

INTERNATIONAL STANDARD

NORME INTERNATIONALE

**Electronic railway equipment – Train communication network (TCN) –
Part 3-3: CANopen Consist Network (CCN)**

**Matériel électronique ferroviaire – Réseau embarqué de train (TCN) –
Partie 3-3: Réseau de rame CANopen (CCN)**



THIS PUBLICATION IS COPYRIGHT PROTECTED

Copyright © 2012 IEC, Geneva, Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either IEC or IEC's member National Committee in the country of the requester.

If you have any questions about IEC copyright or have an enquiry about obtaining additional rights to this publication, please contact the address below or your local IEC member National Committee for further information.

Droits de reproduction réservés. Sauf indication contraire, aucune partie de cette publication ne peut être reproduite ni utilisée sous quelque forme que ce soit et par aucun procédé, électronique ou mécanique, y compris la photocopie et les microfilms, sans l'accord écrit de la CEI ou du Comité national de la CEI du pays du demandeur.

Si vous avez des questions sur le copyright de la CEI ou si vous désirez obtenir des droits supplémentaires sur cette publication, utilisez les coordonnées ci-après ou contactez le Comité national de la CEI de votre pays de résidence.

IEC Central Office
3, rue de Varembé
CH-1211 Geneva 20
Switzerland

Tel.: +41 22 919 02 11
Fax: +41 22 919 03 00
info@iec.ch
www.iec.ch

About the IEC

The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications

The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

Useful links:

IEC publications search - www.iec.ch/searchpub

The advanced search enables you to find IEC publications by a variety of criteria (reference number, text, technical committee,...).

It also gives information on projects, replaced and withdrawn publications.

IEC Just Published - webstore.iec.ch/justpublished

Stay up to date on all new IEC publications. Just Published details all new publications released. Available on-line and also once a month by email.

Electropedia - www.electropedia.org

The world's leading online dictionary of electronic and electrical terms containing more than 30 000 terms and definitions in English and French, with equivalent terms in additional languages. Also known as the International Electrotechnical Vocabulary (IEV) on-line.

Customer Service Centre - webstore.iec.ch/csc

If you wish to give us your feedback on this publication or need further assistance, please contact the Customer Service Centre: csc@iec.ch.

A propos de la CEI

La Commission Electrotechnique Internationale (CEI) est la première organisation mondiale qui élabore et publie des Normes internationales pour tout ce qui a trait à l'électricité, à l'électronique et aux technologies apparentées.

A propos des publications CEI

Le contenu technique des publications de la CEI est constamment revu. Veuillez vous assurer que vous possédez l'édition la plus récente, un corrigendum ou amendement peut avoir été publié.

Liens utiles:

Recherche de publications CEI - www.iec.ch/searchpub

La recherche avancée vous permet de trouver des publications CEI en utilisant différents critères (numéro de référence, texte, comité d'études,...).

Elle donne aussi des informations sur les projets et les publications remplacées ou retirées.

Just Published CEI - webstore.iec.ch/justpublished

Restez informé sur les nouvelles publications de la CEI. Just Published détaille les nouvelles publications parues. Disponible en ligne et aussi une fois par mois par email.

Electropedia - www.electropedia.org

Le premier dictionnaire en ligne au monde de termes électroniques et électriques. Il contient plus de 30 000 termes et définitions en anglais et en français, ainsi que les termes équivalents dans les langues additionnelles. Egalement appelé Vocabulaire Electrotechnique International (VEI) en ligne.

Service Clients - webstore.iec.ch/csc

Si vous désirez nous donner des commentaires sur cette publication ou si vous avez des questions contactez-nous: csc@iec.ch.

INTERNATIONAL STANDARD

NORME INTERNATIONALE

**Electronic railway equipment – Train communication network (TCN) –
Part 3-3: CANopen Consist Network (CCN)**

**Matériel électronique ferroviaire – Réseau embarqué de train (TCN) –
Partie 3-3: Réseau de rame CANopen (CCN)**

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

COMMISSION
ELECTROTECHNIQUE
INTERNATIONALE

PRICE CODE
CODE PRIX

XE

ICS 45.060

ISBN 978-2-88912-072-7

**Warning! Make sure that you obtained this publication from an authorized distributor.
Attention! Veuillez vous assurer que vous avez obtenu cette publication via un distributeur agréé.**

CONTENTS

FOREWORD.....	10
INTRODUCTION.....	12
1 Scope.....	13
2 Normative references	13
3 Terms, definitions and abbreviations	14
3.1 Terms and definitions	14
3.2 Abbreviations	15
3.3 Conventions	15
4 Architecture.....	15
4.1 Content	15
4.2 Logical CANopen-based consist network	15
4.3 Network topology.....	16
4.4 Addressing	16
4.5 Data classes	17
5 Physical layer.....	17
5.1 Content	17
5.2 Cabling.....	17
5.3 Connector	17
5.4 Physical medium attachment	19
5.5 Physical signaling.....	19
6 Data Link layer	19
6.1 Content	19
6.2 CANopen data link layer	20
7 CANopen application layer	20
7.1 Content	20
7.2 Reference model	20
7.3 Field device model	20
7.4 CANopen communication objects	22
7.5 CANopen object dictionary	22
7.6 Predefined CANopen communication objects	24
7.6.1 Content	24
7.6.2 Object 1000 _h : Device type	24
7.6.3 Object 1001 _h : Error register.....	24
7.6.4 Object 1014 _h : COB-ID emergency object.....	24
7.6.5 Object 1017 _h : Heartbeat producer	24
7.6.6 Object 1018 _h : Identity object.....	24
7.6.7 Object 1029 _h : Error behavior	24
7.6.8 Object 67FF _h : Device type.....	25
7.6.9 Service data objects (SDOs).....	25
7.6.10 Process data objects (PDOs).....	25
8 Application data.....	25
8.1 Content	25
8.2 CANopen application data representation.....	25
8.3 Recommended representation principle of application data	25
8.3.1 Content	25

8.3.2	Application data for door control	25
8.3.3	Consumed door control application objects	26
8.3.4	Produced door control application objects	27
9	CANopen network management	29
9.1	Content	29
9.2	CANopen NMT slave functionality	30
9.3	CANopen manager functionality	30
9.3.1	General	30
9.3.2	Object dictionary usage	31
9.3.3	Redundant networks	31
9.4	CANopen NMT start-up	32
9.4.1	NMT startup	32
9.4.2	NMT startup simple	35
9.4.3	Start process boot NMT slave	36
9.5	Boot NMT slave	37
9.5.1	Check configuration	42
9.5.2	Check NMT state	43
9.5.3	NMT flying master start up	43
9.5.4	Error status	44
9.6	Error control	45
9.6.1	Start error control	45
9.6.2	Error handler	46
9.6.3	Bootup handler	47
9.7	Additional NMT master services and protocols	47
9.8	Object dictionary entries	47
9.8.1	Object 1020 _h : Verify configuration	47
9.8.2	Object 102A _h : NMT inhibit time	48
9.8.3	Object 1F20 _h : Store DCF	49
9.8.4	Object 1F22 _h : Concise DCF	50
9.8.5	Object 1F26 _h : Expected configuration date	52
9.8.6	Object 1F27 _h : Expected configuration time	53
9.8.7	Object 1F80 _h : NMT startup	54
9.8.8	Object 1F81 _h : NMT slave assignment	56
9.8.9	Object 1F82 _h : Request NMT	58
9.8.10	Object 1F83 _h : Request node guarding	61
9.8.11	Object 1F84 _h : Device type identification	63
9.8.12	Object 1F85 _h : Vendor identification	64
9.8.13	Object 1F86 _h : Product code	65
9.8.14	Object 1F87 _h : Revision number	66
9.8.15	Object 1F88 _h : Serial number	67
9.8.16	Object 1F89 _h : Boot time	68
9.8.17	Object 1F8A _h : Restore configuration	69
9.8.18	Object 1F91 _h : Self starting nodes timing parameters	70
10	Gateway functions	71
10.1	Content	71
10.2	Gateway architecture	72
10.3	General principles and services	73
10.3.1	Content	73
10.3.2	Gateway class definitions	73

