iCC 1994

1st international CAN Conference

in Mainz (Germany)

Sponsored by

Allen Bradley
National Semiconductor
Philips Semiconductors

Organized by

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

Dipl.-Ing. Martin Rostan, Prof. Dr.-Ing. Gerhard Gruhler
STA Reutlingen

CAN Real Time Communication Profile

Abstract

A communication profile for real time systems which is based on the layer 7 specification CAN Application Layer (CAL) was developed in the ESPRIT project „ASPIC“. It supports the quick exchange of real time data as well as the utilisation of conventional device profiles.
The concept involves the introduction of two communication channels with differing features: an operational channel for real time data and a service channel for parameter communication. Both are established between the central controller and the connected devices. The concept uses a subset of CAL. NMT and DBT are employed to perform network management and identifier distribution whilst two CMS services are sufficient to accomplish the communication channels.
Device profiles interface to the communication profile. Here the communication objects are defined which enable standardised access to functions and features of the devices. The objects (e.g. parameter, modes or programs) can be addressed via an index in the service channel.

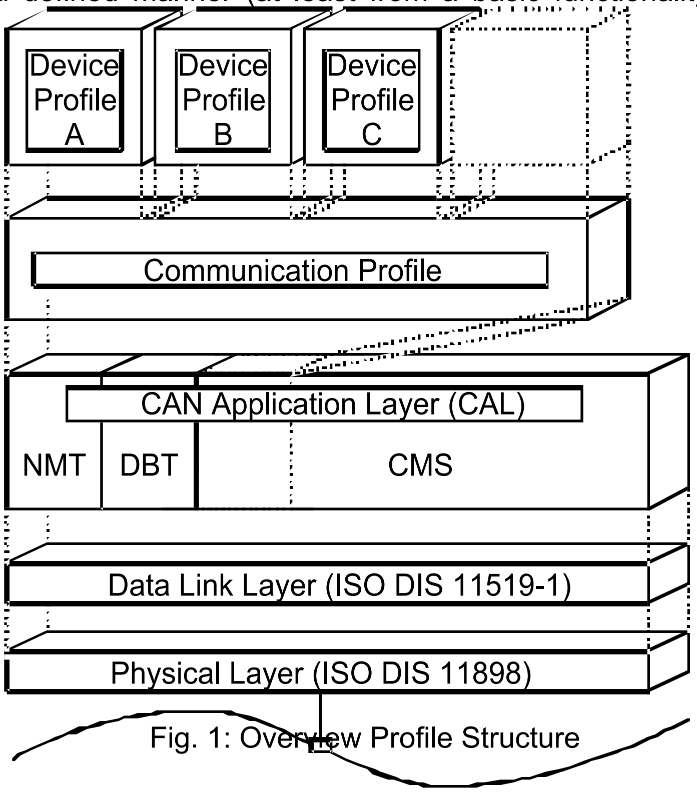
Introduction

The CAN world is still dominated by manufacturer specific solutions, especially in the higher protocol layers. This approach is successful if the market consists, like in the automotive industry, of a huge number of identical or similar networks, or if the CAN network is regarded as an internal communication link which remains a „black box“ for the customer. However, these isolated, incompatible solutions do not satisfy the majority of fieldbus users, who demand an open network concept which ensures that devices of different vendors can communicate without an interface adaptation. With open systems the engineering efforts in application design and device integration can be widely reduced. In order to compete successfully in a tight market, CAN has to migrate towards a more open system.
Open systems require standardised ISO/OSI layers as well as standardised profiles. Due to its origin, initially only the CAN data link layer was standardised (ISO 11519-1). Later international standards for the physical layer were established. In the automation area generally ISO 11898 (CAN „high speed“) is applied in conjunction with the CiA recommendations [1]. For the application layer CiA has developed the CAL (CAN Application Layer) standard [2]. Several profile proposals are currently being discussed in the CAN community.

