CiA Draft Standard Proposal 447

Application profile for special-purpose car add-on devices

Part 1: General definitions

This DSP is for CiA members only and may be changed without notification.

Version: 1.0

21 May 2008

HISTORY

Date	Changes
2008-05-21	*Publication of version 1.0 as draft standard proposal*

General information on licensing and patents

CAN in AUTOMATION (CiA) calls attention to the possibility that some of the elements of this CiA specification may be subject of patent rights. CiA shall not be responsible for identifying any or all such patent rights.

Because this specification is licensed free of charge, there is no warranty for this specification, to the extent permitted by applicable law. Except when otherwise stated in writing the copyright holder and/or other parties provide this specification "as is" without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The entire risk as to the correctness and completeness of the specification is with you. Should this specification prove failures, you assume the cost of all necessary servicing, repair or correction.

Trademarks

CANopen® and CiA® are registered community trademarks of CAN in Automation. The use is restricted for CiA members or owners of CANopen vendor ID. More detailed terms for the use are available from CiA.

CAN in Automation e. V.
Kontumanzgarten 3
DE - 90429 Nuremberg, Germany
Tel.: +49-911-928819-0
Fax: +49-911-928819-79
Url: www.can-cia.org
Email: headquarters@can-cia.org

CONTENTS

1 Scope 5
2 Normative references..5
3 Definitions and abbreviations 5
3.1 Definitions 5
3.2 Abbreviations..6
4 Physical layer specification 6
4.1 Introduction 6
4.2 Transmission rates 7
4.3 Connectors..7
5 Node-ID assignment 8
5.1 Introduction 8
5.2 Node-ID range..8
5.3 Usage for LSS services 8
6 Boot-up process..9
6.1 Introduction 9
6.2 NMT master configuration..9
6.3 Boot-up procedure 10
6.4 Sub-processes of the boot-up procedure 12
6.4.1 Sub-process ‘Detect next fix slave’ (see Figure 2) 12
6.4.2 Sub-process ‘Detect ‘old’ LSS slave’ (see Figure 2)..12
6.4.3 Sub-process ‘Prepare operation’ (see Figure 2)..12
6.4.4 Sub-process ‘Dynamic LSS detection’..13
7 Operating principles..13
7.1 Introduction 13
7.2 NMT master and slave functionality 13
8 Error and diagnostic handling..13
8.1 Introduction 13
8.2 Error behavior..13
8.3 Additional error codes..14
9 Data type definitions 14
9.1 Introduction 14
9.2 UTF8 and UTF8 string 14
10 Record definitions 15
10.1 Object 0080$_h$: Start route guidance record..15
10.2 Object 0081$_h$: Taximeter configuration record..15
11 General communication parameter..15
11.1 Introduction..15
11.2 1000$_h$: Device type 15
11.3 1001$_h$: Error register..16
11.4 1003$_h$: Pre-defined error field 16
11.5 1016$_h$: Heartbeat consumer time 16
11.6 1017$_h$: Heartbeat producer time..16
11.7 Application parameters for CANopen devices..16
11.7.1 Introduction 16
11.7.2 Object 6000$_h$: Virtual device support..16

Annex A Power management (normative) 18
A.1 Scope 18
A.2 Operation principles 18
A.2.1 Introduction 18
A.2.2 Finite state automaton 18
A.2.3 Services 19
A.2.4 Protocols 21
Annex B Timing information (normative) 24
B.1 Scope 24
B.2 Protocol patterns 24
B.2.1 Introduction 24
B.2.2 Handshake pattern 24
B.2.3 Consecutive pattern 24
B.3 Network management (NMT) timing 25
B.3.1 Introduction 25
B.3.2 Boot-up 25
B.3.3 NMT state transitions 26
B.4 LSS timing 27
B.4.1 Introduction 27
B.4.2 LSS identify non-configured remote slave 27
B.4.3 LSS FastScan service 27
B.5 Power management timing 28
B.5.1 Sleep/wake-up 28
B.6 Miscellaneous timing values 29

1 Scope

This CANopen application profile specifies the CAN physical layer as well as application, configuration and diagnostic parameters for the add-on devices used in special-purpose passenger cars such as taximeter, roof bar, etc. The specification comprises the following parts:

- Part 1: General definitions
- Part 2: Virtual device definition
- Part 3: Detailed process data specification
- Part 4: Pre-defined CAN-IDs and communication objects

This part defines the physical layer, the general system architecture, and some common communication parameter objects.

2 Normative references

/CiA301/	CiA 301, CANopen application layer and communication profile
/CiA302-1/	CiA 302, CANopen additional application layer functions – Part 1: General definitions
/CiA302-2/	CiA 302, CANopen additional application layer functions – Part 2: Network management
/CiA305/	CiA 305: CANopen layer setting services (LSS) and protocols
/ISO11898-1/	ISO 11898-1, Road vehicles – Controller area network (CAN) – Part 1: Data link layer and physical signaling
/ISO11898-2/	ISO 11898-2, Road vehicles – Controller area network CAN) – Part 2: High-speed medium access unit
/ISO15765-3/	ISO 15765-3, Road vehicles – Diagnostics on Controller Area Networks (CAN) – Part 3: Implementation of unified diagnostic services (UDS on CAN)
/ISO14229-1/	ISO 14229-1, Road vehicles – Unified diagnostic services (UDS) – Part 1: Specification and requirements
/ISO10646/	ISO 10646, Information technology – Universal multiple-octet coded character set (UCS)

3 Definitions and abbreviations

3.1 Definitions

Fix-ID
Fixed programmed node-ID

Fix slave
Device of category fix-ID or HW-ID

Full meshed SDO communication
Ability of each CANopen device to communicate with each other CANopen device in a peer-to-peer manner

Functional server
Instance that implements the application part of a function. The CANopen objects of the corresponding virtual device act as an interface to the CAN network. Example: A physical printer device implements the functional server of the virtual device printer.

Functional client
Instance of the application, which uses an interface for accessing a function via the CAN network. Objects of the corresponding virtual device are required only as a buffer for data exchange, if PDOs are used. Example: Any device that requests printer services may want to receive the printer status via PDO. For defining of a valid RPDO mapping, it needs to implement the *printer status* object. In fact, this is only a receive buffer and does not implement a real printer function. Therefore this part of the virtual device is called the functional client.

HW-ID
Selectable node-ID via hardware settings like DIP-switches, jumpers or other, or via manufacturer-specific software settings

LSS-ID
Variable node-ID set by means of LSS services

LSS slave
Device of category LSS-ID

NMT master
In-vehicle network gateway device performing the CANopen NMT master services and protocols

The definitions given in /ISO11898-1/, and /ISO11898-2/ apply to this specification, too.

3.2 Abbreviations

CAN	Controller area network
CAN-ID	CAN identifier
DLC	Data length code
DST	Daylight saving time
IVN	In-vehicle network
LSB	Least significant bit
MSB	Most significant bit
PDO	Process data object
PTT	Push-to-talk
RAM	Random access memory
RPDO	Receive PDO
RTC	Real time clock
SDO	Service data object
TP	Transport protocol
TPDO	Transmit PDO
UDS	Unified diagnostic services
UTC	Universal time coordinated

4 Physical layer specification

4.1 Introduction

The general physical layer specification given in /ISO11898-2/ applies to devices compliant to this specification, too.

4.2 Transmission rates

The device shall use only the transmission rate of 125 kbit/s. The bit timing as defined in /CiA301/ shall be used.

4.3 Connectors

It is recommended to use the 18-pin VDA interface connector (e.g. *micro quadlok system 0.64* from Tyco or equivalent connectors from other manufacturers). The 18-pin VDA interface socket connector (car side) is shown in Figure 1.

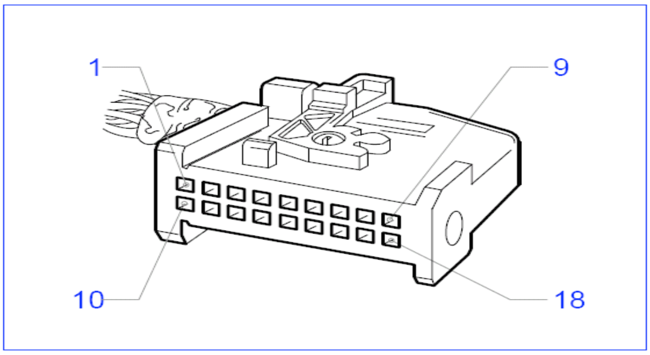

Figure 1 — 18-pin VDA interface socket connector (car side)

Table 1 specifies the pin assignment of the 18-pin VDA interface connector. The signals S1 to S6 are car add-on device application-specific signals.