10.3.3	Service primitives definitions	73
10.4	Network access service specification.....	73
10.4.1	SDO access services.....	73
10.4.2	PDO access services.....	75
10.4.3	CANopen NMT services.....	78
10.4.4	Device failure management services	81
10.4.5	CANopen interface configuration services	82
10.4.6	Gateway management services	84
10.4.7	Manufacturer-specific services	85
10.5	ASCII mapping of network access services	85
10.5.1	Content	85
10.5.2	Definitions	86
10.5.3	Network access command specification.....	89
11	Train network management	97
11.1	Content	97
11.2	Manager, Agents and interfaces (informative).....	98
11.3	Management message protocol (informative)	98
11.4	Object interfaces (informative).....	98
11.5	CANopen-specific management services	98
11.5.1	General	98
11.5.2	Agent interfaces on a Station connected to CANopen consist network	98
11.5.3	Management message structure for CANopen consist networks	99
11.5.4	Notation for the CANopen specific SIF_codes	99
11.5.5	Notation for a call CANopen management message	100
11.5.6	Notation for a reply CANopen management message	100
11.5.7	Notation for the TNM CANopen services command codes	100
11.6	TNM CANopen services	101
11.6.1	Content	101
11.6.2	Call_Write_CANopen_Command (with reservation)	101
11.6.3	Reply_Write_CANopen_Command (with reservation)	102
11.6.4	Call_Read_CANopen_Command (without reservation)	102
11.6.5	Reply_Read_CANopen_Command (without reservation).....	103
12	CANopen management message data handling.....	103
12.1	General.....	103
12.2	Message data format.....	105
12.3	Requirements for message data communication within CANopen networks	105
12.4	Object 1F78 _h : CANopen message data reception	106
13	Conformance testing	107
	Bibliography.....	108
	Figure 1 – Logical network architecture of the consist network.....	16
	Figure 2 – Network topology of CANopen-based consist network.....	16
	Figure 3 – 9-pin D-sub connector.....	18
	Figure 4 – 5-pin micro style connector	18
	Figure 5 – Field device model	20
	Figure 6 – Minimum field device.....	21
	Figure 7 – CANopen device structure.....	22

Figure 8 – Structure of the device type object	24
Figure 9 – Object structure	26
Figure 10 – Object structure	27
Figure 11 – Object structure	28
Figure 12 – NMT startup, part 1	32
Figure 13 – NMT startup, part 2	34
Figure 14 – NMT startup simple	35
Figure 15 – Start process boot NMT slave	36
Figure 16 – Boot NMT slave, part 1	37
Figure 17 – Boot NMT slave, part 2	39
Figure 18 – Boot NMT slave, part 3	40
Figure 19 – Check configuration	42
Figure 20 – Check NMT state	43
Figure 21 – Start error control	45
Figure 22 – Error handler	46
Figure 23 – Bootup handler	47
Figure 24 – Data stream definition of concise DCF	51
Figure 25 – Object structure	54
Figure 26 – Bit structure of the configuration value	54
Figure 27 – Object structure of the value	56
Figure 28 – Bit structure of the configuration value	57
Figure 29 – Gateway between Train backbone and CANopen consist network	72
Figure 30 – Management messages (informative)	97
Figure 31 – Agent interface on a CANopen (gateway) station for message data	99
Figure 32 – Call_Write_CANopen_Command	102
Figure 33 – Reply_Write_CANopen_Command	102
Figure 34 – Call_Read_CANopen_Command (without reservation)	103
Figure 35 – Reply_Read_CANopen_command (without reservation)	103
Figure 36 – CANopen device capable to handle TNM management messages	104
Figure 37 – Message data format comparison	105
Table 1 – Pinning for 9-pin D-sub connector	18
Table 2 – Pinning for 5-pin micro style connector	19
Table 3 – Bit timing	19
Table 4 – CANopen object dictionary structure	23
Table 5 – Value definition	26
Table 6 – Object description	26
Table 7 – Entry description	27
Table 8 – Value definition	27
Table 9 – Object description	27
Table 10 – Entry description	28
Table 11 – Value definition	29
Table 12 – Object description	29

Table 13 – Entry description	29
Table 14 – Error status	44
Table 15 – Object description	48
Table 16 – Entry description	48
Table 17 – Object description	49
Table 18 – Entry description	49
Table 19 – Object description	49
Table 20 – Entry description	50
Table 21 – Object description	51
Table 22 – Entry description	52
Table 23 – Object description	52
Table 24 – Entry description	53
Table 25 – Object description	53
Table 26 – Entry description	54
Table 27 – Value NMT master (bit: 0)	55
Table 28 – Value Start all nodes (bit: 1)	55
Table 29 – Value NMT master start (bit: 2)	55
Table 30 – Value Start node (bit: 3)	55
Table 31 – Reset all nodes (bit: 4)	55
Table 32 – Flying master (bit: 5)	55
Table 33 – Stop all nodes (bit: 6)	55
Table 34 – Exceptions for NMT start-up capable devices	56
Table 35 – Object description	56
Table 36 – Entry description	56
Table 37 – NMT slave (bit: 0)	57
Table 38 – NMT boot slave (bit: 2)	57
Table 39 – Mandatory (bit: 3)	57
Table 40 – Reset communication (bit: 4)	57
Table 41 – Software version (bit: 5)	57
Table 42 – Software update (bit: 6)	57
Table 43 – Restore (bit: 7)	58
Table 44 – Object description	58
Table 45 – Entry description	58
Table 46 – Value definition	60
Table 47 – Object description	60
Table 48 – Entry description	61
Table 49 – Value definition	62
Table 50 – Object description	62
Table 51 – Entry description	63
Table 52 – Object description	64
Table 53 – Entry description	64
Table 54 – Object description	65
Table 55 – Entry description	65

Table 56 – Object description	66
Table 57 – Entry description	66
Table 58 – Object description	67
Table 59 – Entry description	67
Table 60 – Object description	68
Table 61 – Entry description	68
Table 62 – Object description	69
Table 63 – Entry description	69
Table 64 – Object description	69
Table 65 – Entry description	70
Table 66 – Object description	70
Table 67 – Entry description	71
Table 68 – Upload SDO service	74
Table 69 – Download SDO parameters	75
Table 70 – Configure SDO timeout parameters	75
Table 71 – Configure RPDO service parameters	76
Table 72 – Configure TPDO service parameters	77
Table 73 – Read PDO data service parameters	77
Table 74 – Write PDO data service parameters	78
Table 75 – RPDO received service parameters	78
Table 76 – Start node service parameters	78
Table 77 – Stop node service parameters	79
Table 78 – Set node to pre-operational service parameters	79
Table 79 – Reset node service parameters	79
Table 80 – Reset communication service parameters	80
Table 81 – Enable node guarding service parameters	80
Table 82 – Disable node guarding service parameters	80
Table 83 – Start heartbeat consumer service parameters	81
Table 84 – Disable heartbeat consumer service parameters	81
Table 85 – Error control event received parameters	81
Table 86 – Read device error service parameters	82
Table 87 – Emergency event received service parameters	82
Table 88 – Initialize gateway service parameters	82
Table 89 – Store configuration service parameters	83
Table 90 – Restore configuration service parameters	83
Table 91 – Set heartbeat producer service parameters	83
Table 92 – Set node-ID service parameters	84
Table 93 – Start emergency consumer service parameters	84
Table 94 – Stop emergency consumer service parameters	84
Table 95 – Set default network service parameters	85
Table 96 – Start default node-ID service parameters	85
Table 97 – Get version service parameters	85
Table 98 – Syntax and CANopen data types	86

