CAN Bus as a Spacecraft Onboard Bus

Chris Plummer – Cotectic Ltd
Dolf Ledellaan 7, B-3090 Overijse, Belgium, +32 2 6753499, c.plummer@skynet.be

Peter Roos – ESTEC
Keplerlaan 1, 2200AG Noordwijk, The Netherlands, +31 71 565 3692, Peter.Roos@esa.int

Luca Stagnaro – ESTEC
Keplerlaan 1, 2200AG Noordwijk, The Netherlands, +31 71 565 4667, Luca.Stagnaro@esa.int

1. Introduction

Recently there has been a growing interest in the use of CAN bus as a spacecraft onboard bus in place of more traditional buses such as ESA OBDH and MIL-STD-1553B. While CAN has similar performances to both the OBDH bus and to MIL-STD-1553B, it is an attractive option for several reasons, including:

- Very low cost;
- High reliability;
- Readily available commercial hardware for terrestrial applications;
- Readily available commercial components and VHDL cores;
- Very widely deployed in terrestrial applications;
- Underlying physical layers with high reliability and high availability characteristics;
- Many well established and emerging higher layer protocols for use over CAN bus;

However, the CAN bus is significantly different in several respects from the buses that have traditionally been used onboard spacecraft. Most notably, CAN does not operate in a master/slave mode. Instead, control of the bus, and in particular bus medium access, is distributed amongst all nodes connected to the bus. Two other important differences are that the CAN bus is inherently asynchronous in nature, and uses frame labelling rather than node addressing.

The goal of this paper is to highlight the work that is currently being carried out by the ESTEC CAN Bus Working Group, which is trying to ease the implementation of the CAN bus by establishing recommendations and guidelines for its use in spacecraft applications. The paper also provides an introduction to the CAN bus and its associated physical layer and higher layer protocols.

2. The Role of the ESTEC CAN Bus Working Group

The ESTEC CAN bus working group has been established to define recommendations and guidelines for the use of CAN bus in spacecraft. The goal is to avoid the confusion and large diversity of implementations that has plagued other buses used in spacecraft. For example, although MIL-STD-1553B has been used in many spacecraft, there is no commonly agreed protocol used over this bus. New, unique protocols have been developed for each mission, and consequently there is no possibility of interoperability, re-use potential of equipment is compromised, and specific ground check-out equipment must be developed for every mission.

The recommendations and guidelines defined by the CAN bus working group are expected to result in interoperable implementations of flight equipment and commonality of test and check-out equipment. This in turn should reduce the development costs of CAN bus connected equipment, reduce technical risks, and encourage the development of standard CAN components for spacecraft.

The CAN bus working group is addressing all aspects of the use of CAN bus in spacecraft, including the electrical specification, higher level protocols for use over CAN bus, techniques for implementing redundancy, transfer of large data units, and time distribution. Another important role being undertaken by the working group is the establishment of a CAN bus test bed that will be used to gain real implementation experience and to validate the working group recommendations. In the future, this test bed may become a CAN bus reference system that can be used in the early phases of modelling and simulation of a mission proposing to use CAN bus.

Because the working group is defining recommendations at the very early stages of the adoption of CAN bus, i.e. before there are a significant number of projects planning to use CAN bus, we have high hopes that it will be able to help projects make an informed choice of bus, and avoid the dissonance that often results from an uninformed selection.

3. Characteristics of CAN Bus

The CAN bus specification [2] defines frame formats and data exchange protocols to support distributed real-time control over a serial communication bus. With reference to the OSI 7-layer reference model defined in [1], the CAN specification defines the Medium Access Control (MAC) sub-layer and part of the Logical Link Control (LLC) sub-layer of the Data Link layer.

The physical layer is undefined, and the implementer consequently has a freedom of choice here. A number of complementary standards have been defined (see next section), which are suitable for use in spacecraft.

