CANopen Devices becoming intelligent with IEC 1131-3

Dipl.-Ing. (FH) Hansjürgen Eberle
IXXAT Automation GmbH, Weingarten, Germany

Prof. Dr.-Ing. Konrad Etschberger
stzp, Weingarten, Germany

Abstract

Shortly after availability, the CANopen standard for industrial automation became widely accepted within a wide range of applications. With the newest extensions of the standard, also IEC-1131 programmable devices in distributed, intelligent automation systems are supported.

When merging CANopen into IEC 1131-based PLCs, facilities have to be provided for downloading and controlling of programs and tasks and for the dynamic mapping of PLC network variables into the CANopen Object Dictionary of the controller device. For program debugging via the CAN bus also an appropriate debugging channel has to be provided. Furthermore, there are specific CANopen functions like network, configuration and SDO management which have to be accessible and controllable from the PLC kernel.

In the paper, the main results of the CiA special interest groups "Framework for programmable CANopen Devices" and "CANopen Interface Profile for IEC 1131-3 Environments" will be summarized and the basic principle for the implementation of a IEC 1131-3 programmable CANopen-based PLCs will be described. Also, an integrated CANopen system configuration and programming tool will be presented shortly.

1 Introduction

The well-known advantages of the CANopen approach together with the availability of standardized device profiles[1] made CANopen in only two years to the most accepted solution for CAN-based automation systems in Europe.

As the common interface between communication and application processes, the Object Dictionary (OD) of a CANopen device provides a standardized access to all of its parameters, functions and data which are accessible from the CAN bus.

The address of a specific Object Dictionary entry is given by a 16-bit object address (index) and a 8-bit subobject identifier (subindex). Accessing of an OD entry is performed by means of a so-called Service Data Object (SDO), in which the object address is specified. The SDO transfer mechanism supports a peer-to-peer connection between two devices in a master-slave relationship, and provides a direct read/write access to each entry of a slave Object Dictionary.

Normally the transfer of SDOs takes place only during system setup for transferring of device configuration data. Additionally, the transmission of rarely used low priority process data is also possible via SDO transfer.

For transferring of time critical process data, the so-called Process Data Objects (PDOs) are provided. These allow an unconfirmed transfer of data up to 8 bytes without any protocol overhead in form of broadcast messages.

The data content of a PDO is specified by its ‘PDO-mapping’ which describes the assembling of different application data into a PDO. The transmission of a PDO is possible via several alternative transmission modes like asynchronous, acyclically synchronous or cyclically synchronous transmission.

With the synchronous transmission modes and an additional system time object, advanced synchronization and time stamping mechanisms for network wide synchronous time and data distribution are supported.

CANopen also provides functions for supervising and controlling of devices. For this purpose, one device in the network has to take over the network management mastership. Simple CANopen devices only have to support the NMT slave functionality.

2 Specification of a Framework for Programmable CANopen Devices

With respect to the requirements of advanced distributed industrial automation systems, the draft standard proposal DSP 302, ‘Framework for programmable CANopen devices’[2], specifies the required improved flexibility and functionality. The additional features supported by programmable CANopen devices are:

- Dynamic establishment of SDO connections between two devices by means of a SDO Manager. This instance is responsible for managing of all of the available and established SDO connections in a network.
- Storage of the system configuration and parametration data of the network by means of a Configuration Manager.
- Transmission of multiplexed broadcasting of data with message grouping by means of ‘Multipexed PDOs’.
- Support of a standardized program debugging interfaces via the CAN bus. For that purpose, some additional Object Dictionary entries have been reserved.
- Support of a standardized mechanism for downloading of the application program, based on a basic CANopen operating system.

3 A Standard Profile for IEC 1131-3 Programmable CANopen Devices

Based on the well established IEC-1131-3 standard for PLC programming languages and the multi-master capability of the CAN protocol, the draft standard proposal DSP 405 [3] provides a future-oriented solution for the implementation of IEC 1131-3 compliant programmable CANopen devices.

3.1 Network Variables provide Access across the Network

With the new standard a direct representation of input and output ‘Network Variables’ (NWVs) in the Device Object Dictionary is supported. Network Variables represent remote I/O objects of other CANopen devices containing the actual data of those objects. The time critical transfer of NWVs is based on PDOs. For the representation of NWVs a separate object range is reserved within the Device Object Dictionary according to Fig. 3-1.