Table 99 – Command notation in BNF.....	87
Table 100 – Response notation.....	88
Table 101 – Internal error code (InEC).....	88
Table 102 – Notation for event triggered messages	88
Table 103 – Syntax for upload SDO command	89
Table 104 – Examples for upload SDO command	89
Table 105 – Syntax for Download SDO command	89
Table 106 – Examples for download SDO command	89
Table 107 – Syntax for configure SDO timeout command.....	89
Table 108 – Syntax for configure RPDO command.....	90
Table 109 – Examples for configure RPDO command	90
Table 110 – Syntax for configure TPDO command	90
Table 111 – Examples for configure TPDO command.....	90
Table 112 – Syntax for read PDO data command.....	91
Table 113 – Response syntax for read PDO data command.....	91
Table 114 – Syntax for write PDO data command	91
Table 115 – Syntax for RPDO receive command.....	91
Table 116 – Examples RPDO received command	91
Table 117 – Syntax for start node command	91
Table 118 – Syntax for stop node command	92
Table 119 – Syntax set node to pre-operational command.....	92
Table 120 – Syntax reset node command	92
Table 121 – Syntax reset communication command.....	92
Table 122 – Syntax enable node guarding command	92
Table 123 – Syntax disable node guarding command.....	93
Table 124 – Syntax start heartbeat consumer command	93
Table 125 – Syntax disable heartbeat consumer command	93
Table 126 – Syntax for error control event received command	93
Table 127 – Syntax for read device error command	94
Table 128 – Syntax for emergency event received command	94
Table 129 – Syntax for initialize gateway command	94
Table 130 – Bit rate indices	94
Table 131 – Syntax for store configuration command.....	95
Table 132 – Storage specifier	95
Table 133 – Syntax restore configuration command.....	95
Table 134 – Syntax set heartbeat producer command.....	95
Table 135 – Syntax set node-ID command.....	95
Table 136 – Syntax set default network command.....	96
Table 137 – Syntax set default node-ID command	96
Table 138 – Syntax for get version command.....	96
Table 139 – Response syntax for get version command.....	96
Table 140 – Example for get version response.....	97
Table 141 – Management message structure.....	99

Table 142 – CANopen specific SIF_codes	100
Table 143 – Notation for a call CANopen management message	100
Table 144 – Notation for a reply CANopen management message	100
Table 145 – TNM CANopen services command codes (reservation required).....	101
Table 146 – TNM CANopen services command codes (reservation not required)	101
Table 147 – Value definition for Call_Write_CANopen_Command	102
Table 148 – Value definition Reply_Write_CANopen_Command	102
Table 149 – Value definition for Call_Read_CANopen_Command (without reservation)	103
Table 150 – Value definition for Reply_Read_CANopen_Command (without reservation)	103
Table 151 – Object description	106
Table 152 – Entry description	106

INTERNATIONAL ELECTROTECHNICAL COMMISSION

**ELECTRONIC RAILWAY EQUIPMENT –
TRAIN COMMUNICATION NETWORK (TCN) –****Part 3-3: CANopen Consist Network (CCN)**

FOREWORD

- 1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
- 2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
- 3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
- 4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
- 5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.
- 6) All users should ensure that they have the latest edition of this publication.
- 7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
- 8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
- 9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC 61375-3-3 has been prepared by IEC technical committee 9: Electrical equipment and systems for railways.

The text of this standard is based on the following documents:

FDIS	Report on voting
9/1646/FDIS	9/1670/RVD

Full information on the voting for the approval of this standard can be found in the report on voting indicated in the above table.

This publication has been drafted in accordance with the ISO/IEC Directives, Part 2.

A list of all parts of IEC 61375 series, under the general title *Electronic railway equipment – Train Communication Network (TCN)*, can be found on the IEC website.

The committee has decided that the contents of this publication will remain unchanged until the stability date indicated on the IEC web site under "<http://webstore.iec.ch>" in the data related to the specific publication. At this date, the publication will be

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

INTRODUCTION

TCN is an International Standard with the aim of defining interfaces so as to achieve plug-in compatibility:

- a) between equipment located in different vehicles or consists, and
- b) between equipment and devices located within the same vehicle or consist.

One of the key success factors for the deployment of any technology is standardization and ensuring interoperability among various implementations. To facilitate interoperability a conformance test should be implemented.

In this part of IEC 61375, the TCN deals with:

the consist network based on CANopen.

In addition gateway devices between the Train Backbone and the CANopen-based consist network are considered.

This standard is structured into 13 clauses.

ELECTRONIC RAILWAY EQUIPMENT – TRAIN COMMUNICATION NETWORK (TCN) –

Part 3-3: CANopen Consist Network (CCN)

1 Scope

This part of IEC 61375 specifies the data communication bus inside consists that are based on CANopen. CANopen was developed for use in, but is not limited to, industrial automation applications. These applications may include devices such as input/output modules, motion controllers, human machine interfaces, sensors, closed-loop controllers, encoders, hydraulic valves or programmable controllers.

In the application field of rail vehicles CANopen networks are utilized to network subsystems in consists such as e.g. brake control system, diesel engine control system and interior or exterior lighting control system. In addition CANopen is utilized as consist network to enable the data exchange between the different subsystems within one single rail vehicle or a group of rail vehicles sharing the same Consist Network.

This part of IEC 61375 applies to all equipment and devices operated on a CANopen-based consist network within TCN architecture as described in IEC 61375-1.

The applicability of this standard to a TCN implementation allows for individual conformance checking of the implementation itself and is a pre-requisite for further interoperability checking between different TCN implementations. In any case, proof of compatibility between Train Backbone and the Consist Network will have to be brought by the supplier.

This part of IEC 61375 applies to the architecture of communication systems in Open trains. In addition it may be applicable to closed trains and multiple unit trains when so agreed between purchaser and supplier.

2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 61131 (all parts): *Programmable controllers*

IEC 61375-1, ed3, *Electronic railway equipment – Train Communication Network (TCN) – Part 1: General Architecture*

IEC 61375-2-1, *Electronic railway equipment – Train Communication Network (TCN) – Part 2-1: Wire Train Bus (WTB)*

IEC 61375-2-2, *Electronic railway equipment – Train Communication Network (TCN) – Part 2-2: WTB – Wire Train Bus conformance testing*

ISO/IEC 646:1991 *Information technology – ISO 7-bit coded character set for information interchange*

ISO/IEC 9899:1999, *Programming languages – C*

ISO 11898-1:2003, *Road vehicles – Controller area network (CAN) – Part 1: Data link layer and physical signaling*

ISO 11898-2:2003, *Road vehicles – Controller area network (CAN) – Part 2: High-speed medium access unit*

EN 50325-4:2002, *Industrial communication subsystems based on ISO 11898 (CAN) for controller- device interfaces – Part 4: CANopen*

3 Terms, definitions and abbreviations

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO 11898-1, ISO 11898-2, IEC 61375-1, IEC 61375-2-1 and EN 50325-4, as well as the following apply.