The CAN bus specification contains a number of interesting features including:

Priority based bandwidth allocation, which ensures that high priority frames are treated with minimum latency;

Multi-master capability, whereby any node on the bus can transmit at any time that the bus is free. Although this is ideally suited for asynchronous messaging between nodes, an increasingly common requirement with modern message based software architectures, it makes it more difficult to implement the synchronous data acquisition systems traditionally used in spacecraft. However, there are a number of techniques for synchronous data transfer provided by higher layer protocols designed for use with CAN;

Lossless bus bandwidth arbitration, i.e. if two nodes try to transmit on the bus simultaneously a bus collision will be detected but one of the nodes will be able to transmit. This makes maximum use of the bus bandwidth and obviates the need for

complicated bus contention resolution techniques often required with Collision Detect/Multiple Access buses;

Fault confinement, where the protocol engine running in each node can distinguish short term disturbances from permanent bus failures. A node that becomes faulty can disconnect itself from the bus;

Automatic retransmission. CAN frames that are corrupted in transmission are automatically retransmitted. This leads to a very reliable transfer capability, but results in a reduction in determinism of the bus.

The basic unit of data transfer on the bus is the CAN frame. Two different frame formats are defined by CAN 2.0B, one that uses an 11-bit frame identifier, and one that uses a 29-bit frame identifier. In both cases, the maximum amount of data that can be transferred in a single frame is 8-octets. The two formats can be mixed on a single CAN bus.

Four different frame types are defined, namely data frames, remote frames, error frames, and overload frames. The last two types are used for bus control and are not involved in data transfers. Data frames are used by a node to transmit data on the bus. Remote frames are used to request the transmission of a data frame. We anticipate that, for most spacecraft data transfer applications, only data frames will be used. The format of the standard (11-bit identifier) extended (29-bit identifier) data frame is illustrated in Figure 1[1].

CAN bus uses frame content labelling rather than the more usual node addressing used by other buses that have been used onboard spacecraft. The identifier labels the data contained within the frame, and any node on the bus can be configured to receive frames with specific labels. This makes it very easy to implement point to multi-point communications and frame broadcasting, and means that data generating nodes do not need to know the addresses of other nodes on the bus, but it also means that additional specifications must be generated by a project allocating identifiers to frames, and a pseudo-addressing scheme may need to be defined. However, most high level protocols for use with CAN bus provide for this.

[1] This diagram courtesy of G. Leen of the University of Limerick, Ireland. It has been previously published in *"Time-triggered Controller Area Network",* Leen, G. and Heffernan, D., **IEE Computing and Control Engineering Journal.** Vol. 12, Issue 6, Dec. 2001, pp. 245 – 256.

