduced to specify more clearly the *network functionality* of a network controlling device.
- The definition of dynamically allocated entries in an object dictionary which can be used for the representation of I/O data e.g. on programmable nodes like PLCs.
- A general mechanism for downloading program data and functions for the control of programs on a device.
- A possibility for detecting and configuration of unconfigured nodes during system boot-up by means of a Configuration Manager.
- A debugging mechanism in the form of an OS command and prompt.
- A multiplexed PDO which allows to write data of object dictionary entries on a group of nodes simultaneously. The multiplexed PDO also has non-group applications.

Some of these new mechanisms are also useful not only for intelligent or programmable devices. Therefore it is expected, that these probably will be included in a future revision of DS-301.

CAN in Automation (CiA) International Users and Manufacturers Group
Am Weichselgarten 26 ◆ D-91058 Erlangen
Phone +49-9131-69086-0 ◆ Fax +49-9131-69086-79
Email: headquarters@can-cia.de ◆ URL: http://www.can-cia.de