The „profile-layer“ generally utilises application layer mechanisms that are based on MMS. There are profiles with varying states of development e.g. for Profibus, Interbus-S and CAN. This paper introduces a communication profile that provides the means to use device profiles for real time applications together with CAN/CAL. This communication profile has been developed in the ESPRIT-project „ASPIC[1]“. The aim of „ASPIC“ is to achieve a new generation of control architecture and components based on fieldbus technology for flexible and modular production units in order to gain installation flexibility and advanced maintenance management [3]. Following an extensive investigation [4,5] CAN has been chosen for this task.

Profiles: Overview

Profiles can be classified into two separate parts (Fig. 1). First the ***device profile*** is described. This part contains the application and device specific definitions of the meaning of the transmitted data, it defines the device parameters. In doing so it describes the functionality of the device as it can be accessed via the bus. It is common to arrange the device parameters in function groups. The parameters are listed in an object dictionary. This object dictionary contains description, structure and data type as well as an address (index/sub-index) of the parameter. There is a range of mandatory entries in this dictionary which ensure that all devices of a particular type behave in a defined manner (at least from a basic functionality viewpoint). The object dictionary concept caters for optional device features which means a manufacturer does not have to provide certain extended functionality on his device but if he wishes to do so he must do it in a pre-defined fashion. Additionally, there is sufficient address space for truly manufacturer specific functionality. The overall functionality of the device is further described by state machines.


Device Profile A
Device Profile B
Device Profile C
Communication Profile
CAN Application Layer (CAL)
NMT
DBT
CMS
Data Link Layer (ISO DIS 11519-1)
Physical Layer (ISO DIS 11898)
Fig. 1: Overview Profile Structure

The second profile part is the ***communication profile***. It contains the description of the interface to the underlying communication system, in the present case to the application layer of the fieldbus. According to the bus system and the application area it makes sense to adapt the features of the application layer to the present requirements by using a subset of the offered services. E.g. Profibus with its comprehensive functionality of the application layer abandons the multi-master functionality for some profiles in order to improve the performance.

In the „ASPIC“ project first the communication profile [6] and a „design guide“[7] for device profiles were developed. This guide contains the „common denominator“ for the device profiles: the object dictionary structure and the encoding rules. Based on these definitions a number of device profiles (i.e.: see Fig. 2) have been developed for typical components of automated production units like servo drives [8], low level sensors [9] (digital and analogue I/O, encoder, serial interfaces, identification systems and simple user interfaces), vision systems [10] and intelligent gateways [11].

The Profibus device profiles (i.e.:[12]) are organised in frame sheet and various function and device data sheets. The frame sheet is equivalent to the communication profile (i.e.[13]). The draft of a sensor/actuator profile [14] exists for Interbus-S, its objective is similar to the design guide mentioned above. The profiles of Drivecom [15,16], Encom [17] and the other server device profiles are based on this sensor/actuator profile.

Index	Object	Name	Type	M/O
6400H	Array	8 Input Lines	Unsigned8	M
6401H	Array	8 Output Lines	Unsigned 8	M
6402H	Array	Single Input Line	Boolean	M
6403H	Array	Single Output Line	Boolean	M
6404H		Reserved for future use		
:	:	:	:	:
64FFH		Reserved for future use		
6500H	Array	Input Polarity	Unsigned8	O
6501H	Var	Output Polarity	Unsigned8	O
6502H	Array	Interrupt Mask	Unsigned8	O
6503H	Array	Output Configuration	Unsigned16	O
6504H	Array	Input Configuration	Unsigned16	O
6505H		Reserved for future use		
:	:	:	:	:
6BFFH		Reserved for future use		

Object 6500H

Defines the polarity of a group of 8 input lines.

INDEX	6500H
Variable Name	Input-Polarity
Object Code	8H
Number of Elements	1-256
Data Type Index	8H
Length	8

Value Description

Sub-Index	0H (Total number of input line groups of 8)
Value Range	Unsigned8
Manadory Range	No

Sub-Index	1H (Input Port Lines 0-7)
Value Range	Unsigned8
Manadory Range	No

to

Sub-Index	FFFH (Input Port Lines 2032-2039)
Value Range	Unsigned8
Manadory Range	No

Object Function

Object Class	Optional
Operational Channel Mapping	Possible
Error Codes	Yes

Fig. 2: Device-Profile for digital I/O Modules (Extract)

CAN Communication Profile

The following goals were in the focus of the communication profile development for CAN real time applications:

- simple and clear structure in order to facilitate implementation and maintenance
- use of existing higher layer protocol standards, especially for layer 7 of the ISO/OSI model
- use of FullCAN controller Intel 82527 for simple devices
- minimum number of communication object identifiers
- use of low-cost microcontrollers for I/O modules
- deterministic bus behaviour
- far reaching compatibility with other standards, e.g. Drivecom profiles, Sercos-Interface

These goals led to the introduction of two communication channels with different features (see Fig. 3)

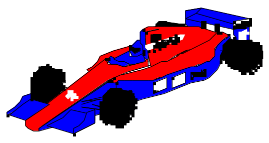
	
Operational Channel	**Service Channel**
Used for Real-Time Data	**Used for Non-Real-Time Data**
Synchronous Messages and Interrupt-driven Messages	**Asynchronous Messages**
High Priority Identifiers	**Lower Priority Identifiers**
Optimised for High Speed Data Exchange	**Lower Speed**
Confirmed Services	**Confirmed Services**
8 Bytes per Message	**Multi-Telegram Messages Possible**
Format must be negotiated between Communication Partners	**Uses Indexing to Reference Data Fields in Object Dictionary**

Fig. 3: Channel Concept

Service Channel and Operational Channel

The so called ***service channel*** is the communication channel for the transmission of data without real time requirements like device parameters or programs. For such asynchronous data a reduced transmission speed is acceptable but requires sophisticated error handling (with confirmed services). The type of data transferred may range from a simple Boolean to a large file, therefore domain handling capability is necessary. As the data content may differ from telegram to telegram a description of the data meaning (or a data identification set) has to be transmitted as well. The quickest way to do this is by including the object dictionary address (index/sub-index) in the message.
The ***operational channel*** is the data channel for high speed exchange of real time data. It can be run in two different modes: i.e. drives transmit command and actual values cyclically in the synchronous mode whilst an I/O module can react in the event driven mode after an input signal has changed. This avoids data being constantly and unnecessarily placed on the bus contributing to the bus traffic. The format and content of the operational channel data is negotiated via the service channel, generally at boot-up of the network. Therefore there is no need to transmit index or sub-index information, the operational channel contains only process data, which can „by-pass“ the device profile software at runtime.
For each device a service channel and an operational channel is established to the master control unit (see Fig. 4). The communication profile does not exclude additional channels between the modules, i.e. between I/O modules and decentralised PLCs for I/O data preprocessing.