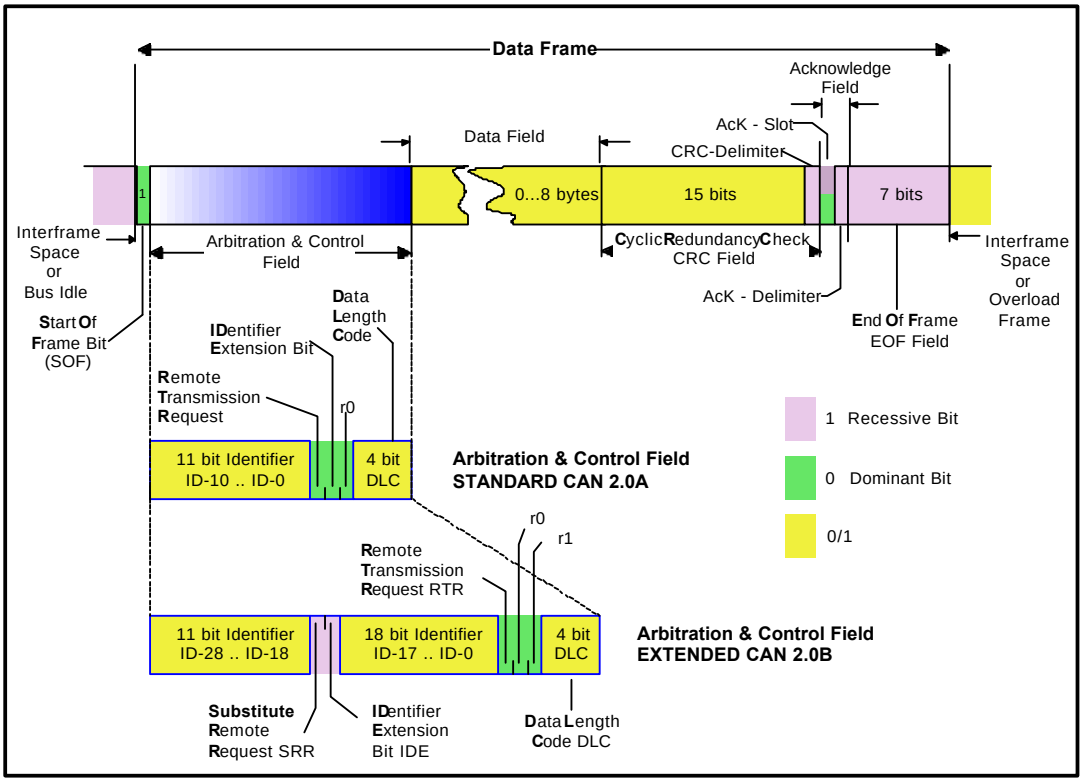

Figure 1 - CAN Bus Frame Formats

4. CAN Bus Physical Layers

The ESTEC CAN WG is analyzing suitable standards to be applied to the physical layer of the CAN bus. The aim of the physical layer specification is to define a standard interface to the CAN bus that supports implementation in the overwhelming majority of ESA space missions where the performance requirements of specific implementations are commensurate with that of the CAN bus.

The goal is to leverage the physical layer from commercial industry, which means primarily the automotive and related industries. The physical layer specification includes definition that can be met using off-the-shelf components that are either space qualified or likely to be qualify-able in terms of component performance and regarding radiation tolerance and other key space environment concerns.

The selected approach to ensure device-on-bus electrical compatibility and device-across-the-industry electrical compatibility is to specify full adherence to [9] and [10] as the physical layer electrical reference requirement.

5. CAN Bus Higher Layer Protocols

The ESTEC CAN WG has looked into a number of existing Higher Layer Protocols in order to find a widely used and well-established protocol that would be suitable for use in space

applications. As a result of this exercise it quickly became clear that the CANOpen protocol [3] provided many of the services needed.

CANOpen is an open Higher Layer Protocol standard widely use in automation and industrial applications, including safety critical and maritime applications implementing redundant communication buses.

The following paragraphs give a brief overview of some of the main features of the CANOpen standard. The overview is not intended as an exhaustive description but rather to highlight a few key properties that have led to the conclusion that the adoption of the CANOpen protocol are useful also for space borne applications. For full details of CANOpen please refer to [3] and [4].

5.1 Object Dictionary

A key concept in CANOpen is the Object Dictionary (OD). The OD is essentially a group of objects accessible via the network in an ordered pre-defined fashion.

The OD defines how a particular CAN frame is to be interpreted and formatted both in terms of the CAN frame identifier as well as the interpretation of each individual bit in the frame. The OD also includes parameters for defining timing and communication characteristics of frames as well as areas for device specific definition and configuration parameters.

The Object Dictionary can thus be considered as a translation and configuration interface between the CAN bus and the Application residing in the node. A simplified illustration of the OD concept is shown in Figure 2.

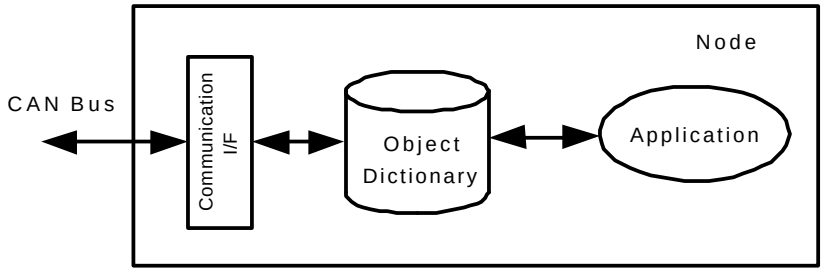

Figure 2 - Principle of the Object Dictionary

As an example, for certain nodes the OD may be used to define that a particular frame is to be interpreted as a pulse command. The OD would then also define which bits in the frame are to be interpreted as the output channel for the pulse. In addition there could be entries in the OD that define which bits in the frame contain other parameters such as pulse duration, voltage levels etc.

For frames transmitted from a node, the OD would, in addition to the frame formatting definitions, include entries specifying events to trigger the sending of the frame, minimum time between two subsequent frames etc.

When a node is to be integrated into a system, the Object Dictionary is configured according to the capabilities of the node and the needs derived from the system. By using this approach it is possible to reuse an existing piece of equipment in a quite different system environment simply by means of configuring the Object Dictionary appropriately.

A specific feature of the CANOpen OD is that it is possible to configure the OD of a node while it is integrated into a system. This is performed by means of dedicated CAN frames and eases the upgrading of a system during its lfetime. As an example, in case a new node is introduced in a system the allocation of CAN frame identifiers might need to be changed due to different frame priority assignments. This could be achieved by simply reprogramming the relevant OD entries of affected nodes via the CAN bus.

