CiA® 417

Application profile for lift control systems

Part 1: General definitions

Version: 2.0.0
02 February 2011

HISTORY

Date	Changes
2002-07-15	*Publication of version 1.0* as draft standard proposal
2010-02-01	*Publication of version 2.0* as draft standard proposal NOTE: Version 2.0 is partly incompatible to version 1.0
2011-02-02	*Publication of version 2.0.0* as public specification NOTE: This document has been converted into “docx format”. The conversion caused minor layout differences to the predecessor document in “doc format”. The technical content word-by-word is the very same.

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.
Kontumazgarten 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

This set of CANopen application profile specifications describes the *CANopen Lift* control network system. It specifies the CANopen communication interfaces and the application functionality of several functional elements (virtual devices).

This application profile specification consists of several parts:

- Part 1 provides general definitions
- Part 2 specifies the functionality of the virtual devices
- Part 3 specifies the pre-defined PDOs
- Part 4 specifies the application objects

Besides some general definitions such as general virtual device descriptions, this part specifies the CAN physical layer as well as the error handling. Additionally some network architecture examples are given.

2 Normative references

/CiA301/	CiA 301, CANopen application layer and communication profile
/CiA302/	CiA 302 (all parts), Additional CANopen application layer functions
/CiA305/	CiA 305, Layer setting services (LSS)
/CiA303-1/	CiA 303-1, Cabling and connector pin assignment
/CiA303-2/	CiA 303-2, Representation of SI units and prefixes
/ISO8859-15/	ISO/IEC 8859-15, Information technology – 8-bit single-byte coded graphic character sets – Part 15: Latin alphabet No. 9
/ISO11898-2/	ISO 11898-2, Road vehicles – Controller area network (CAN) – Part 2: High-speed medium access unit

3 Definitions, acronyms, and abbreviations

3.1 General

The definitions, acronyms, and abbreviation given in /CiA301/ apply for this specification, too.

3.2 Acronyms

CAN	Controller area network
COB	Communication object
COB-ID	COB identifier
CSDO	Client SDO
LSB	Least significant bit
LSS	Layer setting services
MSB	Most significant bit
SSDO	Server SDO
VD	Virtual device

3.3 Abbreviations

Acc.	Access
Cat.	Category
const	constant

ro read-only

rw read/write

4 General architecture

4.1 Overview

This application profile specification describes the virtual devices of lift control systems. The virtual controllers (e.g. call, car door, and car drive controller) perform dedicated control functions of the lift application. In a lift application, all controller functions may be implemented in one CANopen device. In other applications, the controller functions are implemented in different CANopen devices. The virtual units (e.g. input and output panels, car door, light barrier, car position, car drive, load-measuring) are implemented each in single CANopen devices or combined in one or more CANopen devices. This flexible implementation options allow the use of this application profile in simple as well as sophisticated lift applications.

The virtual interfaces are implemented as CANopen interfaces or as CANopen device internal interfaces, if the virtual devices reside in the same CANopen device. If the virtual interfaces between virtual devices are implemented as CANopen interfaces they use SDO or PDO services to read or write application objects.

Each virtual device supports a set of dedicated mandatory and optional application objects. Most of the application objects are mapped into pre-defined PDOs. If an implemented application object is not mapped into one of the pre-defined PDOs, other CANopen devices may access them by means of SDO. The CSDOs, which corresponds to the Default SSDO, shall be implemented always in the call controller. CANopen devices compliant to this application profile without call controller functionality shall not implement any CSDO that relates to Default SSDO. It is recommended to implement in the call controller an SDO manager compliant to /CiA302-5/, if an external tool for configuration or trouble-shooting purposes is used.

4.2 Virtual device descriptions

4.2.1 General

Every virtual device represents specific functional elements. The following brief descriptions give an overview on the functionality of the different virtual devices. The supported application objects and PDOs are summarized in part 2 of this application profile. The detailed PDO interfaces are specified in part 3 of this application profile. The detailed application objects are specified in part 4 of this application profile.