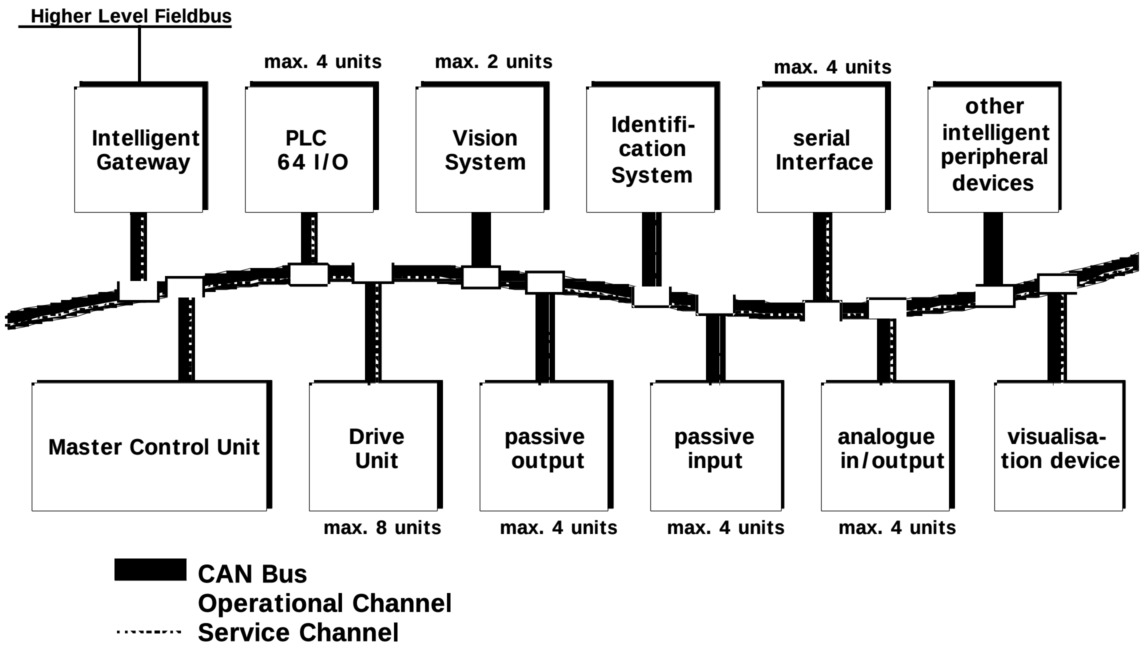


Fig.4: Devices and Communication Channels in a Production Unit

CAN Application Layer (CAL)

The search for suitable protocol standards for the higher layers of the ISO/OSI model lead to CAL. The CAL-specification describes the service elements that are required for distributed controls. Network-management services like lifeguarding or node initialisation are included in the NMT service group. The DBT services provide for dynamic identifier distribution in 8 priority classes and consistency checks at boot-up of the network. Both service groups are used in the communication profile presented here. Partial implementation as specified in CAL is permitted. I.e. small network configurations can be clear enough to abandon the dynamic distribution of the CAN identifiers.
The CAN message specification (CMS) inside CAL supplies all the elements for the actual

are defined and it is distinguished between the access types read_only, write_only and read_write. Additionally there are local and remote services and a variety of data types. The degrees of freedom amount to a fullness of services and consequently to a large number of functions in the application layer.
By using the channel concept only two CMS services cater for all the features required by the CAN real time communication profile (see fig. 5). The application layer then becomes lean and comprehensible, and the CAN specific advantages remain fully intact.

The operational channel does not require confirmation of each message, the emphasis here lies on the performance. Therefore for this channel the write_only-service is utilised, which basically describes the simplest and quickest CAN way to transfer data of up to 8 bytes.