For a space application, the possibility to be able to perform in flight system upgrades is not normally a design driver. However, the capability to configure the object dictionary of nodes opens the door to some significant advantages. First, it increases the possibility of equipment reuse without the need for extensive re-qualifications similarly to the benefits seen in the commercial market.

Secondly, and this is may be even more important, it could ease the spacecraft integration activities. In particular considering the fact that the Object Dictionary concept provides for a structured and uniform way to perform in system adaptations without affecting the application. As an example, in case of late system changes or problems, the characteristic of the bus traffic such as frame priorities, timing, formatting etc. could easily be modified without the need for substantial software updates or equipment re-qualifications.

5.2 Communication Objects

CANOpen specifies two main categories of communication objects (CAN frames) transmitted on the CAN bus, the Service Data Object (SDO) and the Process Data Objects (PDO).

Service Data Objects are primarily used to access entries in the object dictionary of a node. An SDO is transferred as a series of segments (CAN frames) and enables the transfer of data of any length. The service is confirmed, implying that communication by means of SDOs is rather slow and slightly inefficient due to the confirmation frame that occupies bus bandwidth. However, since the SDOs are mainly intended for accessing and configuration of the object dictionaries in the nodes and not for real time communication the additional overhead is not of major importance.

In order to provide capability for more efficient data transfer of any length the CAN WG is defining a complementary Large Data Unit Transfer protocol. This protocol will allow unconfirmed transfers with reduced protocol overhead and will coexist with the CANOpen protocol on the same network. The details of the Large Data Unit Transfer protocol can be found in the draft standard [6].

Process Data Objects are used to perform real time communication on the CAN bus. The content and format of a PDO is defined by the corresponding entries in the Object Dictionaries of the communicating nodes.

Typically, a PDO can be used for the control of actuators and the acquisition of data, e.g. generation of pulse commands or acquisition of sensor values, but it is applicable for any time critical communication where the amount of data to be transferred is limited.

The main characteristics of the PDO are:

- data is transferred with no protocol overhead.
- used for data of a length from 1 up to 8 octets maximum.
- the frame is not confirmed.
- transferred from a single producer to one or more consumers.
- both the producer and the consumers of the PDO have corresponding entries in their OD defining the characteristics of the PDO.

A PDO can have several transmission modes. A particular feature is the capability to relate the sending of the PDO to a dedicated synchronization frame, SYNC. The SYNC frame can be sent periodically by a SYNC producer to synchronize the activities of all nodes in the network. Figure 3 illustrates this scheme. Each PDO can then be defined to react or not to the SYNC frame.

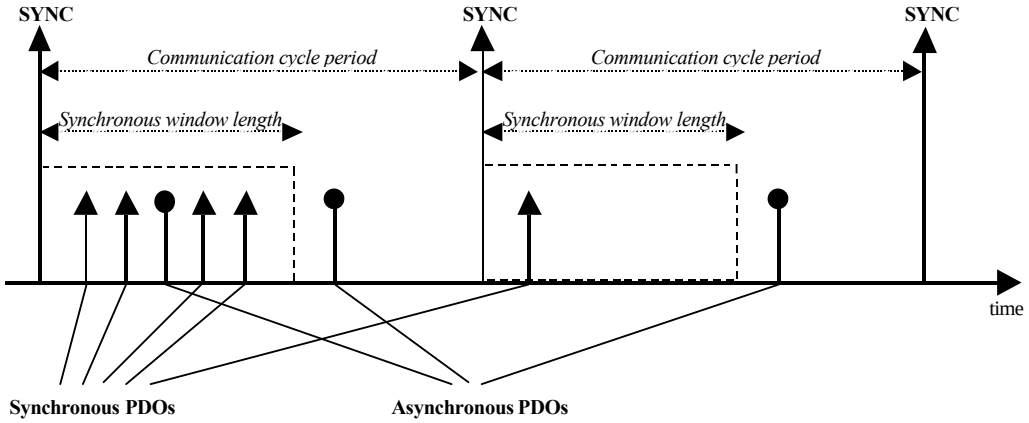

Figure 3 – Use of SYNC frame to control frame transmission

The transmission possibilities for a PDO can be summarized as follows:

- Synchronous
  The transmission of the PDO is triggered by a SYNC frame and can be either:
  - Acyclic
    - pre-triggered by another PDO received from another device on the network.
    - pre-triggered by the occurrence of a device specific event, e.g. a change of a value of a specific parameter.
  - Cyclic
    - the transmission of the PDO is triggered periodically after every 1, 2 or more SYNC frames.