3.1.1

ASCII

7-bit coded character set according to ISO/IEC 646

3.1.2

CANopen device

End Device connected to a CANopen-based consist network

3.1.3

Field device

networked independent physical entity of an automation system capable of performing specified functions in a particular context and delimited by its interfaces

3.1.4

logical consist network

the way the data passes through the consist network without considering the physical interconnection of the End Devices

3.1.5

logical device

representation of a field device in terms of its objects and behavior according to a field device model that describes the device's data and behavior as viewed through a network

3.1.6

Layer Setting Services

services for adjusting bit rate and Node-ID via the communication interface of the CANopen device

3.1.7

Node-ID

- a) network-wide unique identifier for each CANopen device
- b) 7-bit coded Device Address in CANopen-based consist networks

3.1.8

Object

entity with a well-defined boundary and identity that encapsulates state and behavior

3.1.9

virtual device

entity of software capable of accomplishing a functional element of a field device

3.2 Abbreviations

For the purposes of this document, abbreviations given in ISO 11898-1, ISO 11898-2, IEC 61375-1, IEC 61375-2-1 and EN 50325-4, as well as the following apply.

ASCII	American Standard Code for Information Interchange
BNF	Backus Naur Form
CAN	Controller Area Network
CPU	Central Processing Unit
CR	Carriage Return
CRLF	Carriage Return and Line Feed
DCF	Device configuration file
FSA	Finite state automaton
ID	Identifier
LF	Line Feed
LSS	Layer Setting Services
NMT	Network management
OSI	Open system interconnect
PAS	PDO access service
PDO	Process data object
SAS	Service data object access service
SRD	SDO requesting device

3.3 Conventions

The conventions given in IEC 61375-1 shall apply.

4 Architecture

4.1 Content

This clause provides the definition of the architecture for Consist Networks based on CANopen.

4.2 Logical CANopen-based consist network

The overall Logical CANopen consist network, connecting several Virtual Devices, is illustrated in Figure 1. The CANopen-based Consist Network interconnects several Virtual Devices and sub-systems such as e.g. Train operating system (TOS), Monitoring and safety system (MSS), Auxiliary operating system (AUX), Power (drive) system (PDS), Running gear system (RGS), Brake control system (BCS), Ancillary operating system (ANC), Vehicle linkage system (VLS), Exterior lighting system (ELS), Interior lighting system (ILS), Door control system (DCS), HVAC system (HS), Passenger information system (PIS), Diagnostic system (DS), or Train-to-ground communication system (TCS).

The Logical CANopen-based consist network is connected to the Train Backbone via a Gateway. The Gateway manages the information exchange as well as the process data marshalling between the Train Backbone and the Consist Network.

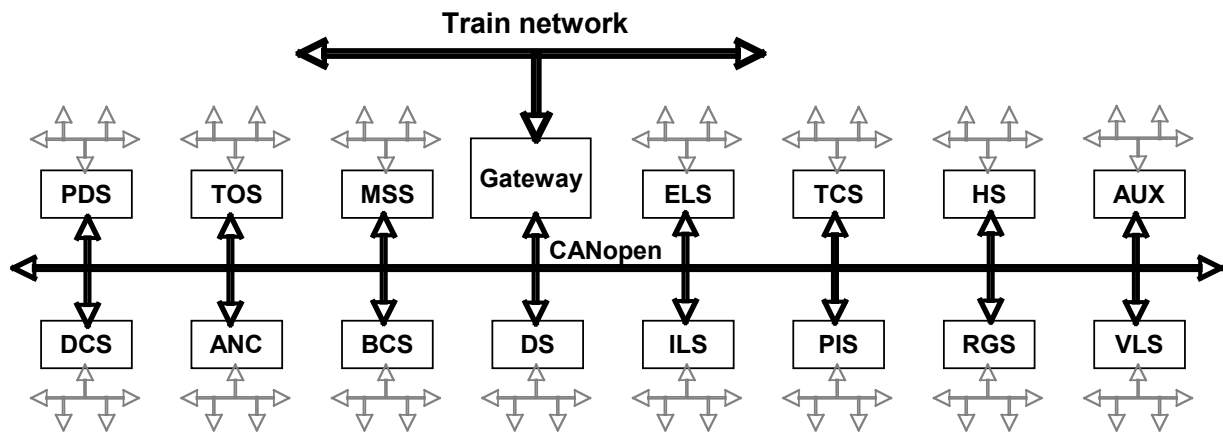


Figure 1 – Logical network architecture of the consist network

4.3 Network topology

The network topology shall be as close as possible to a single line structure with termination resistors on both sides of the network as suggested by ISO11898-2. The total network length shall not exceed 450 m, if a transmission rate of 125 kbit/s is used.

The accumulated stub length shall not exceed 110 m, and the single stub length shall not exceed 22 m, in case a transmission rate of 125 kbit/s is used. It is recommended to keep the stub lengths as short as possible. The CAN transceiver chips shall be galvanically isolated. Optocouplers are placed between CAN controller and CAN transceiver. This affects the maximum bus length depending upon the propagation delay of the optocouplers. The termination resistor on both ends of the bus lines shall be 120 Ω or higher.

The network topology is illustrated in Figure 2.

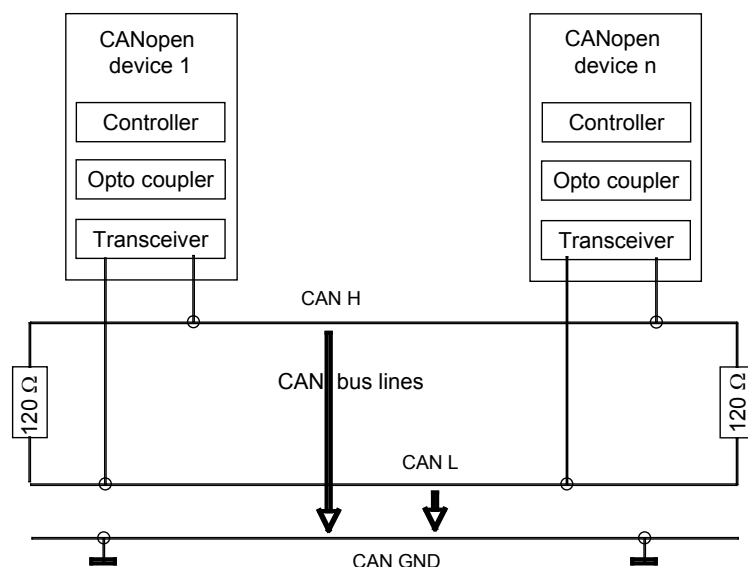


Figure 2 – Network topology of CANopen-based consist network

4.4 Addressing

Any End Device that is connected to a CANopen-based Consist Network requires a unique CANopen Node-ID in the range from 01_h to 7F_h. The CANopen Node-ID is either adjusted via hardware switches or via software. In case the CANopen Node-ID is adjusted via the CANopen communication interface of the device, the CANopen Layer Setting Services shall

be used. The CANopen Node-ID shall not be adjusted via the CANopen object dictionary of the related device.

The definition of the Layer Setting Services (LSS) is not in the scope of this standard

NOTE LSS are specified in the document CiA 305.