Index	Function / Description
0001	**Type Definitions**
...	
009F	
1000	**Communication Parameters**
...	
1FFF	
2000	**Vendor Specific Parameters**
...	
5FFF	
6000	**Standard Device Parameters**
...	
9FFF	
A000	**Input Network Variables**
...	
A47F	
A480	**Output Network Variables**
...	
A8FF	

Fig. 3-1 : Object Dictionary of a IEC 1131-3 programmable CANopen Device

Thereby an 'Input-NWV' represents an input to the CANopen Object Dictionary as seen from the local application of the CANopen device and such will be transmitted via a Transmit-PDO. An 'Output-NWV' represents an output from the Object Dictionary with respect to the local application.

Since the assignment of Object Dictionary entries to real remote I/O devices depends on the specific PLC application normally it has to be configured by means of a CANopen configuration tool. After configuration there is no difference between a standard CANopen I/O device e.g. according to DS 401 and an IEC 1131-3 programmable CANopen PLC.

%IB 0000	%IB 0001	%IB 0002	%IB 0003	%IB 0004	%IB 0005	%IB 0006	%IB 0007	....	%IB 8188	%IB 8189	%IB 8190	%IB 8191
%IW 0000		%IW 0001		%IW 0002		%IW 0003		....	%IW 4094		%IW 4095	
%ID 0000				%ID 0001				....	%ID 2047			

Fig. 3-2 : Mapping of the Process Image in the Object Dictionary

The number of available NWVs is dependent on the size of the internal process image of the PLC. In the standard a large number of different data types for NWV are defined and for each supported data type a range of 64 object entries within the Object Dictionary is reserved (Fig. 3-3).

NWV's Input	NWV's Output	Data Type
A000	A480	Integer 8
A040	A4C0	Unsigned 8
A080	A500	Boolean
A0C0	A540	Integer 16
A100	A580	Unsigned 16
A140	A5C0	Integer 24
A180	A600	Unsigned 24
A1C0	A640	Integer 32
A200	A680	Unsigned 32
A240	A6C0	Float (Single Format)
A280	A700	Unsigned 40
A2C0	A740	Integer 40
A300	A780	Unsigned 48
A340	A7C0	Integer 48
A380	A800	Unsigned 56
A3C0	A840	Integer 56
A400	A880	Unsigned 64
A440	A8C0	Integer 64

Fig. 3-3 : Mapping of Network Variables of different Data Types in the Object Dictionary

With a maximum number of 254 subobjects per index, a maximum of 64 * 254 = 16256 objects per supported data type is possible. For mapping of network variables into the Object Dictionary of a device, the DSP 405 uses an overlapped memory assignment according to .

With the overlapped mapping of NWVs of different data type, the maximum number of objects of the largest data type is determined by the maximum number of the smallest data type. For example, if the smallest data type is Boolean, a maximum size of 16 265 /8 = 2030 Bytes for the process image is available. By supporting a data type of Unsigned 8, the maximum size of the process image is 16256 Bytes.

3.2 Provision of Standard CANopen-specific Function Blocks

In addition to NWVs DS-405 specifies some basic functions for CANopen-specific network control and monitoring:

- SDO read/write access to the Object Dictionary of a remote node. For the SDO transfer, two different protocols are available. If less than 5 byte of data have to be transferred, the 'expedited multiplexed domain protocol'[4] should be used. Therefore the function blocks CIA405_ SDO_WRITE4 respectively CIA405_SDO_ READ4 are specified. For transferring of larger data blocks, the function blocks for segmented multiplexed domain transfer are available. Depending on the largest supported object length, the PLC-vendor has to implement function blocks for the segmented SDO transfer of 7, 14, 21 etc. bytes of data.
- The selective, respectively global reception of emergency messages is supported by the function blocks CIA405_RECV_ EMY_DEV and CIA405_RECV_EMY. With the later the reception of emergency messages from any device is possible.
- For controlling of the communication state of a device the function blocks CIA405_GET_STATE respectively CIA405_ SET_STATE are specified.
- A PLC application may ask for its own device node ID by means of the function CIA405_GET_LOCAL_NODE_ID or read the current specific error state of its CANopen kernel by means of the block CIA_GET_CANOPEN_KERNEL_STATE.