Table 1 — Pin assignment of the 18-pin VDA connector

Optional/ mandatory	Pin no.	Name	Description	Recommended use in taxi	Recommended use in emergency vehicle	Input/output from car view
Mandatory if 2-pin power-connector not used	1	KL31	Ground (0 V/ max. 4 A)	Ground (0 V/ max. 4 A)	Ground (0 V/ max. 4 A)	Output
Optional	2	PTT	Reserved for PTT	PTT	PTT mobile radio	Input/output
Optional	3	S1	Reserved for PTT	Not used	PTT siren	Input/output
Optional	4	S2	Reserved for audio mute	Reserved for audio mute	Reserved for audio mute	Input/output
Optional	5	S3	Reserved	Passenger detection status	Beacon low	Input/output
Optional	6	S4	Reserved	Radio emergency call request	Radio emergency call request	Output
Optional	7	AUDIO_OUT +	Reserved for audio receiver	Speaker +	AUDIO_OUT +	Input/output
Mandatory	8	CAN_L	CAN low line	CAN_L	CAN_L	Bus
Optional	9	AUDIO_IN -	Reserved for microphone shield	MIC -	AUDIO_IN -	Input/output
Optional	10	KL15	Ignition	KL15	KL15	Output

Optional/ mandatory	Pin no.	Name	Description	Recommended use in taxi	Recommended use in emergency vehicle	Input/output from car view
Mandatory if 2-pin power-connector not used	11	KL30	Power supply voltage (+11 V to 16 V/ max. 4 A)	Power supply voltage (+11 V to 16 V/ max. 4 A)	Power supply voltage (+11 V to 16 V/ max. 4 A)	Output
Optional	12	KL58	Low beam status	KL58	KL58	Output
Optional	13	SPEED_PULS	Speed pulse signal	Speed pulse signal	Speed pulse signal	Output
Optional	14	S5	Reserved	Taximeter status	Radio main switch	Input/output
Optional	15	S6	Reserved	Roof sign status request	Beacon high	Input/output
Optional	16	AUDIO_GND	Reserved for ground for audio signals	Speaker -	AUDIO_OUT -	Input/output
Mandatory	17	CAN_H	CAN high line	CAN_H	CAN_H	Bus
Optional	18	AUDIO_IN/MIC	MIC +	MIC +	AUDIO_IN +	Input/output

For high power devices (> 4 A) a 2-pin power connector AMP926474-1 or an equivalent connector from Tyco or other manufacturer shall be used. Table 2 specifies the pin assignment of the 2-pin power connector.

Table 2 — Pin assignment of the 2-pin power connector

Optional/ mandatory	Pin no.	Name	Description	Recommended use in taxi	Recommended use in emergency vehicle	Input/output from car view
Recommended for current > 4A	1	KL31	Ground (0 V/ max. 15 A)	Ground (0 V/ max. 15 A)	Ground (0 V/ max. 15 A)	Output
Recommended for current > 4A	2	KL30	Power supply voltage (+11V to 16V/ max. 15A)	Power supply voltage (+11V to 16V/ max. 15A)	Power supply voltage (+11V to 16V/ max. 15A)	Output

5 Node-ID assignment

5.1 Introduction

The node-ID assignment is manufacturer-specific. It is recommended to use the LSS services using the LSS FastScan procedure as defined in /CiA305/ for node-ID assignment.

5.2 Node-ID range

The valid range of node-IDs for car add-on devices shall be 1 to 16. The IVN gateway device shall have the node-ID 1. It serves as NMT master and LSS master.

All other devices shall support one of the following categories:

- Fix-ID
- HW-ID
- LSS-ID

If the categories Fix-ID and HW-ID are used, the system integrator shall guarantee that the same node-ID is not used twice.

5.3 Usage for LSS services

Devices of the categories Fix-ID and HW-ID shall not respond to LSS services at all.

Each device shall implement the following services required for power management as defined in Annex A.

- Query sleep objection
- Sleep objection
- Set sleep mode
- Wake-up
- Request sleep

Devices of the category LSS-ID shall support the following services (see /CiA305/):

- LSS identify non-configured remote slave
- LSS identify non-configured slave
- LSS FastScan
- LSS identify slave
- Switch state global
- Switch state selective
- Configure node-ID

Devices of the category LSS-ID shall not store their node-ID non-volatile – after a power-up or performance of NMT Reset node service they always shall start with node-ID FF_h.

For timing relevant information of LSS FastScan procedure see chapter B.4.3.

A device of category LSS-ID with a node-ID outside the range 2 to 16 shall immediately reset its node-ID to FF_h.

6 Boot-up process

6.1 Introduction

A network is started according to the procedure as defined in chapter 6.3. For a self-starting device the object $1F80_h$ is mandatory. Self-starting devices are not recommended.

6.2 NMT master configuration

The CANopen device containing the IVN gateway virtual device shall be the NMT master. Additionally it shall provide the following functions and parameters:

- LSS master supporting the “FastScan” procedure. The procedure shall be started within five seconds after the first message of special car add-on network.
- Object $1F80_h$: The access attribute shall be ro (read only), the parameter value shall be 09_h. Table 3 shows the description of object $1F80_h$ bit settings.

Table 3 – Description of object $1F80_h$ bit settings

Bit	Value	Meaning
0 - Value NMT master	1_b	CANopen device is the NMT master
1 - Value Start all nodes	0_b	NMT service start remote node for each node-ID separately
2 - Value NMT master start	1_b	Shall not switch into the NMT state Operational by itself.
3 - Value Start node	1_b	The NMT master shall not start the NMT slaves and the application may start the NMT slaves.
4 - Reset all nodes	0_b	Not relevant
5 - Flying master	0_b	CANopen device shall not participate the NMT flying master negotiation
6 - Stop all nodes	0_b	Not relevant

The physical device implementing the IVN gateway virtual device needs not to provide:

- Configuration manager
- SDO manager
- Flying master
- Objects $1F81_h$, $1F84_h$ to $1F91_h$

6.3 Boot-up procedure

The process NMT start-up shall consist of the steps as specified in Figure 2.