4.5 Data classes

To be able to exchange meaningful data across the CANopen-based Consist Network, it is necessary that the format of this data and its meaning is known by the producer and the consumer(s). This specification models this by the concept of data types.

The encoding rules define the representation of values of data types and the transfer syntax for the representations. Values are represented as bit sequences. Bit sequences are transferred in sequences of octets (bytes). For numerical data types the encoding is little endian style.

Applications often require data types beyond the basic data types. Using the compound data type mechanism, it is possible to extend the list of available data types. Some general extended data types are defined as “Visible String” or “Time of Day”. Compound data types are a means to implement user defined “DEFTYPES” in the terminology of the EN 50325-4 and not “DEFSTRUCTS”.

The data types and encoding rules provided in EN 50325-4 shall apply.

5 Physical layer

5.1 Content

This clause provides definitions for the physical layer of a CANopen-based consist network.

5.2 Cabling

The main trunk circuit shall be at least a single twisted pair of nominal characteristic impedance as defined in ISO 11898-2. In addition a CAN_Ground line as clear reference level for CAN_H and CAN_L voltage level shall be applied.

5.3 Connector

It is recommended that devices connected to a CANopen-based consist network support either the 9-pin D-sub connector or the 5-pin micro style (M12) connector. The connectors are illustrated in Figure 3 and Figure 4. The 9-pin D-sub connector shall support a pinning as defined in Table 1. The 5-pin micro style connector (M12) shall support a pinning as defined in Table 2. As the CANopen devices shall be galvanically isolated, the optional V+ line shall only be used for supply of CAN transceiver and optocoupler, in case no additional power supply is used.

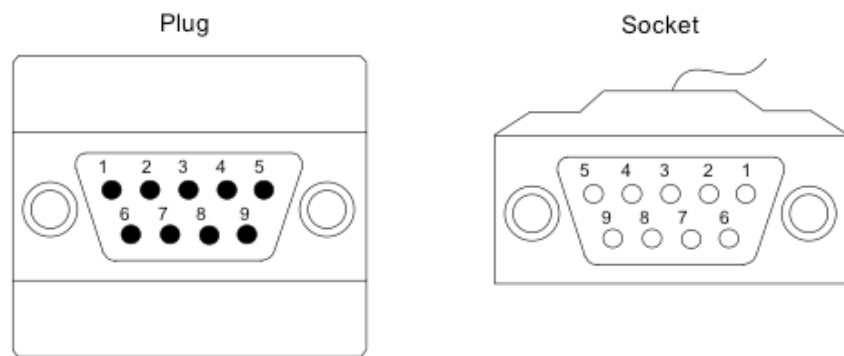


Figure 3 – 9-pin D-sub connector

Table 1 – Pinning for 9-pin D-sub connector

Pin	Signal	Description
1	-	Reserved
2	CAN_L	CAN_L bus line (dominant low)
3	CAN_GND	CAN ground
4	-	Reserved
5	(CAN_SHLD)	Optional CAN shield
6	(GND)	Optional ground
7	CAN_H	CAN_H bus line (dominant high)
8	-	Reserved
9	(CAN_V+)	Optional CAN external positive supply (dedicated for supply of transceiver and optocouplers for galvanically isolated bus nodes)

NOTE It is highly recommended not to connect CAN_GND and GND.

In case shielding is required, it is recommended to connect a shield via the metal housing of the used connector. Optionally the CAN_SHLD pin can be used.

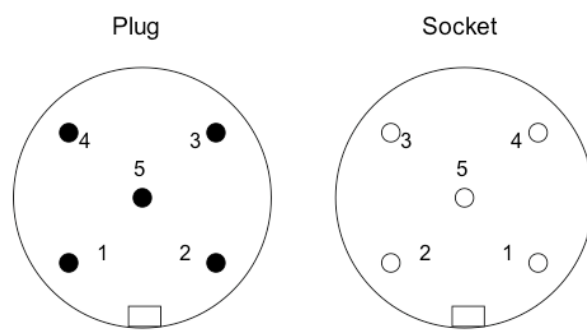


Figure 4 – 5-pin micro style connector

Table 2 – Pinning for 5-pin micro style connector

Pin	Signal	Description
1	(CAN_SHLD)	Optional CAN shield
2	(CAN_V+)	Optional CAN external positive supply (dedicated for supply of transceiver and optocouplers, if galvanic isolation of the bus node applies)
3	CAN_GND	Ground / 0V / V-
4	CAN_H	CAN_H bus line (dominant high)
5	CAN_L	CAN_L bus line (dominant low)

5.4 Physical medium attachment

The physical medium for a device connected to CANopen-based consist networks shall be a differentially driven two-wire bus line with common return according to high-speed transmission specification in ISO 11898-2.

Using the high-speed transceiver according to ISO 11898-2 the maximum rating for V_{CAN_H} and V_{CAN_L} shall be +16 V. Galvanic isolation between CANopen devices is optional. It is recommended to use a CAN transceiver that is capable of sustaining misconnection of any of the wires of the connector including the optional V+ voltages of up to 30 V.

5.5 Physical signaling

The bit encoding/decoding and synchronization shall meet the requirements defined in ISO 11898-1.

The bit timing shall meet the requirements defined in ISO 11898-1 and the definitions as given in Table 3. The recommended location of the sample point is as close as possible to 87,5 % of the related bit time.

Table 3 – Bit timing

Bit rate	Nominal bit time t_b μs	Valid range for location of sample point %
1 Mbit/s	1	75 to 90
800 kbit/s	1,25	75 to 90
500 kbit/s	2	85 to 90
250 kbit/s	4	85 to 90
125 kbit/s	8	85 to 90
50 kbit/s	20	85 to 90
20 kbit/s	50	85 to 90
10 kbit/s	100	85 to 90

Devices connected to a CANopen-based consist network shall support a bit rate of 125 kbit/s. Optionally further bit rates as given in Table 3 may be supported.

6 Data Link layer

6.1 Content

This clause provides definitions for the data link layer of a CANopen-based consist network.

6.2 CANopen data link layer

The described CANopen-based consist network shall be based on a data link layer and its sub-layers according to ISO 11898-1.

This specification is based on the CAN base frame format with 11-bit CAN-Identifier. Therefore it is not required to support the CAN extended frame format with 29-bit CAN-Identifier.

NOTE As certain applications require the usage of the CAN extended frame format, the network can be operated in this mode as well, in case all connected CANopen devices support this format.

7 CANopen application layer

7.1 Content

This clause provides definitions with regard to the upper layers of the ISO OSI reference model for devices connected to a CANopen-based consist network.

7.2 Reference model

Devices connected to a CANopen-based consist network compliant to this specification shall use the reference model, as defined in EN 50325-4.

7.3 Field device model

Devices connected to a CANopen-based consist network shall comply with the field device model as given in Figure 5.