4.2.2 Call controller

The VD call controller receives all call requests from the VD input panels, and transmits the corresponding acknowledgements to the VD output panels. In addition, the call controller sends commands to the VD car drive controller to move the car and the VD car door controller to open and close the doors. When the call controller, the car drive controller, and the car door controller are implemented on the same CANopen device, the communication between these VDs is handled locally.

The call controller shall provide NMT master capability. If several CANopen devices with call controller functionality are installed in one CANopen network, it is necessary that all of them support the NMT “flying” master function.

4.2.3 Input panel unit

The VD input panel unit is installed as in-car call panel or as floor call panel or as general input device (e.g. key-switch or fire-alarm). The input panel transmits user requests to the VD call controller including access requests.

4.2.4 Output panel unit

The VD output panel unit is installed as in-car display panel or as floor display panel or as general output device (e.g. announcement unit). The output panel is a display device that shows car position and/or car moving direction. Additionally, it announces acoustically the incoming car. It also receives the acknowledgements for the call requests.

4.2.5 Car door controller

The VD car door controller transmits commands (e.g. open and close) to the VD car door unit and receives status information from the VD car door unit and the VD light barrier unit.

4.2.6 Car door unit

The VD car door unit opens and closes the car door(s). It receives the commands from the VD car door controller and provides its status to the VD car door controller. Theoretically there may be four doors installed in each car. However, only for three doors are TPDOs pre-defined that contain status information.

4.2.7 Light barrier unit

The VD light barrier unit detects subjects and objects entering the protected area of the car door unit and sends this information to the VD car door controller.

4.2.8 Car drive controller

The VD car drive controller transmits commands to the VD car drive unit, on request from the VD call controller. It receives status information from the VD car drive unit and the VD load-measuring unit. If the profile position mode is used, the VD car drive controller needs additionally status information from the VD car position unit.

4.2.9 Car drive unit

The VD car drive unit moves the car upwards and downwards. It receives the motion commands from the VD car drive controller. It is based on the CANopen profile for drives and motion control as specified in /CiA402-2/.

NOTE There are some additional objects necessary for lift applications that are not specified in /CiA402-2/). If there is no absolute encoder available, the target velocity (6430_h) shall be provided to the car drive unit using the Profile Velocity Mode; if there is an absolute encoder available, the target position (6420_h) shall be provided to the car drive unit using the Profile Position Mode.

The operation mode is selected by the modes of operation (6403_h). In case of velocity-controlled drives the Profile Velocity Mode shall be used. The objects for the velocity profile are stored in the drive unit and may be configured by the drive controller. Due to safety reasons, configuration shall not be possible in Operation Enable state of the VD car drive unit.

The car drive unit state machine is controlled by the controlword (6400_h). Drive-specific functions such as motor relays are operated locally in the drive unit. Motion is determined by a target velocity unequal 0. Direction is indicated by the sign of target velocity; positive values shall indicate upward motion of the car. Sense of rotation depends on mounting position.

Depending on the given target velocity and the velocity profile curve parameters, the drive unit calculates the control effort (6406_h). Reaching the target floor-switch the controller shall give the end velocity (6424_h) as new target velocity. Giving a target velocity of 0 shall terminate the drive. The drive unit shall indicate reaching the target velocity in the 10^{th} bit of the statusword (6401_h).

In case of position-controlled drives the Profile Position Mode shall be used. To configure the position profile curve the same parameters as for the velocity profile curve are used. After setting a new position, the drive unit calculates the curve and starts motion. During motion the drive controller may change target position. If the control_effort allows stopping at the new target position, this shall be indicated in the 12^{th} bit of the statusword. If the drive cannot stop at the new target position, the drive unit shall move to the previous target position. Reaching a target position shall be indicated in the 10^{th} bit of the statusword.

4.2.10 Car position unit

The VD car position unit measures the current position of the car and provides additionally values like speed and acceleration. It is very similar to the function of an encoder device. Therefore it is based on the CANopen profile for encoders as specified in CiA 406. Up to four car position units may be used for each lift application.

4.2.11 Load-measuring unit

The VD load-measuring unit measures the current load of the car and signals certain situations like zero load, full load (no additional calls will be served), and overload (motion is not allowed).