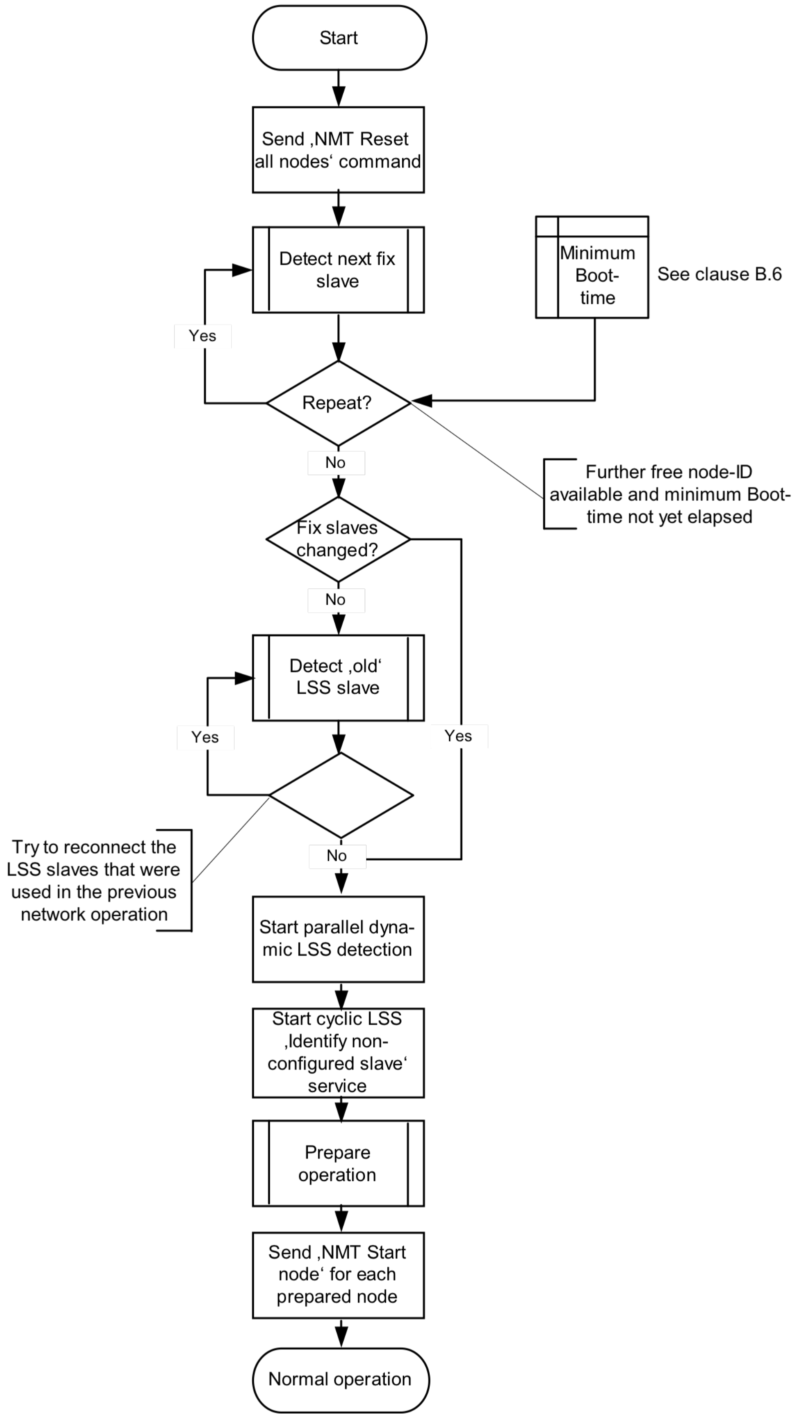

Figure 2 – The process NMT Start-up

The basic steps are:

a) The NMT master shall perform the NMT Reset node service with node-ID set to 0 (all nodes).

b) The NMT master shall run the processes ‘Detect next fix slave’ as specified in chapter 6.4.1. If the list of fix NMT slaves (devices of category Fix-ID) changed the process continues with step d).

c) The NMT master shall detect, if the same set of LSS slaves is available as with the last network operation. It shall do so by using the sub-process ‘Detect ‘old’ LSS slave’ for each of the stored LSS slave settings according to chapter 6.4.2.

d) The NMT master shall start the ‘Dynamic LSS detection’ according to chapter 6.4.4.

e) The NMT master shall start the cyclic service ‘LSS identify non-configured remote slave’. The recommended cycle time is 1 s.

f) The NMT master shall start the process ‘Prepare operation’ according to chapter 6.4.3.

g) The NMT master shall transmit the command NMT Start node to each detected node individually.
If it is ensured by the application, that no outstanding LSS slaves are in a status, which may be affected by the NMT command, then the NMT master may instead use the command NMT Start all nodes.

h) The process is finished; the network is in normal operation.

6.4 Sub-processes of the boot-up procedure

6.4.1 Sub-process ‘Detect next fix slave’ (see Figure 2)

The sub-process ‘Detect next fix slave’ has the task to detect slaves with a Fix-ID and HW-ID according to chapter 5.2.

In order to achieve this, the device with NMT master functionality shall read for each possible node-ID in the range 2 to 16 the object 1000_h, 6000_h and the sub-indexes 01_h to 04_h of object 1018_h. It shall do so with running the node-IDs sequentially from 2 to 16 and then starting with 2 again until for a node-ID an NMT slave has responded or the Minimum boot-up time has elapsed and each node-ID was scanned at least twice. The value of the Minimum boot-up time is specified in Table 19.

NOTE: This mechanism allows detecting slaves with different boot-up times. The term ‘sequential’ leaves it open to send a request after the other without waiting for the response and afterwards gathering the responses. If doing so, the time-out of each SDO request should be slightly less than half the Minimum Boot-up time.

6.4.2 Sub-process ‘Detect ‘old’ LSS slave’ (see Figure 2)

It is recommended that the LSS master stores the list of attached LSS slaves after each time an LSS slave is attached or detached. This list includes the complete LSS addresses and the assigned node-IDs of all connected devices.

NOTE: The list described above is not the NMT slave assignment (object $1F81_h$) as defined in /CiA302-2/.

If the LSS master supports this storage, it shall distribute the node-IDs according to the stored information. For this purpose the services “LSS identify slave”, “Configure node-ID” and “Switch state selective” according to /CiA305/ shall be used.

NOTE: On switching an LSS slave to “waiting mode” the NMT slave will start-up with the set node-ID.

6.4.3 Sub-process ‘Prepare operation’ (see Figure 2)

The sub-process ‘Prepare operation’ is up to the IVN gateway device manufacturer. This specification gives only a recommendation, as follows:

a) The NMT master or the application on the IVN gateway may start Heartbeat services according to /CiA302-2/. It may have to configure the corresponding object dictionary entries on the NMT slaves.
b) The NMT master may evaluate the attached NMT slaves by reading the entries for the implemented virtual devices and by storing them internally.

c) The NMT master may pass this information to the application.

d) The application may set-up, configure or re-configure PDOs depending on the available virtual devices on the local IVN gateway device as well as on the remote devices. Furthermore the application may configure additional entries in this phase or at any time later.

e) The application should set the local device into NMT Operational state.

6.4.4 Sub-process 'Dynamic LSS detection'

In the sub-process 'Dynamic LSS detection' the NMT master shall start the cyclic transmission of the service LSS FastScan. It shall apply the definitions of chapter 5.3.

For detection of each non-configured LSS slave the NMT master shall apply LSS configuration after the sub-process 'Prepare operation' according to chapter 6.4.3.

7 Operating principles

7.1 Introduction

Each CANopen special-purpose car add-on device implements one or more virtual devices as specified in this application profile. Which virtual devices are implemented in a CANopen device is indicated in the *virtual device support* (object 6000_h).

Virtual devices are described generally in /CiA301/. The virtual devices share the object dictionary index range 6000_h to $67FE_h$. It is possible to implement more than one CANopen virtual device in one CANopen logical device. A single virtual device shall not be distributed to several CANopen devices. Each virtual device implements different process and configuration parameter, some shall be supported (Mandatory) and some may be supported (Optional). One virtual device of the same type may be implemented in each physical CANopen device. Several virtual devices of the same type shall not be implemented in one physical CANopen device.

7.2 NMT master and slave functionality

CANopen devices compliant to this application profile shall support NMT slave functionality as specified in /CiA301/. The CANopen device that implements the car IVN gateway virtual device shall support additionally NMT master functionality and shall use the boot-up procedure specified in chapter 6.

CANopen devices compliant to this application profile shall provide Heartbeat functionality as specified in /CiA301/. They shall not support Node and Life guarding functionality.

8 Error and diagnostic handling

8.1 Introduction

CANopen devices compliant to this application profile shall support the Emergency service and the Emergency protocol as specified in /CiA301/.

Emergency messages are triggered by internal errors in the device. They are structured as specified in /CiA301/. The manufacturer-specific error field (msef) shall be reserved for future use. The structure of the Emergency message is specified in /CiA301/ and shown in Figure 3.

63 ... 24	23 ... 16	15 ... 0
msef	*er*	*eec*
MSB		LSB

Figure 3 – Structure of the Emergency message

8.2 Error behavior

If a severe device failure is detected, the device shall automatically enter by default the NMT Pre-operational state (see /CiA301/).

If object 1029_h (see /CiA301/) is implemented, the device may be alternatively configured in case of a device failure to automatically enter the NMT Stopped state or remain in the current NMT state.

Device failures shall include the following communication errors:

- Bus-off conditions on the CAN interface
- Heartbeat event with state ‘occurred’

Severe device errors may also be caused by device internal failures.

8.3 Additional error codes

CANopen devices compliant to this application profile shall support the additional error codes defined in Table 4.

Table 4 – Additional error codes

eec	Description
$F001_h$	General warning
$F002_h$	Severe warning
$F003_h$	General failure
$F004_h$	Severe failure
$F005_h$	Partial network operation
$F006_h$	Network start-up
$F007_h$	Buses in sleep
$F008_h$	Communication failure in the base vehicle networks

9 Data type definitions

9.1 Introduction

A data type determines a relation between values and encoding for data of that type. For general definitions see /CiA301/.

9.2 UTF8 and UTF8 string

The data type UTF8 is defined by Unsigned8, where the interpretation is according to the UTF8 encoding of /ISO10646/.

The data type UTF8 string[length] is defined by:

ARRAY [length] OF UTF8 UTF8 string[length]

Table 5 specifies the data types UTF8 and UTF8 string.

Table 5 – UTF8 and UTF8 string definitions

Index	Object	Name
$007B_h$	DEFTYPE	UTF8
$007C_h$	DEFTYPE	UTF8 string

10 Record definitions

10.1 Object 0080$_h$: Start route guidance record

Table 6 specifies the record structure. The values of the *start route guidance* sub-objects are defined in the objects using this data type.