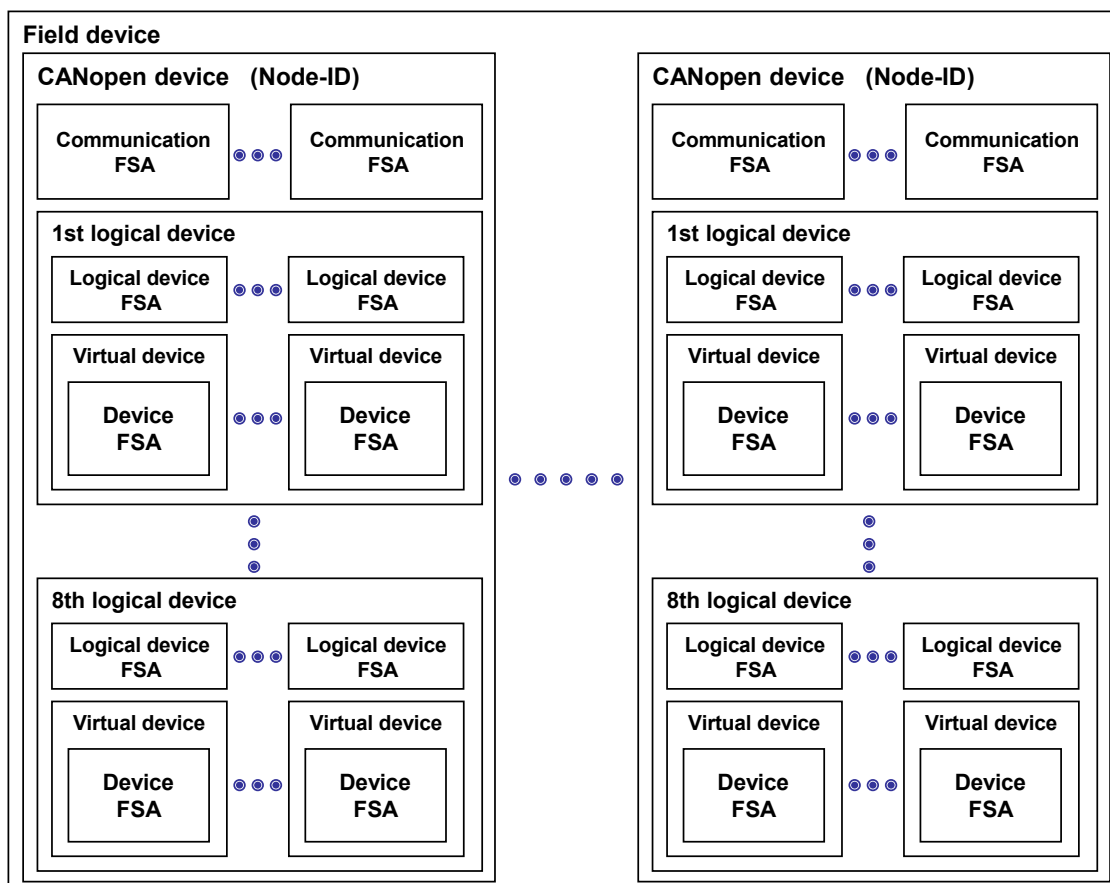


Figure 5 – Field device model

The field device as illustrated in Figure 5 shall provide at least one CANopen device. Each CANopen device within the field device shall provide at least one associated network interface comprising the data link layer protocol (see Clause 6) and the physical layer (see Clause 5), one CANopen Node-ID, and at least one communication FSA. The first communication FSA contains the NMT slave state machine as defined in EN 50325-4. Additional communication FSAs contain an emergency state machine (see EN 50325-4) and others. A CANopen device shall support at least one and may support up to eight logical devices. Each logical device may contain a number of virtual devices and optionally a logical device FSA. A virtual device contains a virtual device FSA and is not distributed to several logical devices. The minimum field device is shown in Figure 6.

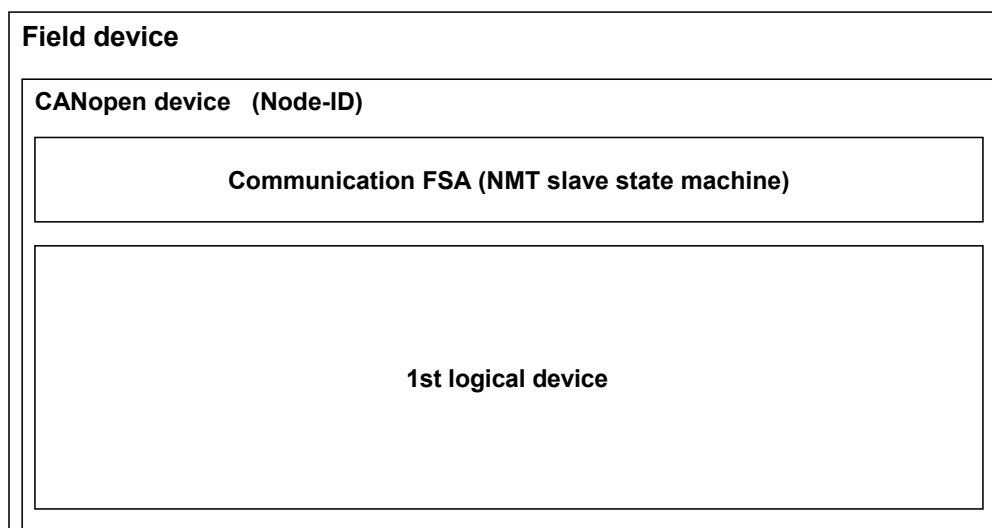


Figure 6 – Minimum field device

A CANopen device is structured as shown in Figure 7:

- **Communication** – This function unit provides the communication objects and the appropriate functionality to transport data items via the underlying network structure.
- **Object dictionary** – The object dictionary is a collection of all the data items which have an influence on the behavior of the application objects, the communication objects and the state machine used on this device.
- **Application** – The application comprises the functionality of the device with respect to the interaction with the process environment.

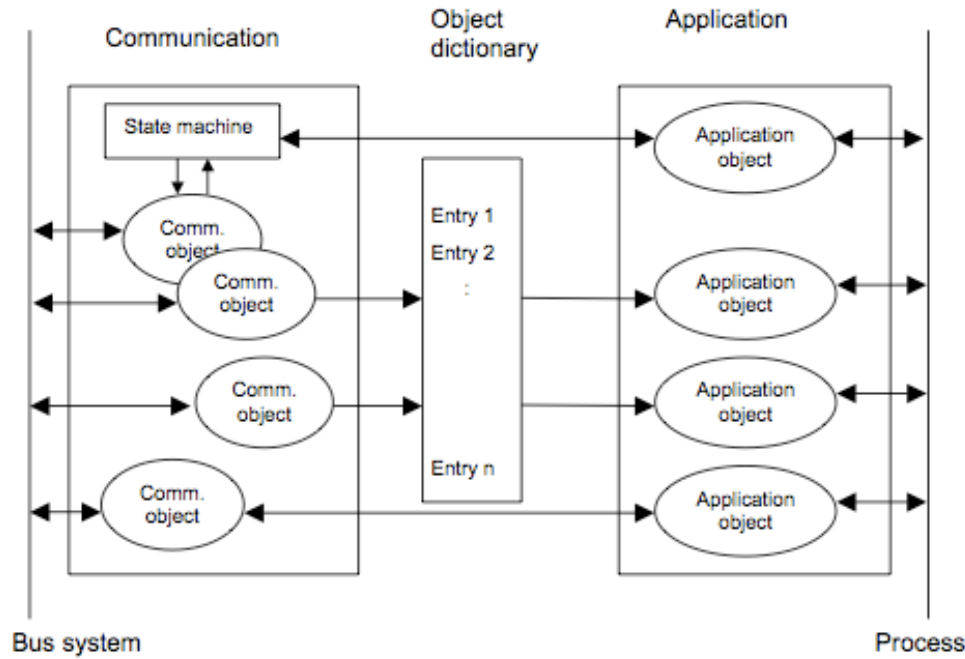


Figure 7 – CANopen device structure

7.4 CANopen communication objects

Devices connected to a CANopen-based consist network shall support all mandatory communication objects of EN 50325-4 as well as the CANopen network management state machine as defined in EN 50325-4.