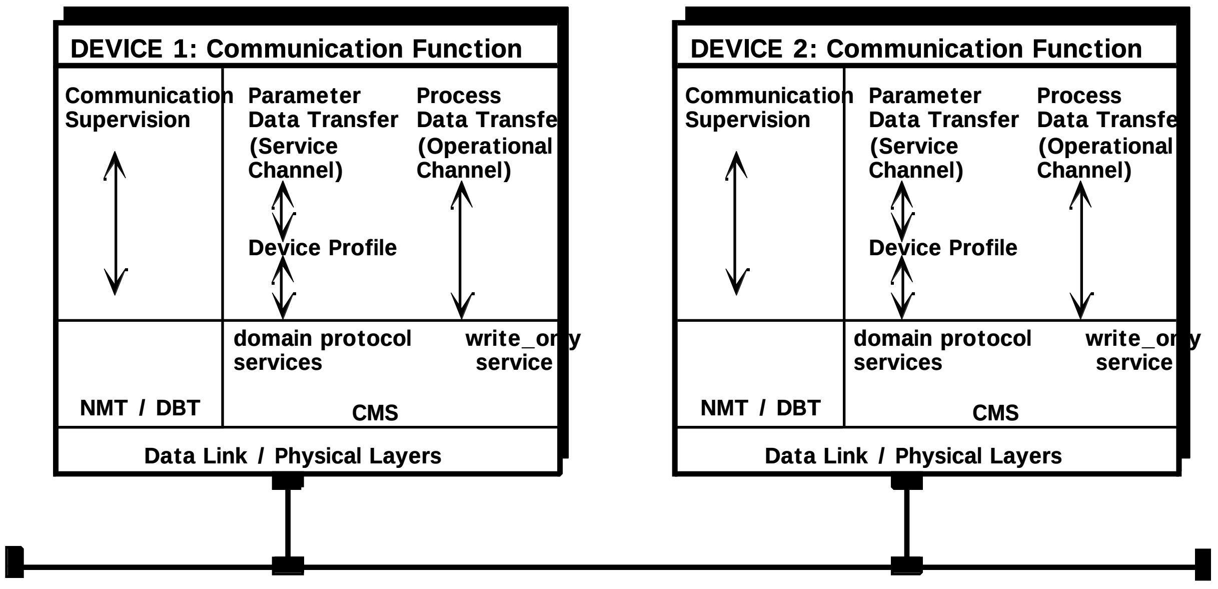

DEVICE 1: Communication Function
Communication Supervision
Parameter Data Transfer (Service Channel)
Process Data Transfer (Operational Channel)
Device Profile
domain protocol services
write_only service
NMT / DBT
CMS
Data Link / Physical Layers
DEVICE 2: Communication Function
Communication Supervision
Parameter Data Transfer (Service Channel)
Process Data Transfer (Operational Channel)
Device Profile
domain protocol services
write_only service
NMT / DBT
CMS
Data Link / Physical Layers

Fig. 5: Communication Channels and CAL

For the service channel the CMS multiplexed domain service is used (see fig. 6) This combination provides:

- Expedited transfer of data of size less than or equal to 4 bytes. This transfer mechanism needs the transfer of 2 CAN messages only (Initiate Request, Initiate Response).
- Segmented transfer of data whose size is greater than 4 bytes. All data objects with more than 4 bytes in size are transferred as a sequence of Segment Commands preceded by an Initiate Command. This mechanism needs the exchange of at least 4 CAN messages.
- Transfer of a Data Identification Set (Index and Sub-Index as used in device profiles) with the data.
- Feedback of Error information with the Server Reply message with optional data set identification.
- Abort of Data transfer by either Client or Server, with error feedback and optional data set identification.

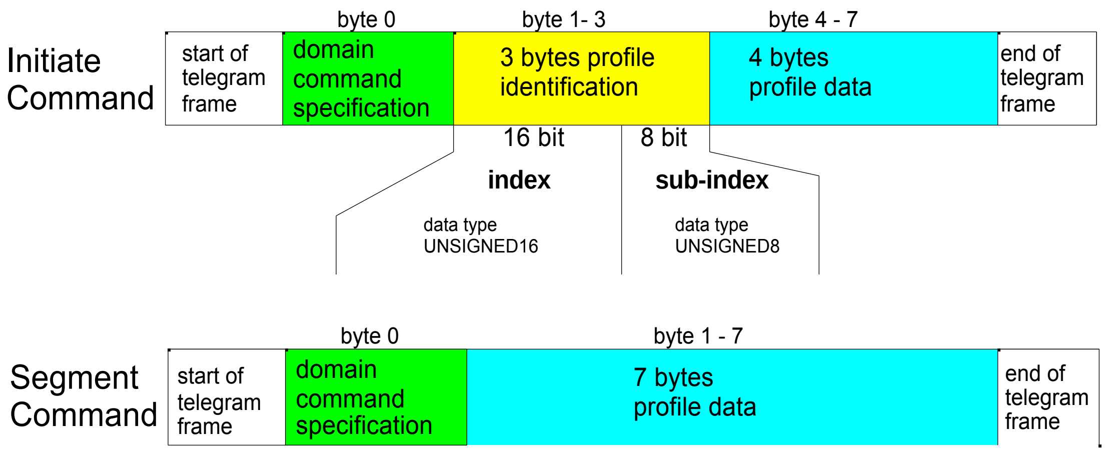


Fig. 6: Service Channel Telegram Structure (CAL Expedited Domain Protocol)

Identifier-Allocation

The key to a working real time behaviour of a CAN network is the clever allocation of the message identifiers. The CAL priority level model describes 8 priority levels each with 220 priorities (or identifiers) and comprises the identifiers 1 to 1760. The remaining identifiers (0, 1761 - 2031) are reserved for network control by NMT and DBT. For transmitting emergency messages each unit in the production cell must be able to use messages with the highest priority. For these exceptional messages a number of identifiers in priority level 0 have been reserved. The real time clock for the network (SYNCH Message, see below) has the highest priority (level 1) of all messages that occur in normal operation. For each communication channel one priority level on the identifier scale has been allocated. (see Fig. 7) To guarantee the cyclic behaviour of the production unit data exchange a higher priority (level 2) is allo-