Table 6 — Record structure

Sub-index	Parameter	Data type
00$_h$	Highest sub-index supported	Unsigned8
01$_h$	Position latitude	Unsigned32
02$_h$	Position longitude	Unsigned32
03$_h$	Start guidance	Unsigned8

10.2 Object 0081$_h$: Taximeter configuration record

Table 7 specifies the record structure. The values of the *taximeter configuration* sub-objects are defined in the objects using this data type.

Table 7 — Record structure

Sub-index	Parameter	Data type
00$_h$	Highest sub-index supported	Unsigned8
01$_h$	Constant of the distance signal generator (k value)	Unsigned32
02$_h$	Taximeter mode	Unsigned8
03$_h$	Identification of the tariff	Unsigned24
04$_h$	Taxi identifier	UTF8 string
05$_h$	Calculation of fare	UTF8 string
06$_h$	Date of securing	Unsigned32
07$_h$	Identification of future tariff	Unsigned24
08$_h$	Actual currency	UTF8 string
09$_h$	Actual local distance unit	UTF8 string

11 General communication parameter

11.1 Introduction

The general communication parameters are specified in /CiA301/. In clause 11 additional specifications are given.

11.2 1000$_h$: Device type

The object shall describe the type of device and its functionality. Figure 4 defines the value structure. The object description and entry description are defined in /CiA301/.

31 … 16	15 … 0
Reserved for future use	*Device profile number:* 447$_d$
MSB	LSB

Figure 4 – Object structure

11.3 1001_h: Error register

The device profile specific bit in the error register object is reserved. For details see /CiA301/.

11.4 1003_h: Pre-defined error field

The pre-defined error field object shall be implemented. For details see /CiA301/.

11.5 1016_h: Heartbeat consumer time

The heartbeat consumer time may be implemented. For details see /CiA301/. A heartbeat consumer time of 700 ms is recommended.

11.6 1017_h: Heartbeat producer time

The heartbeat producer time shall be implemented. For details see /CiA301/. A heartbeat producer time of 200 ms is recommended.

11.7 Application parameters for CANopen devices

11.7.1 Introduction

The parameters described in clause 11.7 are common for CANopen devices compliant to this application profile. They describe the behavior of the CANopen physical device implementation.

11.7.2 Object 6000_h: Virtual device support

This object shall provide the information on supported virtual devices. Only the functional server for the virtual devices shall provide appropriate bits in this object. Figure 5 specifies the structure of the object, and Table 8 defines the values.

31	0
Bit 31 to Bit 0	
MSB	LSB

Figure 5 – Object structure of each sub-index

Table 8 – Value definition for sub-index 01_h

Bit	Definition		
0	IVN gateway class 0	1_b: implemented	0_b: not implemented
1	IVN gateway class 1	1_b: implemented	0_b: not implemented
2	IVN gateway class 2	1_b: implemented	0_b: not implemented
3	IVN gateway class 3	1_b: implemented	0_b: not implemented
4	Fire extinguishing system	1_b: implemented	0_b: not implemented
5	Emergency fresh-air system	1_b: implemented	0_b: not implemented
6	Power supply	1_b: implemented	0_b: not implemented
7	Discrete inputs	1_b: implemented	0_b: not implemented
8	Terminal	1_b: implemented	0_b: not implemented
9	GPS	1_b: implemented	0_b: not implemented
10	Navigation system	1_b: implemented	0_b: not implemented
11	Taximeter	1_b: implemented	0_b: not implemented
12	Printer	1_b: implemented	0_b: not implemented

Bit	Definition		
13	Real time clock (RTC)	1_b: implemented	0_b: not implemented
14	Driver identification	1_b: implemented	0_b: not implemented
15	Tariff display	1_b: implemented	0_b: not implemented
16	Taxi alarm system	1_b: implemented	0_b: not implemented
17	Radio	1_b: implemented	0_b: not implemented
18	Audio switch	1_b: implemented	0_b: not implemented
19	Roof bar light	1_b: implemented	0_b: not implemented
20	Roof bar sound	1_b: implemented	0_b: not implemented
21	“Blue” light flasher module	1_b: implemented	0_b: not implemented
22	Roof bar controller	1_b: implemented	0_b: not implemented
23	Radio controller	1_b: implemented	0_b: not implemented
24	Handicap controller	1_b: implemented	0_b: not implemented
25	Radio hand-free conversation	1_b: implemented	0_b: not implemented
26	Tester/tool	1_b: implemented	0_b: not implemented
27 to 31	Reserved (always 0)		

Table 9 specifies the object description, and Table 10 specifies the entry description.

Table 9 – Object description

Attribute	Value
Index	6000_h
Name	Virtual device support
Object code	Array
Data type	Unsigned32
Category	Mandatory

Table 10 – Entry description

Attribute	Value
Sub-Index	00_h
Description	Highest sub-index supported
Entry category	Mandatory
Access	const
PDO mapping	No
Value range	01_h
Default value	01_h
Sub-Index	01_h
Description	Virtual devices 1
Entry category	Mandatory
Access	const
PDO mapping	No
Value range	See value definition
Default value	Device-specific

Annex A Power management (normative)

A.1 Scope

This annex specifies services and protocols for power management, such as sleep and wake-up mechanisms. It considers CAN transceivers, which support power management and are wake-up capable. A transceiver is wake-up capable if it is capable of waking up via the bus. The reason may be just a dominant state detected on the bus and not a complete message.

A.2 Operation principles

A.2.1 Introduction

The power management provides services and protocols to either set all or none of the devices, which support power management, to a mode of reduced power consumption.

A.2.2 Finite state automaton

The power management state machine as shown in Figure 6 shall be implemented, if the device supports power management services. This state machine may be entered from NMT states pre-operational, operational or stopped.

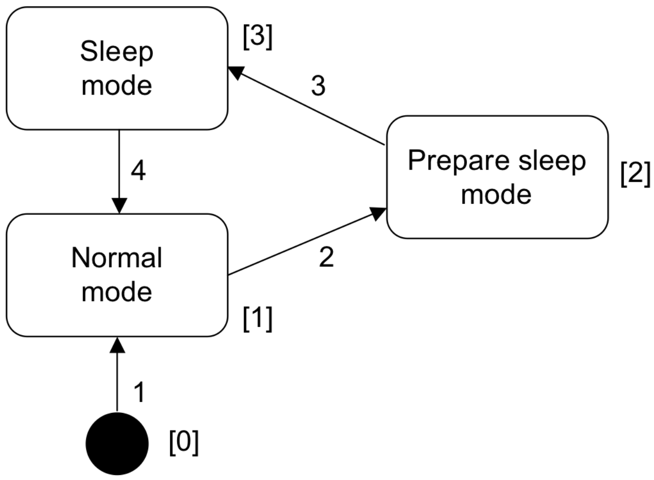

Figure 6 – Power management state machine

The power management FSA shall provide the following states:

[0] Initial: Pseudo state, indicating the activation of the FSA.

[1] Normal mode: In this state, the power consumption of the device is at the default level.

[2] Prepare sleep mode: On entering of this mode the device shall stop communication activities. The transmission of messages is inhibited. Received messages shall be ignored. This state is left based on time condition (t_{swt} = sleep wait time).

NOTE: In this state the transmission and reception of Heartbeat messages is disabled as well. Requests from the local SDO client, Emergency messages and transmission of PDOs are inhibited.

[3] Sleep mode: In this state, the power consumption of the device is reduced. The device applies the actual sleep policy. The state is immediately left, if it was entered with an active wake-up reason. A wake-up reason may be activated by a local event (detected via local inputs) or caused by activity on the network (this may be detected by the transceiver via the remote wake-up detection capability). A wake-up reason needs to be handled by the local application and therefore requires leaving the “sleep mode”.

The power management FSA shall provide the following transitions:

1. Initialization
2. Caused by the “set sleep mode” service. Usually a “query sleep objection” service is executed prior.
3. State transition after expiration of the sleep wait time t_{swt}. The node-ID of a device with category LSS-ID shall be reset on this transition.
4. State transition is caused if:

- The CAN transceiver has recognized a message on the network
- The application of the device requests wake-up

A device performing this transition shall enter the CANopen NMT state machine in the state Initialization.