3.3 Definition of Supporting Functions

Since, in terms of IEC 1131-3, a PLC configuration may consists of several resources, programs and tasks, it is necessary to describe the configuration as well as further functions.

Therefore DSP-405 specifies additional Object Dictionary entries, for accessing and controlling of further functions and data of the PLC. Such functions are for example the starting and stopping of tasks or changing of tasks cycle time and priority. Additionally data has to be provided concerning the version number, name and status of a configuration, resource or task. This type of functionality respectively information is located in the index range between 9500 hex and 9800 hex of the Object Dictionary.

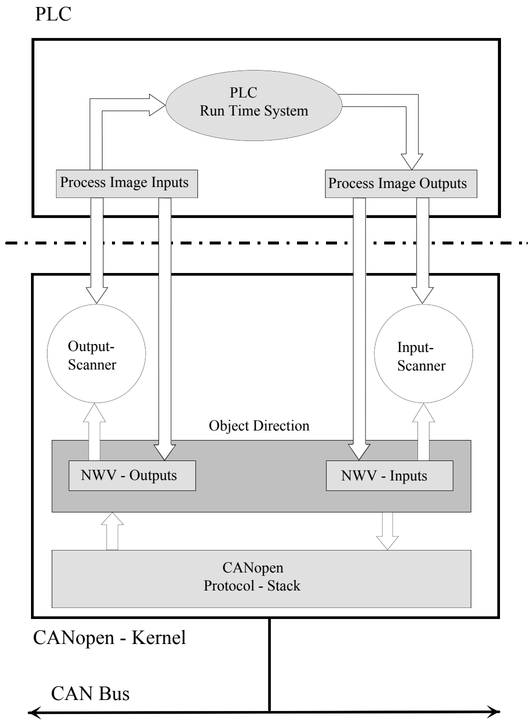

Fig. 4-1 : Basic Structure of a CANopen PLC

4 Basic Structure of a CANopen PLC

In Fig. 4-1 the basic structure of a CANopen PLC is shown as far as the processing of process data is concerned. Thereby the PLC is regarded as a kind of local I/O device of the CANopen kernel. Since the inputs of the PLC are no more available locally, they have to be provided by NWV outputs or standard profile outputs of the CANopen kernel. In the same manner outputs of the PLC have to be provided to the CANopen kernel in form of NWV inputs or standard profile inputs.

Main task of the CANopen kernel therefore is the initiation of a corresponding PDO transmission when a input NWV has changed, respectively the updating of the PLC process inputs when the received PDOs respectively NWV output has changed.

In Fig. 4-2 the basic structure of the interface between a PLC run-time-system/application and the CANopen kernel is shown as implemented by IXXAT Automation GmbH.

The following functions should be requestable from the PLC application/run time system via a command interface channel:

- Initiatiation of client SDO requests for reading or writing of Object Dictionary entries at other CANopen devices, connected to the network
- Setup, starting and stopping of the NMT master, resetting of the network
- Setup, starting and stopping of the network synchronization and system time processes
- Initiation of the transmission of multiplexed PDOs
- Initiation of the transmission of emergency messages
- Setup of the Configuration and SDO Manager.

On the other hand, the CANopen kernel should have the possibility to provide status information and events to the PLC application/run-time-system via an appropriate status channel. This type of information includes the following functions, respectively messages:

- Signaling of local errors
- Signaling of network related events, e.g. guarding errors
- Passing of received emergency messages
- Passing of OS command, debug and prompt requests
- Indication of a reset of the CANopen kernel via the CAN bus.
- Passing of remote procedure calls

The main elements of the interface shown in Fig. 4-2 are:

- **Command-Buffer**

Serves for the confirmed transfer of command data between runtime system and CANopen kernel. Only one message can be posted at a time.

- **Tx-Message-Queue**

For transferring of one or more event-triggerd messages from the PLC to the CANopen kernel.

- **Rx-Message-Queue**

Provides a message channel from the CANopen kernel to the PLC run-time-system, respectively application.

- **Status-Buffer**

Used for passing of the actual status of the NMT master, CANopen kernel, guard timing or other time out values.