4.2.12 Remote data transmission unit

The VD remote data transmission unit provides gateway functionality for remote control and diagnostics purposes. It uses normally SDO clients to access the other CANopen Lift devices.

4.3 Single- and multiple-shaft lift control systems

In case of single-shaft lift control system, it is possible to connect all necessary devices in one single CANopen network (see example in Figure 1). It is also possible to use a parallel dual CANopen network architecture with a centralized controller acting as a PDO bridge (see example in Figure 2). Another option is to cascade CANopen networks (see example in Figure 3); the network connecting devices may provide SDO router and/or PDO bridge functionality (PDO mapping is not changed, CAN-IDs are subject of change). In addition, it may provide PDO gateway functionality (PDO mapping is changed).

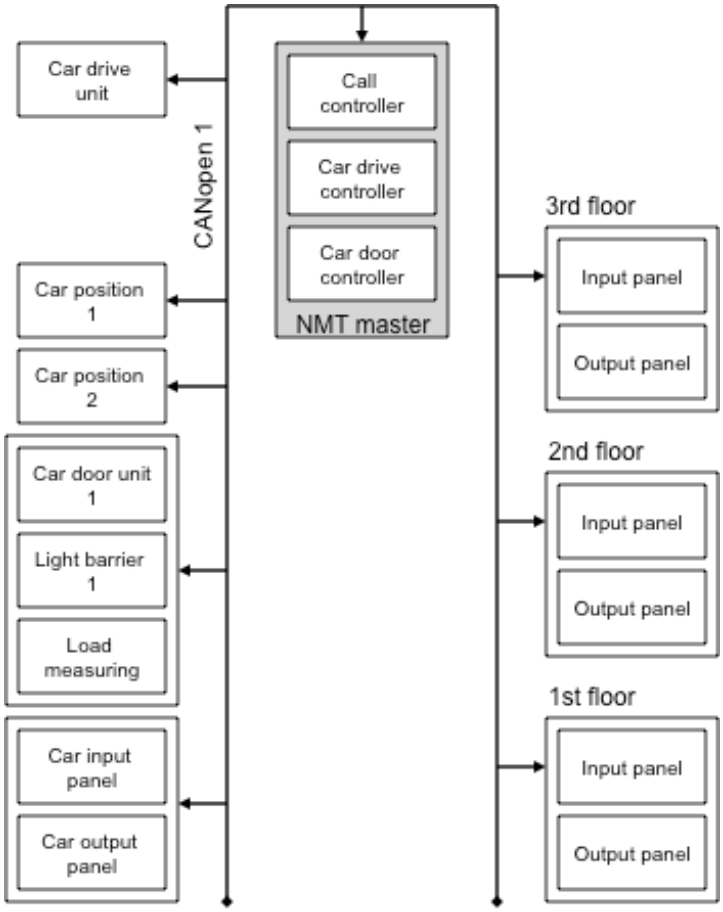

Figure 1 – Single network architecture for a single-shaft lift control system

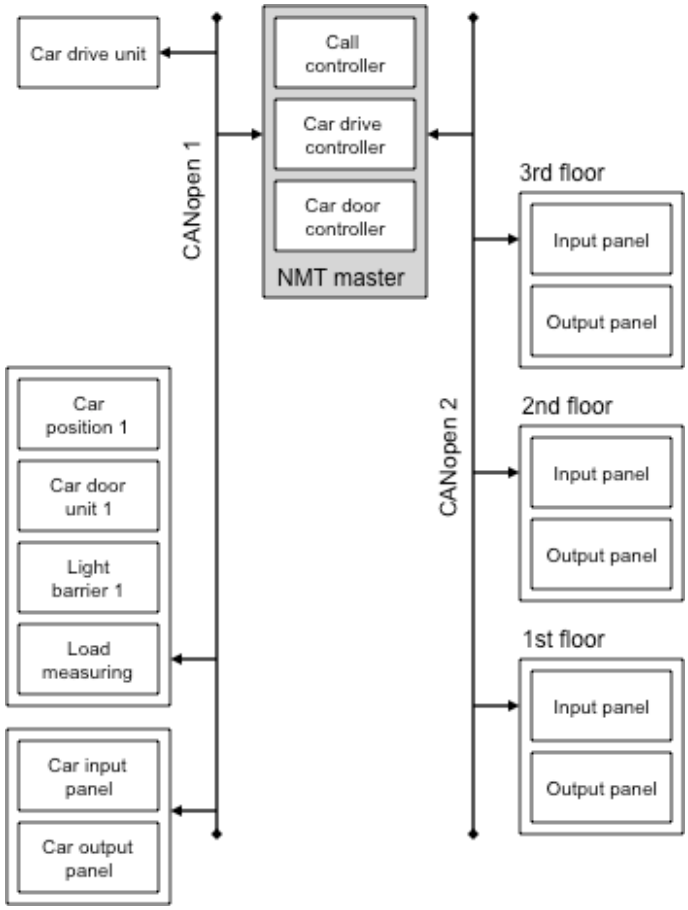

Figure 2 – Parallel network architecture for a single-shaft lift control system

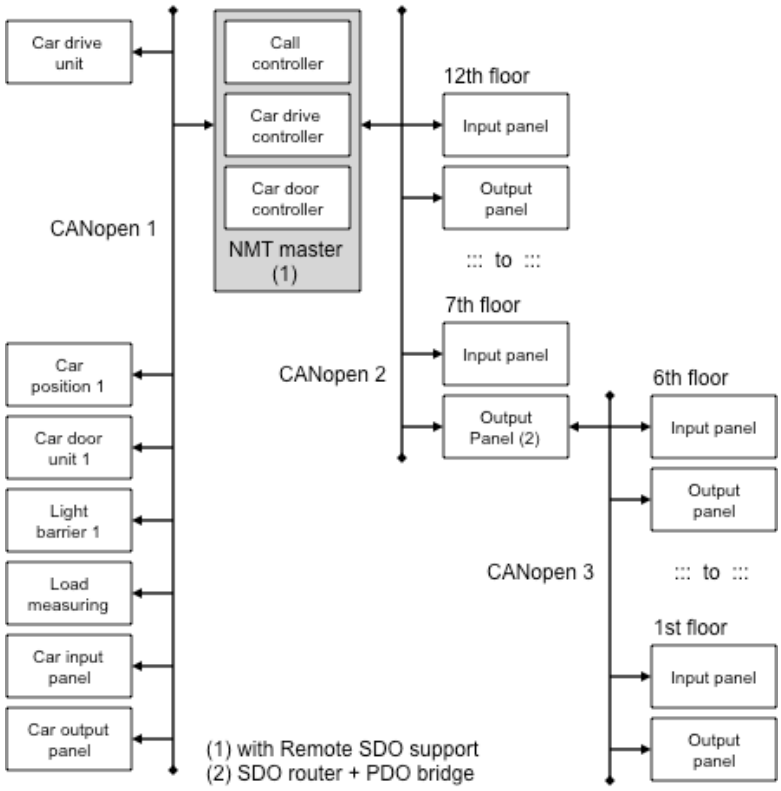

Figure 3 – Parallel network with cascaded network for a single-shaft lift control system

Theoretically it is possible to use for a multi-shaft lift control system just one CANopen network (see example in Figure 4). Due to busload and network length requirements, those systems are normally realized by means of several CANopen network segments (see example in Figure 5). If PDO gateways are used, all single-shaft lift control networks may be implemented as lift application 1 (see example in Figure 6).