A.2.3 Services

A.2.3.1 Service “query sleep objection” and “sleep objection”

By means of the “query sleep objection” service, the master (device with the NMT master functionality) queries all devices in the network if a transition to sleep mode is possible. The “query sleep objection” service is triggered by the local application. All devices having a reason not to go into the sleep mode shall respond with the “sleep objection” message. If the service “wake-up” is detected during the objection time-out t_{oto}, the service “query sleep objection " is executed for a second time. Figure 7 shows the inhibited transition into sleep mode because of reception of an “sleep objection” message.

If the master does not receive a "sleep objection" message within the objection time-out t_{oto}, it may initiate the service "set sleep mode".

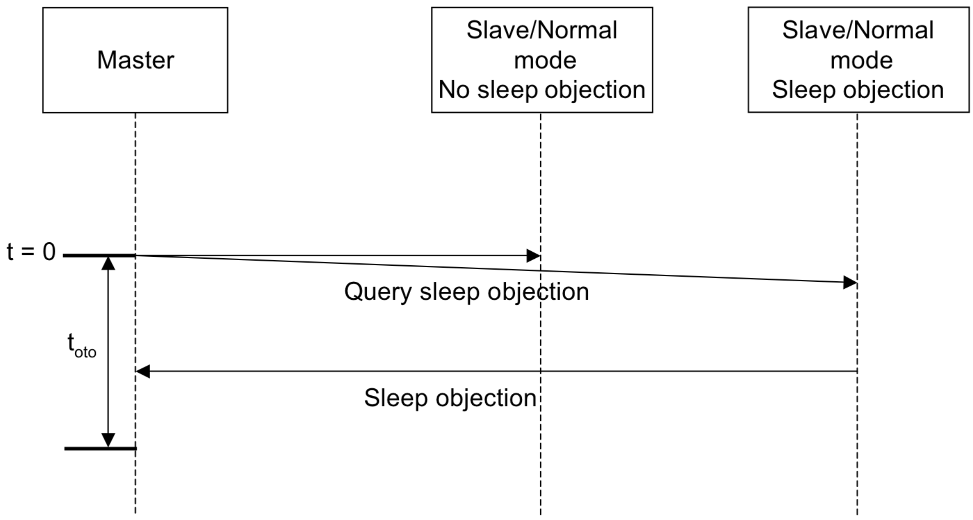

Figure 7 – Sleep mode inhibited by objection

A.2.3.2 Service “set sleep mode”

On receiving the service "set sleep mode", all slaves shall switch into the “prepare sleep mode”. In the “prepare sleep mode” a device shall not transmit or receive any message. After the sleep wait time t_{swt} the sleep mode is reached. Figure 8 shows the transition into the sleep mode without reception of “sleep objection” message. Figure 9 shows the case of execution of “query sleep objection” service for a device in sleep mode.

NOTE: It is not subject to this specification, which further actions a device in sleep mode internally performs and how it switches to reduced power mode.

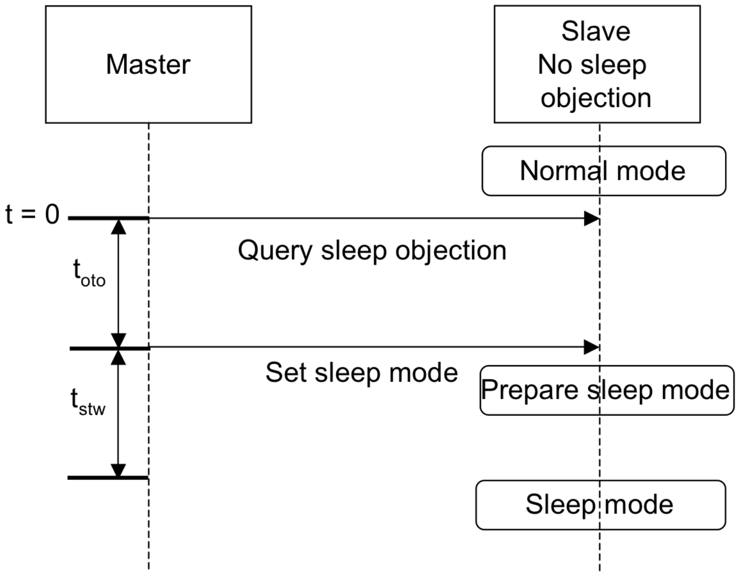

Figure 8 – Transition into sleep mode without objection

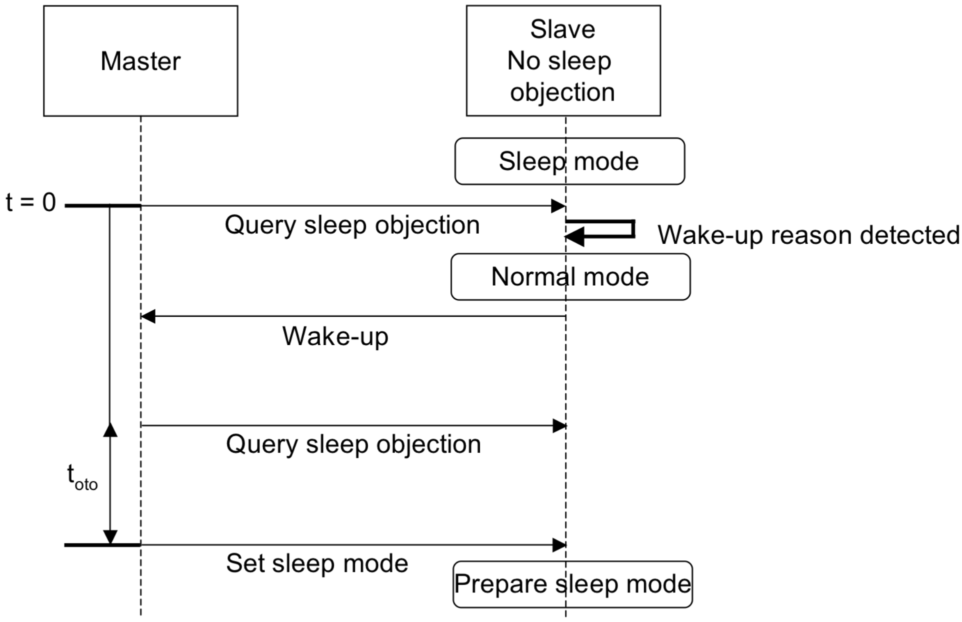

Figure 9 – Execution of "query sleep objection" service for a device in sleep mode

A.2.3.3 Service "wake-up"

A device may leave the sleep mode whenever a local wake-up reason is detected (e.g. the used CAN transceiver recognizes a message on the network). If the device is leaving the sleep mode, it shall initiate the service "wake-up". The "wake-up" message is intended to activate the CAN transceivers and corresponding hardware of the networked devices. If the NMT state machine of the device is not set to NMT Operational within the wake-up repetition time t_{wurpt} (see Table 19) the service "wake-up" shall be executed again.

Figure 10 shows the execution of the "wake-up" service.

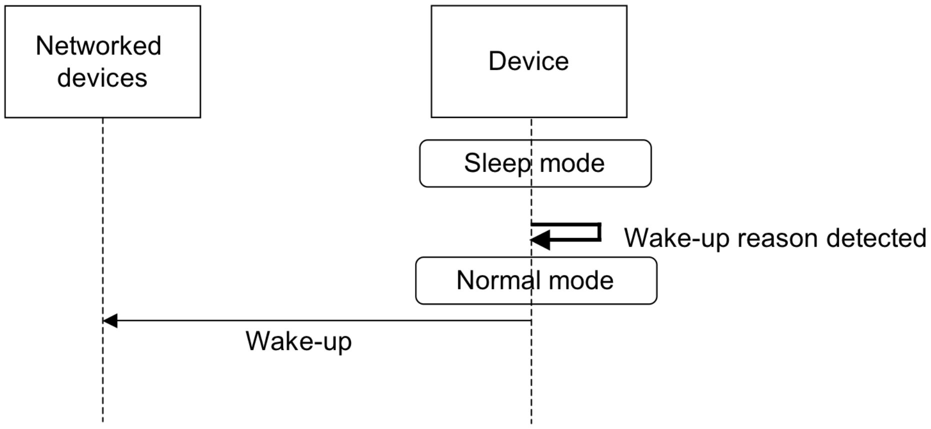

Figure 10 – Execution of "wake-up" service

A.2.3.4 Service "request sleep"

A slave may request the transition to sleep mode from the master. The master may react with execution of the "query sleep objection" service. Figure 11 shows the execution of the "request sleep" service.