In order to reduce configuration effort for simple CANopen networks a CAN-ID allocation scheme is defined. These CAN-IDs shall be available in the NMT state Pre-operational directly after the NMT state Initialization (if no modifications have been stored). The objects SYNC, TIME, EMCY write and PDO may be deleted and re-created with new CAN-IDs by means of dynamic distribution. A CANopen device shall provide the corresponding CAN-IDs only for the supported communication objects. The CAN-ID allocation scheme is defined in EN 50325-4.

7.5 CANopen object dictionary

The CANopen object dictionary, as defined in EN 50325-4, contains a maximum of 65536 objects that are addressed through a 16-bit index and up to 256 sub-indices per object, which are addressed through an 8-bit sub-index.

The static data types at indices from 0001_h to 001F_h contain type definitions for standard data types like BOOLEAN, INTEGER, UNSIGNED, floating point, string, etc.

Complex data types at indices from 0020_h to 003F_h are pre-defined structures that are composed of standard data types and are common to all CANopen devices.

Manufacturer-specific complex data types at indices from 0040_h to 005F_h are structures composed of standard data types but are specific to a particular CANopen device.

CANopen device profiles may define additional data types specific to their device type. The static data types and the complex data types defined by the CANopen device profile are listed at indices from 0060_h to 025F_h.

A CANopen device optionally provides the structure of the supported complex data types (indices from 0020_h to 005F_h and from 0060_h to 025F_h) at read access to the corresponding index. Sub-index 00_h then provides the highest sub-index supported at this index, and the following sub-indices contain the data type encoded as UNSIGNED16 according to EN 50325-4.

The communication profile area at indices from 1000_h to 1FFF_h contains the communication specific parameters. These objects are common to all CANopen devices.

The standardized profile area at indices from 6000_h to 9FFF_h contains all data objects common to a class of CANopen devices that may be read or written via the network. The CANopen device profiles use objects from 6000_h to 9FFF_h to describe parameters and functionality.

The object dictionary concept caters for optional features, which means a manufacturer may not provide certain extended functionality on his CANopen devices but if he wishes to do so he shall do it in a pre-defined fashion. Space is left in the object dictionary at indices from 2000_h to 5FFF_h for manufacturer-specific functionality.

The network variables at indices from A000_h to AFFF_h contain input variables and output variables, which are part of a programmable CANopen device.

The system variables at indices from B000_h to BFFF_h contain input variables and output variables, which are part of an underlying CANopen network in a hierarchical sense.

The general CANopen object dictionary structure is illustrated in Table 4.

Table 4 – CANopen object dictionary structure

Index	Object
0000 _h	not used
0001 _h – 001F _h	Static data types
0020 _h – 003F _h	Complex data types
0040 _h – 005F _h	Manufacturer-specific complex data types
0060 _h – 025F _h	Device profile specific data types
0260 _h – 03FF _h	reserved
0400 _h – 0FFF _h	reserved
1000 _h – 1FFF _h	Communication profile area
2000 _h – 5FFF _h	Manufacturer-specific profile area
6000 _h – 67FF _h	Standardized profile area 1 st logical device
6800 _h – 6FFF _h	Standardized profile area 2 nd logical device
7000 _h – 77FF _h	Standardized profile area 3 rd logical device
7800 _h – 7FFF _h	Standardized profile area 4 th logical device
8000 _h – 87FF _h	Standardized profile area 5 th logical device
8800 _h – 8FFF _h	Standardized profile area 6 th logical device
9000 _h – 97FF _h	Standardized profile area 7 th logical device
9800 _h – 9FFF _h	Standardized profile area 8 th logical device
A000 _h – AFFF _h	Standardized network variable area
B000 _h – BFFF _h	Standardized system variable area
C000 _h – FFFF _h	reserved

7.6 Predefined CANopen communication objects

7.6.1 Content

This clause provides the basic CANopen communication capability for CANopen devices that participate in a CANopen-based consist network according to this specification.

7.6.2 Object 1000_h: Device type

This object provides the type of device and its functionality. The value 0000_h for the device profile number indicates a logical device that does not follow a standardized profile. In this case the additional information is 0000_h (if no further logical device is implemented) or FFFF_h (if a further logical device is implemented).

For multiple logical device modules the additional information parameter is FFFF_h and the device profile number referenced by object 1000_h is the profile of the first logical device in the object dictionary. All other profiles of a multiple logical device module identify their profiles at objects 67FF_h + x * 800_h with x = internal number of the logical device (from 1 to 8) minus 1. These objects describe the device type of the preceding logical device, having the very same value definition as object 1000_h.

Figure 8 illustrates the structure of the object. The value definition, the object description and the entry description are specified in EN 50325-4.

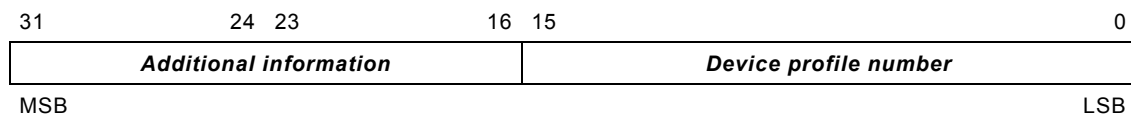


Figure 8 – Structure of the device type object

NOTE Devices connected to CANopen-based consist networks may follow the CANopen application profile for train vehicle control networks CiA 421. This CANopen application profile defines application data, based on UIC 556, that is exchanged in a CANopen-based consist network.

7.6.3 Object 1001_h: Error register

This object provides the error information. The CANopen device maps internal errors into this object. This error information is published as part of the emergency message. The value definition, object description and entry description are specified in EN 50325-4.

7.6.4 Object 1014_h: COB-ID emergency object

This object shall be implemented. It is specified in EN 50325-4. The CAN-ID, which is part of this object, shall not be changed.

7.6.5 Object 1017_h: Heartbeat producer

All CANopen devices that are connected to a CANopen-based consist network shall implement this object. It is specified in EN 50325-4.

The devices shall support heartbeat message transmissions from 100 ms to 1 000 ms.

7.6.6 Object 1018_h: Identity object

This object provides general information about the device as specified in EN 50325-4.

7.6.7 Object 1029_h: Error behavior

This object specifies to which state the device is set, in case a communication error or a device-internal error is detected. It is specified in EN 50325-4.

7.6.8 Object 67FF_h: Device type

This object shall describe the first logical device in a multiple device module according to EN 50325-4.

7.6.9 Service data objects (SDOs)

Any CANopen device supports the first SDO server channel. CANopen devices connected to a CANopen-based consist network may support additional SDO server or client channels.

In case an additional SDO channel is supported, the related SDO parameter set is supported in the CANopen object dictionary as defined in EN 50325-4.

There are no further SDO channels pre-defined by this specification.

7.6.10 Process data objects (PDOs)

CANopen devices operated in a CANopen-based consist network may support up to 512 PDOs in transmit as well as receive direction.

In case a CANopen device supports a PDO, the related PDO communication parameter and mapping entries are supported in the CANopen object dictionary as defined in EN 50325-4.

There are no PDOs pre-defined.

8 Application data

8.1 Content

This clause provides the representation of application data that is communicated between a CANopen-based consist network and a Train Backbone via a Gateway.