CAL priority level	message / channel
0 (Id 1 - 220)	Emergency (fault)
1 (Id 221 - 440)	Cell timing (synch, timestamp)
2 (Id 441 - 660)	Operational Channel (cyclic)
3 (Id 661 - 880)	Operational Channel (event)
4 (Id 881 - 1100)	***reserved***
5 (Id 1101 - 1320)	Service Channel
6 (Id 1321 - 1540)	***reserved***
7 (Id 1541 - 1760)	***reserved***
> ID 1760	NMT/DBT services - lifeguarding (preallocated)

Identifier	Client	Server	message
1111	PUC	DU 1	PU_DU_S / parameter
1112	PUC	DU 2	PU_DU_S / parameter
1113	PUC	DU 3	PU_DU_S / parameter
1114	PUC	DU 4	PU_DU_S / parameter
1115	PUC	DU 5	PU_DU_S / parameter
1116	PUC	DU 6	PU_DU_S / parameter
1117	PUC	DU 7	PU_DU_S / parameter
1118	PUC	DU 8	PU_DU_S / parameter
:			
1141	PUC	DIO 1	PU_LL_S / parameter
1142	PUC	DIO 2	PU_LL_S / parameter
1143	PUC	DIO 3	PU_LL_S / parameter
1144	PUC	DIO 4	PU_LL_S / parameter
:			
1161	PUC	AIO 1	PU_LL_S / parameter
1162	PUC	AIO 2	PU_LL_S / parameter
1163	PUC	AIO 3	PU_LL_S / parameter
1164	PUC	AIO 4	PU_LL_S / parameter
:			
:			

Fig. 7: Identifier Allocation

cated to the cyclic messages than to the event driven messages (level 3). The service channel gets the lowest priority as is carries no real time data.
The real time behaviour of the operational channels is guaranteed if the transmission of all possible operational messages requires significantly less time that the cycle time. Worst case calculations for the ASPIC production cell showed that the real time messages take about 51% of the cycle time. The rest is available for service channel communication.

Bus Synchronisation

Besides the cyclic exchange of data many real time applications demand synchronisation between different bus nodes. I.e. axis of a kinematic have to be synchronised or I/O modules have to set outputs or read inputs simultaneously like a PLC. Synchronised drives expect commanded positions and send actual positions in pre-defined time windows. The CAN real time communication profile meets these requirements by introducing synchronisation telegrams with a high priority, which divide the time axis in equidistant communication cycles (see fig. 8). The synch-messages do not contain data and can be used as an interrupt by I/O modules to then set outputs or read inputs. Intelligent devices like drives can synchronise e.g. using the PLL method. In the report window right after the synchronisation telegram the drives send their actuals and the I/O modules send their input values. Afterwards, in the command window, the commands and the output values are transmitted, which are then set valid at the next synch-signal. As the report window directly follows on the synch-signal it can be hit even by simple components without a timer. Bandwidth not used inside the windows and the time between the command window and the synch telegram is available for low-priority service channel messages.