- Asynchronous
  The transmission of the PDO is triggered by:
  - Remote Transmission Request received from another device.
  - occurrence of a device specific event.

From Figure 3 it can be seen that it is possible to transmit PDOs asynchronously at any time as long as the asynchronous PDO has a CAN frame identifier with sufficiently high priority to gain access to the bus. The PDO mechanism is very flexible. It enables systems to be designed as completely asynchronous, fully synchronous, or a combination of both.

It is important to remember that, unlike OBDH and Mil-Std-1553 buses, CAN and CANOpen allow ad-hoc, asynchronous frames to be transmitted whenever necessary, e.g. in case of alarm conditions detected by a remote device. The access to the bus of the various frames will be determined by the Medium Access Control function of the CAN bus based on the frame priority. It is up to the system designer to define the allocation of asynchronous and synchronous traffic based on his particular needs.

5.3 Node Control Services

In a typical network in a spacecraft there is a need to be able to control the states of each connected node. In addition, there is a need for services that allow detection of faulty nodes and network connection problems.

CANOpen specifies a number of Network Management Services (NMT), ([7], [8]) that allow management of the network and the nodes connected to the network.

The ***Module Control Services*** define services and protocols to be used to switch between states of a node. These services include Start Remote Node, Stop Remote Node, Enter Pre-Operational, Reset Node and Reset Communication.

Figure 4 shows the minimal state diagram specified in CANOpen. It includes the Initialization, Pre-Operational, Operational and Stopped states. After a reset the node will go through the Initilisation phase and then proceed to Pre-operational. In Pre-operational it is possible to set-up and configure the device should that be necessary. This includes the modification of parameters in the OD of the device.

When a node enters the Pre-Operational state from the Initialising state, the node transmits a *Bootup Event* frame to notify a NMT Master of the bootup.

It is worth noting that this state diagram is mandatory for all CANOpen nodes but it does not specify how the different operating modes of a node are to be implemented. This is application specific.

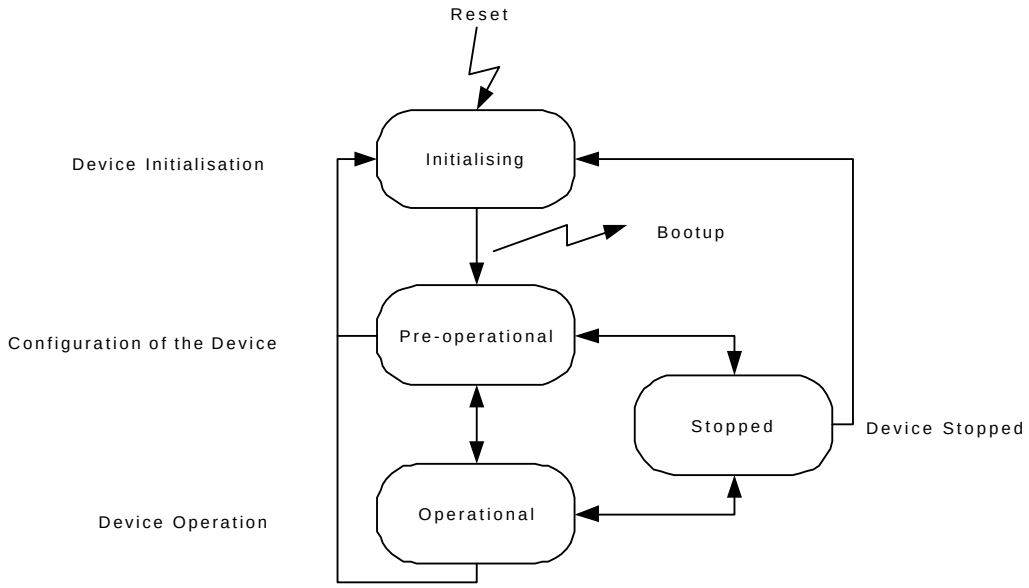

Figure 4 - CANOpen Node State Diagram

The ***Error Control Services*** provide mechanisms to detect failures in a CAN based network. The Error Control Services include *Node Guarding*, *Life Guarding* and *Heartbeat* services. The *Node Guarding* and *Life Guarding* services are based on requests issued by an NMT Master.