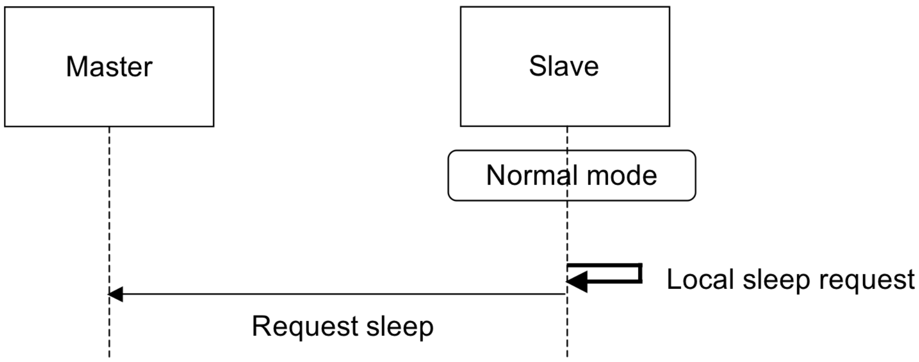

Figure 11 – Execution of "request sleep" service

A.2.4 Protocols

A.2.4.1 Protocol "query sleep objection"

The protocol as specified in Figure 12 shall be used to implement the "query sleep objection" service. L specifies the data length in bytes.

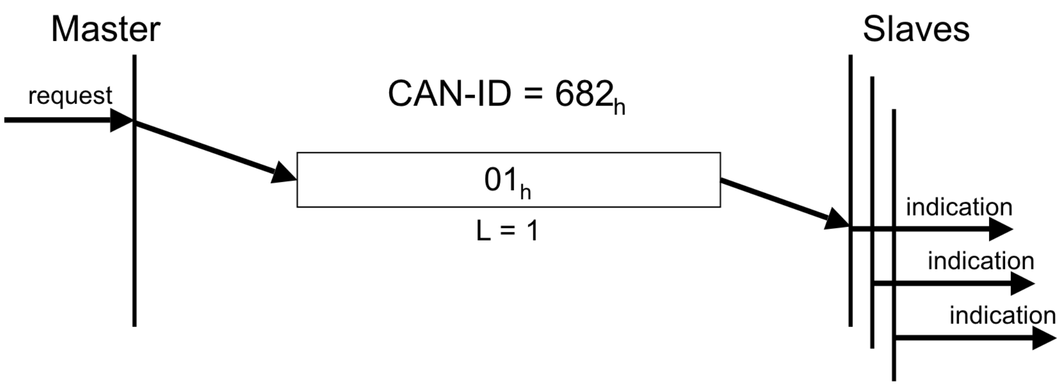

Figure 12 – Protocol "query sleep objection"

A.2.4.2 Protocol “sleep objection”

The protocol as specified in Figure 13 shall be used to implement the “sleep objection” service. L specifies the data length in bytes.

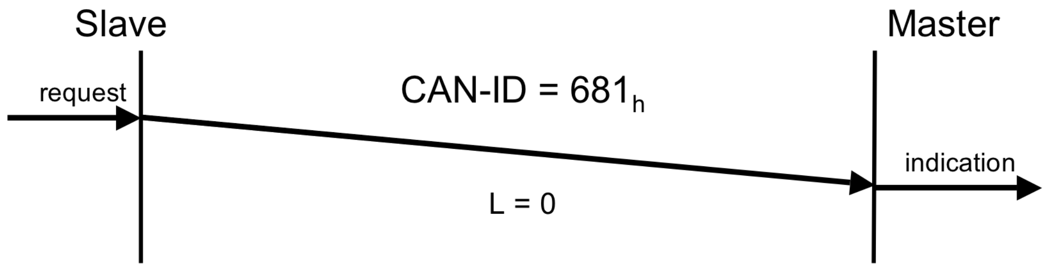

Figure 13 – Protocol “sleep objection”

A.2.4.3 Protocol “set sleep mode”

The protocol as specified in Figure 14 shall be used to implement the “set sleep mode” service. L specifies the data length in bytes.

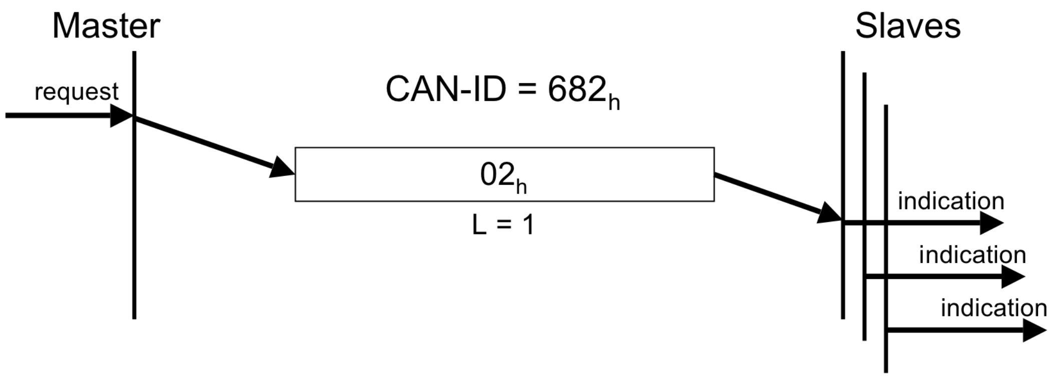

Figure 14 – Protocol “set sleep mode”

A.2.4.4 Protocol “wake-up”

The protocol as specified in Figure 15 shall be used to implement the “wake-up” service. L specifies the data length in bytes.

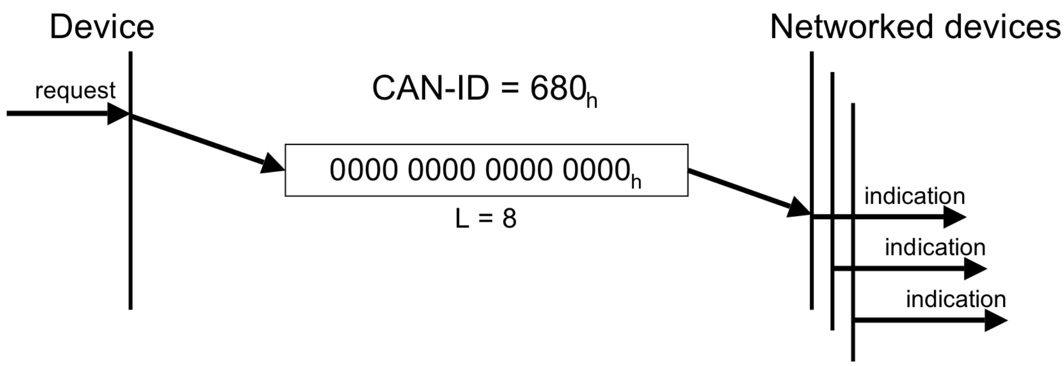

Figure 15 – Protocol “wake-up”

A.2.4.5 Protocol “request sleep”

The protocol as specified in Figure 16 shall be used to implement the “request sleep” service. L specifies the data length in bytes.

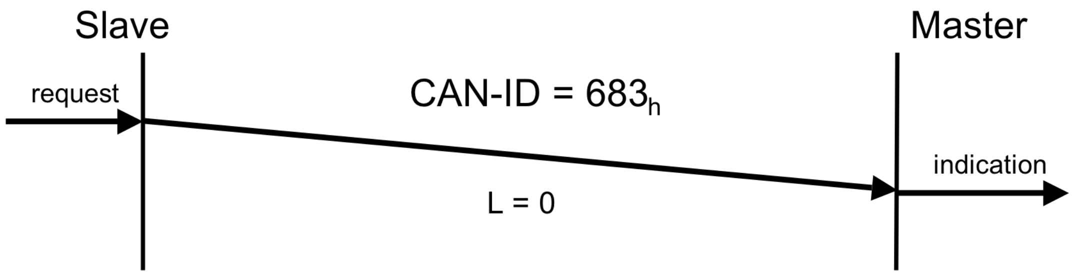

Figure 16 – Protocol “request sleep”

Annex B Timing information (normative)

B.1 Scope

This annex specifies the timely relationship of the protocols used in this specification. The time related information is specified in order to design an efficient system with predictable behaviour.

B.2 Protocol patterns

B.2.1 Introduction

The protocol patterns show the service primitives request (REQ), indication (IND), response (RES) and confirmation (CON) and their timely relationship.