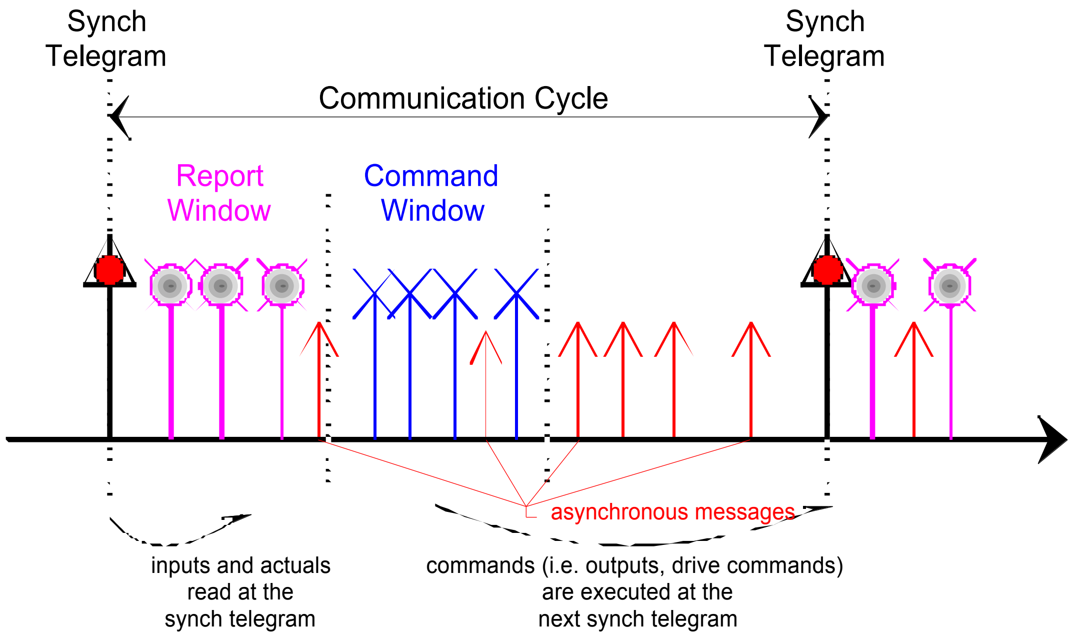

Synch Telegram
Synch Telegram
Communication Cycle
Report Window
Command Window
asynchronous messages
inputs and actuals read at the synch telegram
commands (i.e. outputs, drive commands) are executed at the next synch telegram

Fig. 8: Bus Synchronisation and Sampling/Actuation

Test Method and Certification

The advantages of open systems are only achieved if the protocol implementations are in exact agreement with the specification. Therefore in ASPIC suitable protocol test methods are being developed which are available for interface certification as well for companies not directly participating in the project. Both in conformity testing (ISO/OSI-layers) as in interoperability testing (profiles) the device under test is stimulated with extensive test message sequences. These are combined manually or in automatic mode randomly out of a large number of small test strings in order to achieve1
maximal variance of test states. The test evaluation is performed automatically as well. In doing so long time tests are possible with changing characteristics.

Literature

[1] CiA Standard „CAN Physical Layer for Industrial Applications“, CiA-DS 102-1, 1994.

[2] CiA Standard „CAN Application Layer“, CiA/DS 201-207, 1994.

[3] Gruhler, G.; Jantzer, M.: Anforderungen an einen Feldbus für Montagezellen. Vortragsband iNet ´93, S. 49...58, München: Drebinger 1993

[4] Gruhler, G.; Rostan, M.: Auswahl eines Feldbussystems für Produktionszellen. Elektronik plus 6/93, S. 77...82, München 1993.

[5] Rostan, M.; Wanner, J.: Einführende Darstellung und detaillierter Vergleich von Feldbussystemen. 1993. Available from: STA Reutlingen, phone +49 7121 271-327.

[6] ASPIC Consortium: Specification of CAN Real Time Communication Profile. 1994. Available from STA Reutlingen.

[7] Ratcliff, K.; Booth, R.; Farsi, M.: Principles of ASPIC Device Communication. University of Newcastle upon Tyne, United Kingdom 1994.

[8] Drive Profile, Moog Ltd, Ringaskiddy, Co. Cork, Ireland 1994.

[9] Low Level Sensor Profiles, University of Newcastle upon Tyne, U.K. 1994.

[10] Vision System Profile, JL Automation, Sunderland, U.K. 1994.

[11] Intelligent Gateway Profile, ADL Automation, Malissard, Frankreich 1993.

[12] Profibus Profile Drehzahlveränderbare Antriebe. VDI/VDE-GMA FA 6.8, VDI-Verlag Düsseldorf 1992.

[13] Profibus Profil für Programmable Controller. Profibus-Nutzerorganisation e.V., Wesseling 1993

[14] Sensor/Aktor-Profil (12). Interbus-S Club e.V., Baden-Baden 1993.

[15] Drivecom Profil Antriebstechnik (21). Drivecom Nutzergruppe e.V., Blomberg. 1993.

[16] Drivecom Profil Servo (22). Drivecom Nutzergruppe e.V., Blomberg. 1994.

[17] ENCOM Profil Encoder (71). Encom Nutzergruppe e.V., Blomberg. 1993.