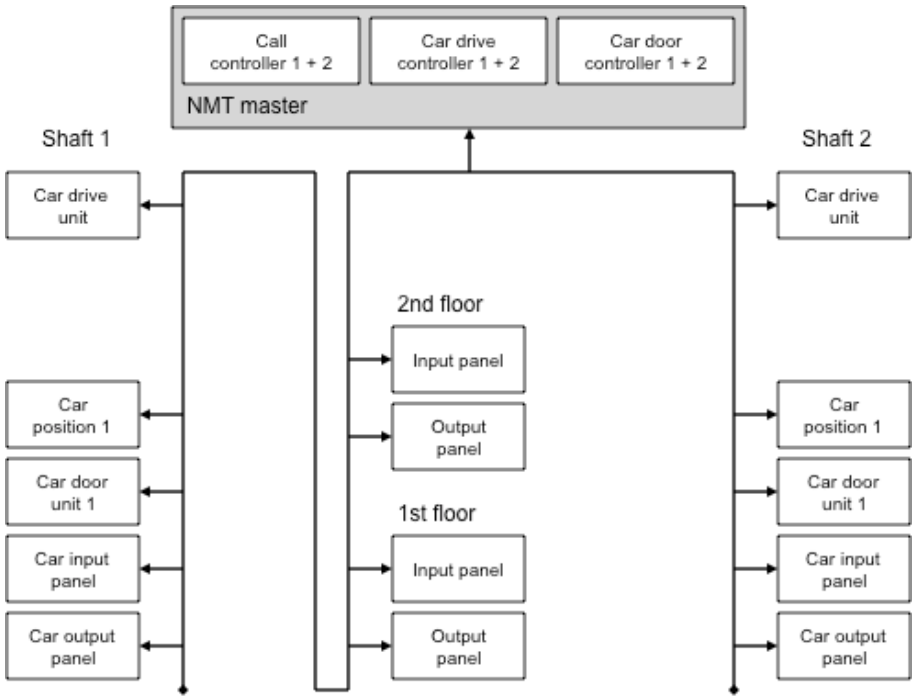

Figure 4 – Single network for a dual-shaft lift control system

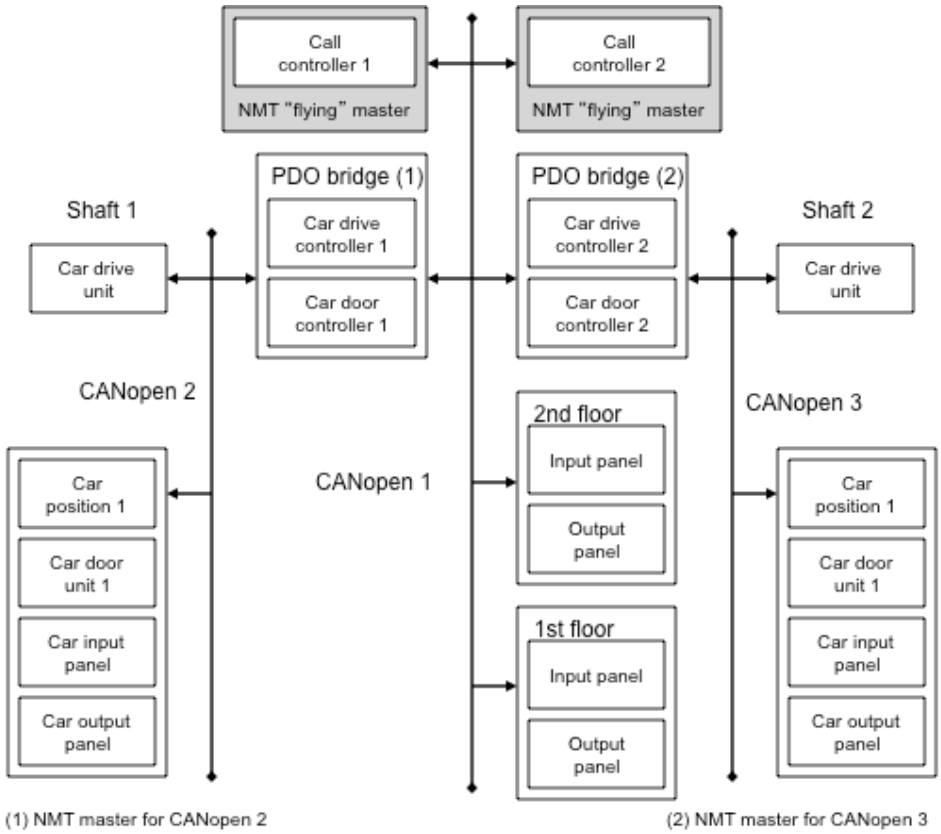

Figure 5 – Three-network architecture for a dual-shaft lift control system

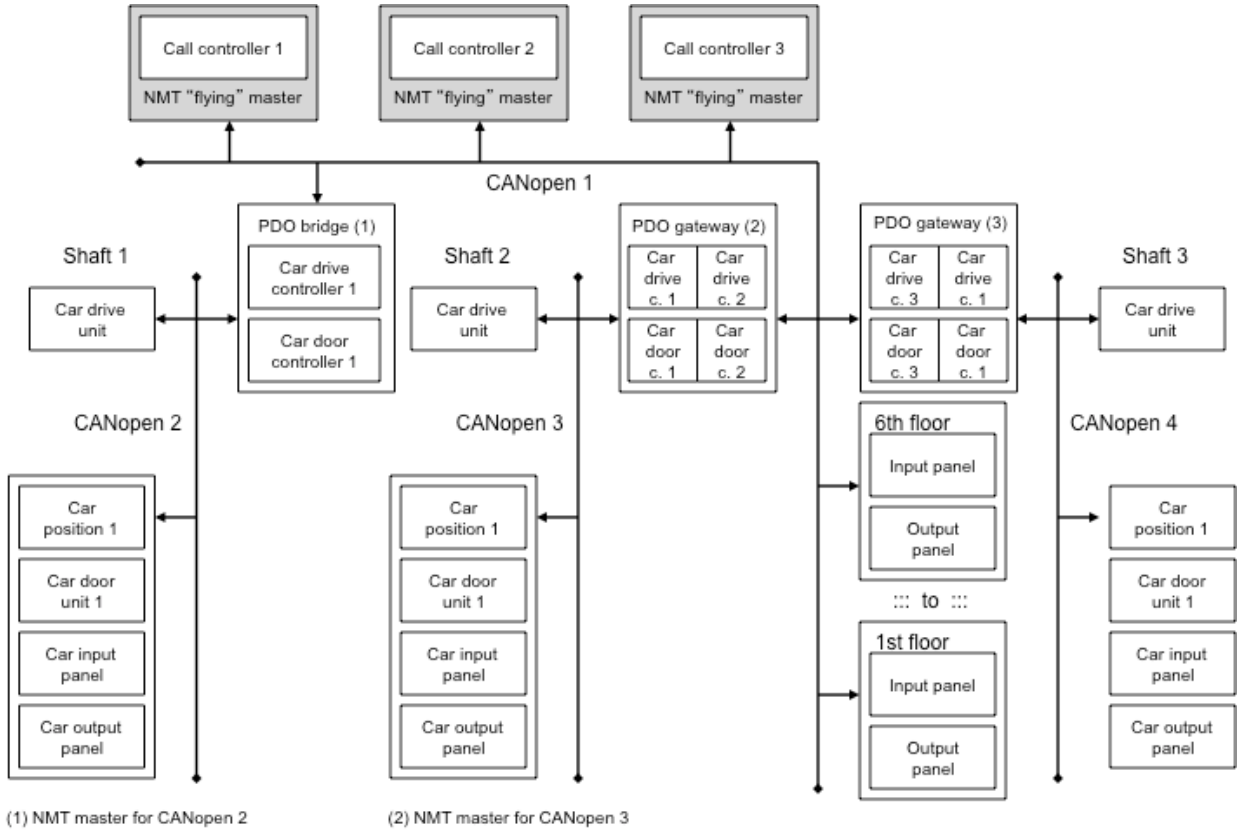

Figure 6 – Multiple-shaft lift control system with PDO bridges

5 Physical layer

5.1 General

CANopen devices conformant to this application profile shall implement /ISO11898-2/ compliant transceiver chips capable to drive in minimum 64 nodes.

5.2 Bit rate

Automatic bit rate detection may be supported. The default bit rate shall be 250 kbit/s, the bit rate of 125 kbit/s shall be supported, too. Other bit rates as defined in /CiA301/ may be supported. The bit timing shall be as defined in /CiA301/.