B.2.2 Handshake pattern

This protocol pattern, as shown in Figure 6, is used for the following services:

- SDO (client/server): For segmented SDO protocol (see /CiA301/) the “next request/response” is used. For the expedited protocol the “next request window” and the “protocol time out” are not considered.
- LSS configure node-ID (LSS master/LSS slave)

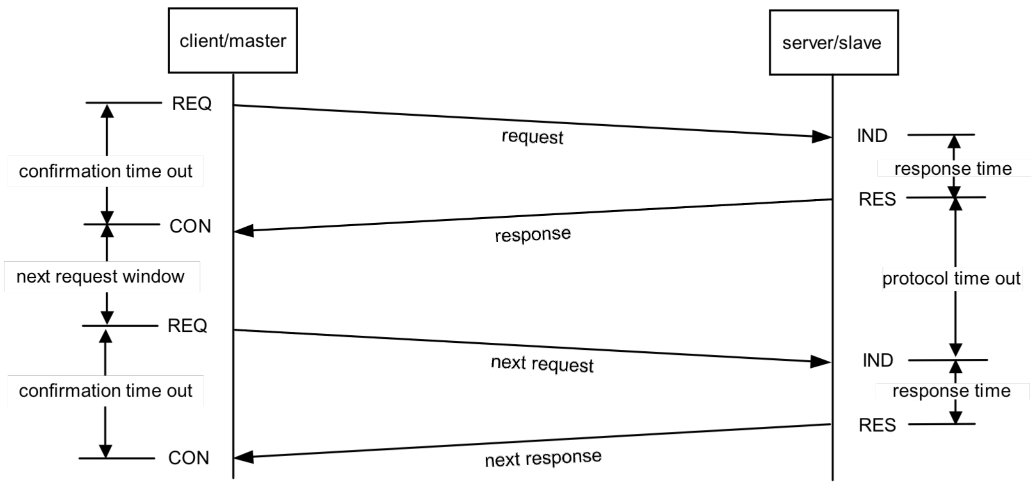

Figure 17 – Timing for handshake pattern

Table 11 lists the timing values, which shall be applied for handshake pattern.

Table 11 – Timing values for handshake pattern

Parameter name	Time value (in ms)
Confirmation time out (t_{cto})	100
Maximum response time (t_{mrt})	50
Maximum next request window (t_{mnrw})	50
Protocol time out (t_{pto})	100

B.2.3 Consecutive pattern

This protocol pattern, as shown in Figure 18, is used for the following services:

- LSS switch state selective (LSS master/LSS slave)

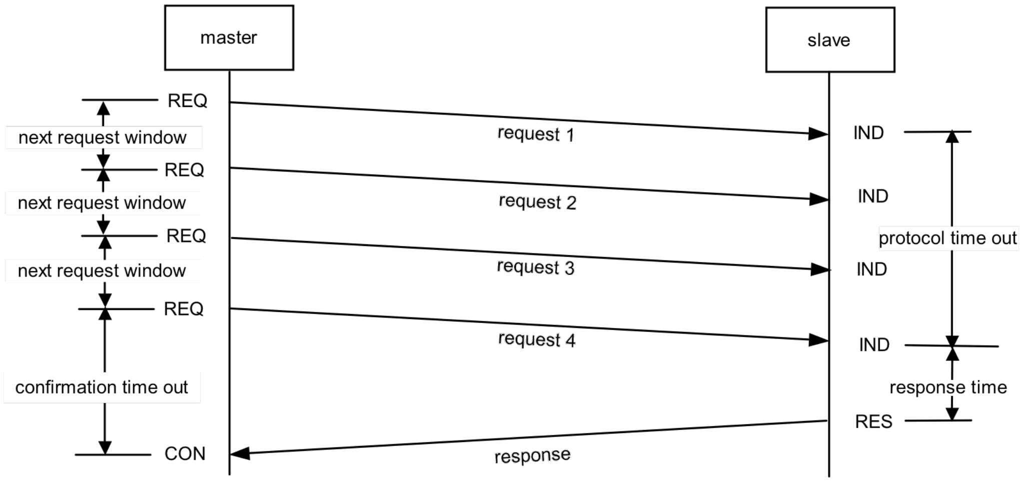

Figure 18 – Timing for consecutive pattern

Table 12 lists the timing values, which shall be applied for consecutive pattern.

Table 12 – Timing values for consecutive pattern

Parameter name	Time value (in ms)
Confirmation time out (t_{cto})	50
Maximum response time (t_{mrt})	25
Minimum next request window	10
Maximum next request window (t_{mnrw})	25
Protocol time out ((t_{pto}) - evaluation is started with every indication (IND))	50

B.3 Network management (NMT) timing

B.3.1 Introduction

This chapter defines the local boot-up time for a single device as well as the transition times in the NMT state machine. The specified timing values are maximum values, which shall not be exceeded.

B.3.2 Boot-up

Generally it is differentiated between boot-up caused by:

- Power on
- NMT reset (issued by the NMT master) applied to a device with a fix node-ID
- LSS switch mode global after setting the node-ID – this will cause an automatic transition to NMT state Pre-operational with the new valid node-ID

The boot-up message (see /CiA301/) only occurs if the node-ID of the NMT slave is valid. The boot-up message indicates that a device has entered the NMT state Pre-operational. Prior to this an SDO request is not responded.

All devices with a cold boot time larger than 1200 ms shall support node-ID distribution via LSS.

Table 13 specifies the boot-up timing.

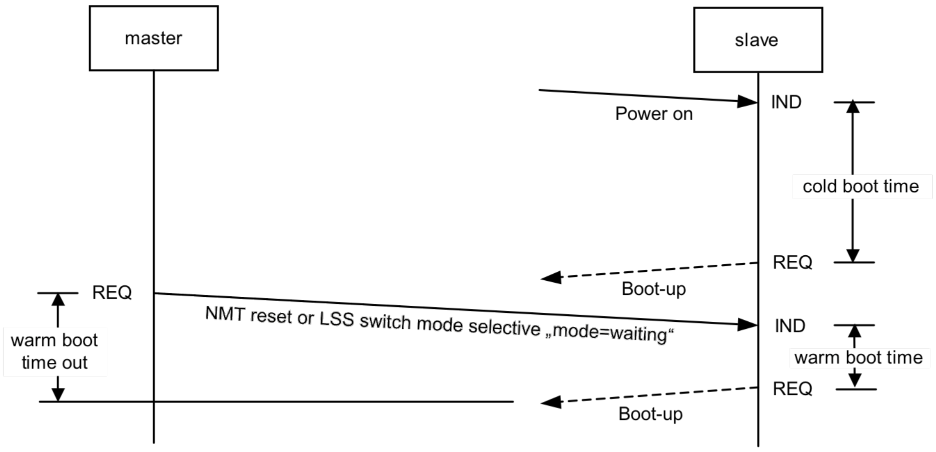

Figure 19 – Timing for boot-up

Table 13 lists the timing values, which shall be applied during boot-up.

Table 13 – Timing values for boot-up

Parameter name	Time value (in ms)
Cold boot time (t_{cbt}) – within this time a device has either transmitted the boot-up message or is ready for the LSS FastScan after a power on.	1200
Warm boot time (t_{wbt}) – within this time a device has either transmitted the boot-up message or is ready for the LSS FastScan after a NMT reset command.	250
Warm boot time out (t_{wbto}) – after this time the NMT master assumes that either all ECUs have sent their boot-up messages or that the LSS is activated for FastScan.	300

B.3.3 NMT state transitions

NMT state transitions (Pre-operational to Operational and vice versa) caused by the NMT message require a time section. This may be detected via heartbeat message and PDO of this device.

A heartbeat message created during the state transition time is allowed to carry the previous NMT state.

Figure 20 specifies the NMT state transition timing on an example for NMT state transition from Pre-operational to Operational.

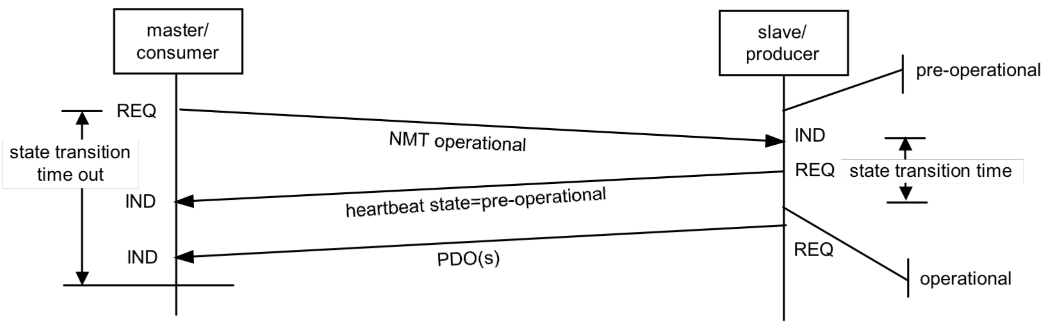

Figure 20 – Timing for NMT state transition from Pre-operational to Operational

Table 14 lists the timing values, which shall be applied for NMT state transitions.