- **NWV-Input-Image**

Represents a memory area for storage of the actual kernel input data.

- **NWV-Output-Image**

Represents a memory area for storage of the actual kernel output data .

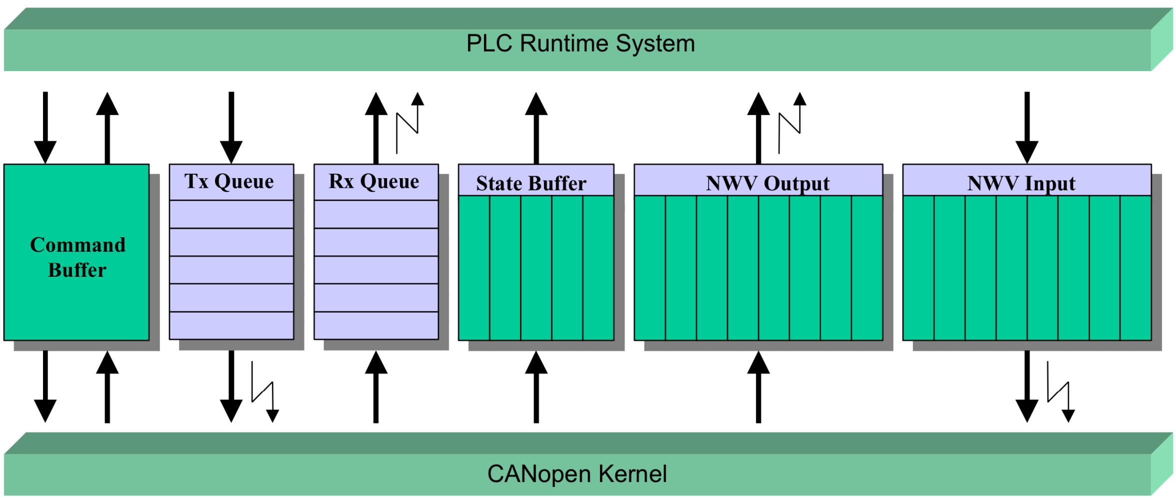

Fig. 4-2 : Structure for Interfacing a PLC Application/Run-Time-System with a CANopen Kernel (Source: IXXAT Automation GmbH)

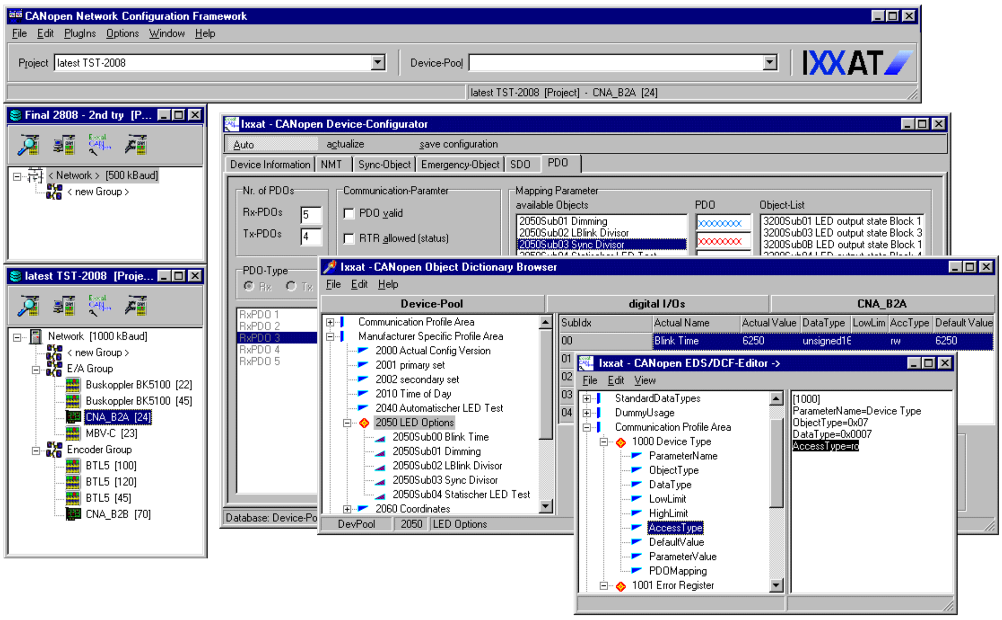

Fig. 5-1 : CANopen System Configuration Tool (IXXAT Automation GmbH)

Of course, for transferring of messages across the specified interface, an appropriate message protocol has to be used which allows the confirmed and unconfirmed transfer of messages across the interface.