In the case of *Node Guarding* the NMT Master issues a request to an individual node and expects a reply within a certain time. In the case of *Life Guarding* a remote node expects guarding requests from the NMT Master within a certain time. The node will take some predefined action in case the guard frames are not received within the defined time.

The *Heartbeat* service is not based on requests. Instead each node transmits a heartbeat frame autonomously with a certain periodicity. One or several nodes on the network will listen to this heartbeat and can take action if the heartbeat frame is not received within the defined time.

In general the node control services specified by CANOpen are well suited for use in the space domain. CANOpen specifies the way to manage the on-off switching of nodes in the network. The specified services can be applied also in redundant systems where well defined control and on-off management of equipments is of major importance. In addition, the *Bootup Event* is very suitable for spacecraft systems since it provides a means for a node to directly communicate to the rest of the system that it has rebooted for some reason.

The use of the *Heartbeat* service is currently considered as the baseline for use in a space application. The main reason being lower bandwidth utilization compared to the guarding schemes. In fact the use of *Heartbeat* as a mechanism to control the switching between a nominal and redundant bus is currently under discussion within the CAN WG.

5.4 Time Distribution

Virtually all space missions require that the spacecraft provide the capability to maintain and distribute time information on board. The time information is used for a variety of functions including time stamping of measurement data and control and scheduling of delayed execution of telecommands.

The on board time is implemented as a centralized time reference. However, there are typically many types of equipment on board a spacecraft that also need to maintain local time information. A mechanism to distribute the central time and to keep the local times synchronized with the central time reference is thus required.

CANOpen specifies two optional protocols and time formats for time distribution and synchronization. Both are suitable for space use but due to its higher accuracy, the CANOpen High Resolution Time Stamp protocol is generally preferred. The specified protocol provides the best possible synchronization accuracy that can be achieved when distributing time over the CAN bus. Disregarding implementation uncertainties and delays the specified protocol allows for synchronization accuracy in the microsecond range.

Due to differences in the data types used for representing the time in CANOpen compared to existing standards for spacecraft on board time, the CANOpen data types are not directly suitable to be applied for space applications. The ESTEC CAN WG recommendation thus specifies the use of the CANOpen High Resolution Time Stamp protocol but with data types and formats compatible with the ones traditionally used on board spacecraft. In addition to the time distribution and synchronization protocol an optional protocol to read the local time of a node is specified. For details about the protocol please refer to [3] and [6].

6. Conclusions

The CAN specification can be used together with appropriate electrical interfaces and higher layer protocols to implement a spacecraft onboard bus. CAN offers many advantages, particularly low cost, ready availability of components, and widespread expertise. Moreover, the CAN bus uses concepts that are significantly different to more traditional spacecraft buses and therefore opens the door to new spacecraft architectures.

The ESTEC CAN working group is actively developing recommendations that will ease the implementation of the CAN bus, and will help to ensure consistent use of this bus in spacecraft, will increase the reuse potential of flight equipment, and will enable projects to reap the full benefits offered by CAN.

7. References

[1] ISO/IEC 7498-1
Information Technology – Open Systems Interconnection – Basic Reference Model: The Basic Model
Second Edition 1994-11-15

[2] CAN Bus Specification
Version 2.0 Part B, 1991, Bosch GmbH, R, Stuttgart, Germany

[3] CANOpen Application Layer and Communication Profile,
CiA Draft Standard 301, Version 4.02, 13 February 2002
http://www.can-cia.de

[4] CAN in Automation Home Page
http://www.can-cia.de

[5] CAN WG Discussion Group
http://groups.yahoo.com/group/CAN_Space/

[6] CAN WG Home Page
http://users.skynet.be/cotectic/CAN-WG/

[7] CAN Application Layer for Industrial Applications,
NMT Service Specification, CiA/DS 203-1
http://www.can-cia.de

[8] CAN Application Layer for Industrial Applications,
NMT Protocol Specification, CiA/DS 203-2
http://www.can-cia.de

[9] ISO 11898:1993
Road vehicles – Interchange of digital information – Controller area network (CAN) for high-speed communication, First Edition 1993

[10] ISO 11898:1995 Amendment 1 (Amended 1995)
International Organisation for Standardisation, Geneva, Switzerland, ISO, ISO 11898:1993(E)