The setting method of the bit rate is manufacturer-specific. It is not recommended to set the bit time by means of SDO write access.

5.3 Bus topology

No specific bus topology is specified. The network may be segmented and/or cascaded.

5.4 Bus cable

No specific bus cable is specified. It is recommended to consider the cabling recommendations as given in /CiA303-1/. For network segments connecting only panel and display devices patch cables of category 5 are recommended.

5.5 Bus connector

No specific bus connector is specified. It is recommended to use 9-pin D-sub, RJ45, or open-style connector. If other connectors are used, it is recommended to apply the pin assignments as given in /CiA303-1/.

6 Node-ID assignment

6.1 General

The node-ID assignment method is not in the scope of this specification. However, it is not recommended to set the node-ID by means of SDO write access.

In order to achieve an off-the-shelf plug-and-play capability, it is recommended to implement in each CANopen device LSS slave services as defined in /CiA305/. If using LSS, it is recommended storing the assigned node-ID in non-volatile memory.

6.2 Recommendation for shaft lift control networks

Table 1 shows the recommendation for the shaft lift network node-ID assignment. If a CANopen device implements several VDs, it is recommended to assign the lowest node-ID.

Table 1 – Node-ID assignment for shaft lift control networks

Node-ID	Virtual device
1	Call controller (NMT master)
2	Car drive controller
3	Car door controller
4	Car position unit 1
5	Car position unit 2
6	Car drive unit
7	Car door unit 1
8	Car door unit 2
9	Car door unit 3
10	Remote data transmission unit
11 *to* 12	reserved
13	Load-measuring unit
14 *to* 15	reserved
16	Car I/O panel unit 1
17	Car I/O panel unit 2
18	Car I/O panel unit 3
19	Car I/O panel unit 4
20	Car I/O panel unit 5
21 *to* 85	Floor I/O panel unit 1 *to* 64
86 *to* 119	free
120 *to* 124	reserved
125	used as floor I/O panel default node-ID (NOTE)
126	used for boot-loader (NOTE)
127	used for CANopen tool (NOTE)
NOTE These node-IDs are not recommended for CANopen lift devices in NMT operational state.	

6.3 Recommendations for the multiple-shaft controller network

Table 2 shows the recommendation for the multiple-shaft controller network node-ID assignment. If a CANopen device implements several VDs, it is recommended to assign the lowest node-ID.

Table 2 – Node-ID assignment for the multiple-shaft controller network

Node-ID	Virtual device
1	Call/car drive/door controller lift 1 (NMT “flying” master)
2	Call/car drive/door controller lift 2 (NMT “flying” master)
3	Call/car drive/door controller lift 3 (NMT “flying” master)
4	Call/car drive/door controller lift 4 (NMT “flying” master)
5	Call/car drive/door controller lift 5 (NMT “flying” master)
6	Call/car drive/door controller lift 6 (NMT “flying” master)
7	Call/car drive/door controller lift 7 (NMT “flying” master)
8	Call/car drive/door controller lift 8 (NMT “flying” master)
9	Call controller for all lifts (NMT master)
10	Data remote transmission unit lift 1
11	Data remote transmission unit lift 2
12	Data remote transmission unit lift 3
13	Data remote transmission unit lift 4
14	Data remote transmission unit lift 5
15	Data remote transmission unit lift 6
16	Data remote transmission unit lift 7
17	Data remote transmission unit lift 8
18 *to* 20	reserved
21 *to* 85	Floor I/O panel unit 1 *to* 64
86 *to* 119	free
120 *to* 124	reserved
125	used as floor I/O panel default node-ID (NOTE 2)
126	used for boot-loader (NOTE 2)
127	used for CANopen tool (NOTE 2)
NOTE 1 If only one call controller is installed. NOTE 2 These node-IDs are not recommended for CANopen lift devices in NMT operational state.	

7 Error handling

7.1 Principle

Emergency messages are triggered by internal errors in the physical device and they are assigned to high priority to ensure that they get access to the bus without latency. By default, the Emergency messages contain the error field with pre-defined error numbers and additional information. For further definitions see /CiA301/.