5 Support of System Configuration and Programming

With the introduction of standardized programmable CANopen devices which may communicate without any restriction with any other intelligent or non-intelligent device in a network, any type of distributed processing is possible.

On the other side, the completely free configurability and programmabilty of such systems requires the availability of sophisticated tools for system configuration, programming and testing.

According to the open philosophy of CANopen, DSP 405 specifies an open, vendor-independent interface for the integration of a system configuration and a IEC 1131-3 programming tool (Fig. 5-2).

According to this specification, the CANopen Configuration Tool provides the information about the configured NWVs to the Programming Tool by means of a standard CANopen Device Configuration File (DCF). This includes any of the information which is necessary for the Programming Tool. Since it should also be possible that NWVs first are defined or later changed by the Programming

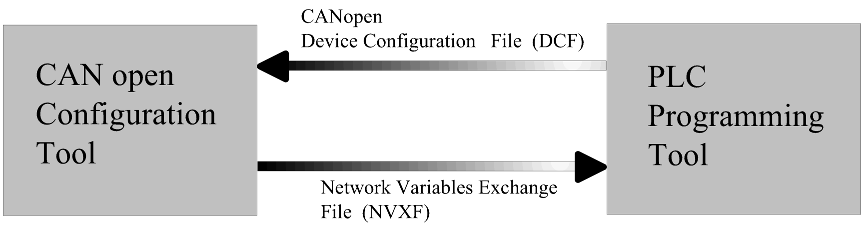

Figure 5-2: Standarized Interface between a CANopen System Configuration and a IEC 1131-3 Programming Tool according to DSP 405

Tool, the related information has to be provided to the Configuration Tool by means of a so-called Network-Variable-Exchange-File (NVX-File). This file contains any information about a newly specified NWV, like name, data type or direction and has to be processed by the Configuration Tool appropriately.

In Fig. 5-1 a screenshot of some device-oriented functions of the IXXAT CANopen System Tool is shown. The IXXAT CANopen System Tool therefore assists the system implementor from starting with system installation, system and device configuration and programming to system testing, debugging and documentation.

The IXXAT CANopen System Tool also supports the automatic mapping and linking of PDOs, based on application-specific communication links between I/O-channels or NWVs. In this case, the communication process is almost hidden to the system implementor. In the final step of a system configuration process the configuration data is downloaded onto the devices by the System Tool, using an SDO channel to each device.

6 Summary

With the availability of considerably extended communication functionalities, specified in CiA DSP 302 (dynamic establishment of SDO channels between devices by means of a SDO Manager facility, Configuration Manager facility, multiplexed PDO communication) and the specification of a standard profile for intelligent, IEC-1131-3 programmable CANopen devices, specified in CiA DSP 405, CANopen now provides any means for the realization of distributed, freely configurable automation systems of high performance. Furthermore, highly sophisticated integrated System Tools support the implementor of CANopen-based systems during system programming, configuration and testing.

IXXAT Automation GmbH
Doggenriedstraße 40
D-88250 Weingarten
Phone +49-(0)751-56146-0
Fax +49-(0)751-56146-29
Homepage www.ixxat.de
eMail info@ixxat.de

[1] CiA DS-301, Version 3.0, October 1996
CAL based Communication Profile for Industrial Systems
CiA DSP-401, Version 1.4, December 1996
CANopen Device Profile for I/O Modules
CiA DSP-402, Version 1.0, August 1196
CANopen Device Profile for Motion Control and Drives
CiA DSP-406, Version 1.91, February 1998
CANopen Device Profile for Encoders

[2] CiA DSP-302, Version 1.0, March 1998
Framework für Programmable CANopen Devices

[3] CiA DSP-405, Version 1.0, March 1998
CANopen Interface Profile for IEC1131 environments

[4] CiA DS-202-2, Version 1,1, February 1997
CAN Application Layer, CMS Protocol Specification