Table 14 – Timing values for NMT state transitions

Parameter name	Time value (in ms)
State transition time (t_{stt}) – within this time the device shall execute the transition.	10
State transition time out (t_{stto}) – after this time the state transition has to be completed. If the transition is from Pre-operational to Operational the PDO(s) has/have to be transmitted.	20

B.4 LSS timing

B.4.1 Introduction

The dynamic assignment of node-IDs relies on the FastScan mechanism (see /CiA305/). This mechanism utilizes a cyclic request of information in combination with a time-out. It is important that every cycle is completed correctly. Always the slowest device will determine the pace of the mechanism until it finally gets its node-ID.

B.4.2 LSS identify non-configured remote slave

On system boot-up this service is executed prior to the FastScan protocol in order to make sure that there are devices with non-configured node-IDs in the network. The FastScan is only executed if there are devices with non-configured node-IDs detected. During normal operation time this service is scheduled every 1000 ms. Figure 21 specifies the identification sequence timing.

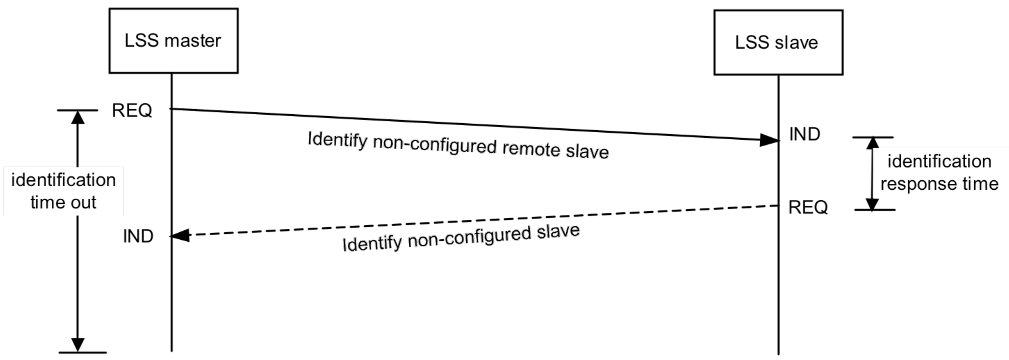

Figure 21 – Identification sequence timing

Table 15 lists the timing values, which shall be applied for the identification sequence.

Table 15 – Timing values for identification sequence

Parameter name	Time value (in ms)
Identification response time (t_{irt}) – within this time the device shall transmit the identification.	25
Identification time out (t_{ito}) – after this time it is assumed that there is no non-configured device in the network.	1000

If the LSS master detects the service “identify non-configured slave” it may immediately start the FastScan service.

B.4.3 LSS FastScan service

The FastScan service relies on a fast reply from the LSS slave(s) within the same cycle. If a response misses, the corresponding cycle of the FastScan is not successful. This means that the last FastScan request will not be responded by any LSS slave. In this case the FastScan service is repeated with the minimum FastScan cycle time incremented by $t_{cycleinc}$. If the service fails again the next increment follows until the maximum FastScan cycle time is reached. Figure 22 specifies the FastScan timing.

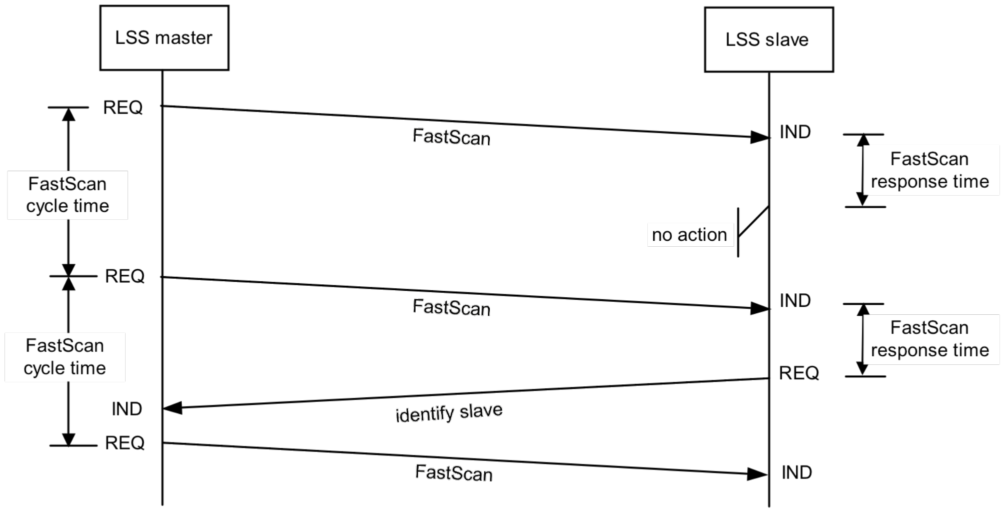

Figure 22 – FastScan timing

Table 16 lists the timing values, which shall be applied for the FastScan.

Table 16 – Timing values for FastScan

Parameter name	Time value (in ms)
FastScan response time (t_{fsrt}) – within this time the device shall handle the request and execute the service "identify slave".	10
FastScan cycle time ($t_{fscycle}$) – the FastScan message is transmitted with this cycle time. The value *n* starts with 0 and is incremented by 1 with every failed FastScan. If a FastScan was successful (LSS slave could be identified) the *n* is reset to 0. The maximum value is *n* = 8 (resulting in a maximum cycle time of 100 milliseconds).	20 + (n * $t_{cycleinc}$)
Cycle increment value ($t_{cycleinc}$) – if the FastScan fails the cycle time is incremented by this value.	10

B.5 Power management timing

B.5.1 Sleep/wake-up

For sleep and wake-up two timing diagrams are necessary. Executing the service "query sleep objection" the LSS master waits for a pre-defined time t_{oto} (objection time out) if there are any objections in the network. Figure 23 specifies the timing for query sleep objection.

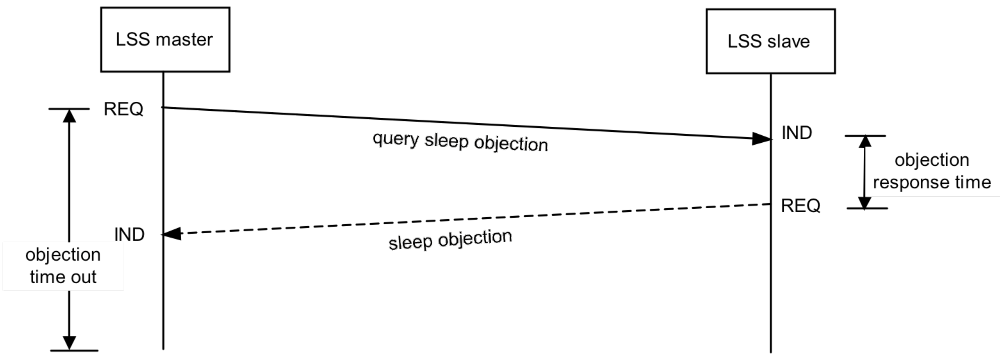

Figure 23 – Query sleep objection protocol timing

Table 17 lists the timing values, which shall be applied for the query sleep objection.

Table 17 – Timing values for query sleep objection

Parameter name	Time value (in ms)
Objection response time (t_{ort}) – within this time the ECU shall transmit the objection.	50
Objection time out (t_{oto}) – after this time it is assumed that there is no objection. The LSS master may now invoke a “set sleep mode” request	1000

When the “set sleep mode” request is received the receiver enters the prepare sleep mode. In this mode all messages are ignored and there are no further messages transmitted. It stays in this mode for a pre-defined time t_{swt} (sleep wait time) and finally enters “sleep mode” automatically. Table 18 specifies the timing values for sleep wait time.

Table 18 – Timing values for sleep wait time

Parameter name	Time value (in ms)
Sleep wait time (t_{swt}) – the device stays in prepare sleep mode for this time.	1000

B.6 Miscellaneous timing values

All other timing values are specified in Table 19.

Table 19 – Miscellaneous timing values

Parameter name	Time value (in ms)
Minimum boot-up time (see chapter B.3.2)	1200
Wake-up repetition time t_{wurpt} (see chapter A.2.3.3)	60000