7.2 Error behavior

If a serious device failure is detected the physical device shall enter by default autonomously the NMT pre-operational state. If object 1029_h is implemented, the physical device can be configured to enter alternatively the NMT stopped state or remain in the current NMT state in case of a device failure. Device failures shall include the following communication errors:

- Bus-off conditions of the CANopen interface
- Life guarding event with the state ‘occurred’
- Heartbeat event with state ‘occurred’

Severe device errors are also caused by device internal failures.

7.3 Emergency messages

Devices compliant to this application profile shall support Emergency messages. An Emergency message with appropriate emergency error code shall be transmitted when the CAN controller chip enters *error passive state* or recovers from bus-off state. Additionally, an emergency message may be transmitted when reaching the warning level of the CAN controller chip.

7.4 Additional emergency error code meanings

Devices compliant to this specification may use additional error codes, as specified in Table 3.

If the error code is $FF04_h$, the additional bytes of the EMCY shall not be used for manufacturer-specific purposes. The use of these bytes is reserved.

Table 3 – Error codes

Error code	Description
0010_h	CAN warning level
3211_h	Over voltage (device internal)
3221_h	Under voltage (devices internal)
$FF01_h$	Light barrier defect
$FF02_h$	Finger protector defect
$FF03_h$	Motion detection defect
$FF04_h$	Application error

8 Character encoding

8.1 General

As the service menus of existing devices are different, no interface for the parameters is specified. However, a standardized way of transmitting and receiving characters for a text-based menu is recommended.

8.2 Encoding of special characters

Latin characters, including German umlauts, shall be encoded with /ISO8859-15/.

8.3 Encodings codes to control the service menu (input)

The codes compliant to /ISO8859-15/ specified in Table 4 shall be used to control the service menu.

Table 4 – Control codes

Dec-code	Hex-code	Character	Description
24	18_h	CAN	End / cancel (Ctrl-X)
45	$2D_h$	'-'	Decrease
43	$2B_h$	'+'	Increase
13	$0D_h$	CR	OK
27 65	$1B_h$ 41_h	ESC A	Cursor up key
27 66	$1B_h$ 42_h	ESC B	Cursor down key
27 67	$1B_h$ 43_h	ESC C	Cursor right key
27 68	$1B_h$ 44_h	ESC D	Cursor left key
27 80	$1B_h$ 50_h	ESC P	F1 key
27 81	$1B_h$ 51_h	ESC Q	F2 key

Dec-code	Hex-code	Character	Description
27 82	$1B_h$ 52_h	ESC R	F3 key
27 83	$1B_h$ 53_h	ESC S	F4 key

8.4 Requirements on VT52 escape sequences (output)

The escape sequences as specified in Table 5 shall be supported.

Table 5 – escape sequences

Code	Name	Description
ESC A	Cursor up	Move up cursor one line, if not already in first line
ESC B	Cursor down	Move down cursor, if not in last line
ESC C	Cursor right	Move cursor to the right if not already at maximum right
ESC D	Cursor left	Move cursor to the left if not already at the maximum left
ESC E	Clear home	Screen is cleared, cursor moves to home position
ESC H	Cursor home	Cursor moves to home position
ESC I	Cursor up and insert	Move up cursor, if already in first line, a new line is inserted
ESC J	Clear to end of frame	Screen is cleared from the position of the cursor to the bottom
ESC K	Clear to end of line	Delete line from cursor position
ESC L	Insert line	Insert new line at cursor position
ESC M	Delete line	Delete line at cursor position, the lower lines move up
ESC Yyx	Move cursor	Moves cursor to column x and row y; a value of 32 is always added to x and y
ESC e	Cursor on	Show cursor
ESC f	Cursor off	Hide cursor
ESC j	Store cursor	Save current cursor position
ESC k	Restore cursor	Go to saved position
ESC l	Clear line	The current line is deleted, the cursor moves to the start of line
ESC o	Clear line to cursor	The current line is deleted to the cursor position