NOTE 1 The application data is described in application profile of TCN.

In general application data is managed in the CANopen object dictionary index ranges 2000_h to 5FFF_h and 6000_h to 9FFF_h. A manufacturer specific device behavior is controlled via the index range 2000_h to 5FFF_h. A standardized CANopen device behavior is controlled via objects in the index range 6000_h to 9FFF_h.

NOTE 2 Object dictionary entries in the index range 6000h to 9FFFh are specified in CANopen device and application profiles and available at CAN in Automation.

8.2 CANopen application data representation

As the application data for TCN is not defined yet, the representation of the TCN application data in the standardized object dictionary index range is not defined.

8.3 Recommended representation principle of application data

8.3.1 Content

In this clause a representation principle of application data is provided for the representation of application data in CANopen-based consist networks. Therefore it is possible to map the process to CANopen process data objects (PDOs) for process data transfer.

8.3.2 Application data for door control

This clause provides the recommended representation principle by means of door control information, which is communicated between a CANopen-based consist network and a Train

Backbone via a Gateway. The represented process data is therefore available in the CANopen-based consist network and can be communicated via the CANopen communication objects.

The access type is given for the interface of the gateway device to the CANopen-based consist network. Other devices such as e.g. door controllers or door units, which are connected to a CANopen-based consist network manage the data as defined in this clause, at the given CANopen object dictionary index. The access type of the object may be changed to the appropriate access type as defined in EN 50325-4.

NOTE The illustrated application data is derived from the document UIC 556 and is given in the representation of the CANopen application profile CiA 421 train vehicle control network.

8.3.3 Consumed door control application objects

8.3.3.1 Object 6007_h: External door status word export

This object shall indicate the status of the external doors of the local rail vehicle. Via the gateway to the Train Backbone, this information is available in the TCN. Figure 9 specifies the object structure and Table 5 specifies the value definition. Table 6 specifies the object description and Table 7 specifies the entry description.

NOTE The object shall correspond to R3-telegram octet 20, which is defined in UIC 556.

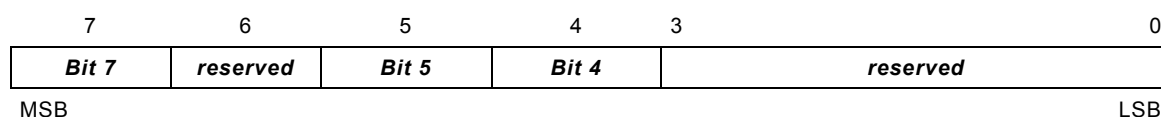


Figure 9 – Object structure

Table 5 – Value definition

Bit	Value	Value definition
Bit4	0	At least one left door is open
	1	All left doors locked
Bit 5	0	At least one right door is open
	1	All right doors locked
Bit 7	0	Side selective door blocking is not in operation
	1	Side selective door blocking is in operation
Reserved		Reserved (shall be ignored)

Table 6 – Object description

Attribute	Value
INDEX	6007 _h
Name	External doors status word export
Object code	Variable
Data type	UNSIGNED8
Category	Optional

Table 7 – Entry description

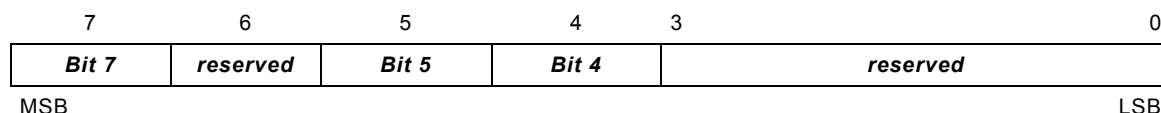
Attribute	Value
Sub-index	00 _h
Access	rw
PDO mapping	Optional
Value range	See value definition
Default value	Manufacturer specific

8.3.4 Produced door control application objects

8.3.4.1 Object 6006_h: External door status word import

This object shall provide the status of the external doors of other rail vehicles. Each sub-index shall provide the door status of that vehicle, which corresponds with the UIC vehicle number. Therefore the external door status of other vehicles is available within the CANopen-based consist network. The overall door status of all vehicles is provided in sub-index 21_h. Figure 10 specifies the object structure and Table 8 specifies the value definition. Table 9 specifies the object description and Table 10 specifies the entry description.

NOTE The object shall correspond to R3-telegram octet 20, which is defined in UIC 556.

**Figure 10 – Object structure****Table 8 – Value definition**

Bit	Value	Value definition
Bit4	0	At least one left door is open
	1	All left doors locked
Bit 5	0	At least one right door is open
	1	All right doors locked
Bit 7	0	Side selective door blocking is not in operation
	1	Side selective door blocking is in operation
Reserved		Reserved (shall be ignored)

Table 9 – Object description

Attribute	Value
INDEX	6006 _h
Name	External doors status word import
Object code	ARRAY
Data type	UNSIGNED8
Category	Optional

Table 10 – Entry description

Attribute	Value
Sub-index	00 _h
Description	Highest subindex supported
Entry category	Mandatory
Access	ro
PDO mapping	No
Value range	01 _h to 21 _h
Default value	No
Sub-index	01 _h
Description	Door status UIC vehicle 1
Entry category	Optional
Access	ro
PDO mapping	Optional
Value range	see Table 46
Default value	No
to	
Sub-index	20 _h
Description	Door status UIC vehicle 32
Entry category	Optional
Access	ro
PDO mapping	No
Value range	Optional
Default value	No
Sub-index	21 _h
Description	Door status train
Entry category	Optional
Access	ro
PDO mapping	No
Value range	Optional
Default value	No

8.3.4.2 Object 6001_h: External door command

This object shall provide the first control word for the external doors. Figure 11 specifies the object structure and Table 11 specifies the value definition. Table 12 specifies the object description and Table 13 specifies the entry description.

NOTE The object shall correspond to R3-telegram octet 20, which is defined in UIC 556.

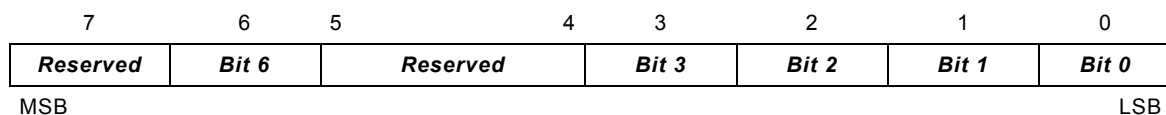


Figure 11 – Object structure

Table 11 – Value definition

Bit	Value	Value definition
Bit 0	0	Inactive
	1	All doors close
Bit 1	0	Inactive
	1	Interrupt door closing
Bit 2	0	Release all left doors
	1	Lock all left doors
Bit 3	0	Release all right doors
	1	Lock all right doors
Bit 6	0	Not extend footstep
	1	Extend footstep
Reserved		Reserved (shall be ignored)

Table 12 – Object description

Attribute	Value
INDEX	6001 _h
Name	External door command 1
Object code	Variable
Data type	UNSIGNED8
Category	Optional

Table 13 – Entry description

Attribute	Value
Sub-index	00 _h
Access	ro
PDO mapping	Optional
Value range	See value definition
Default value	No

9 CANopen network management

9.1 Content

This clause provides the network management for CANopen-based consist networks. The definition of the network management includes the definition of the network startup behavior as well as definitions that are related to networks that operate without NMT master, networks with one CANopen device capable of the NMT master mode. These definitions are intended to be an add-on to the CANopen application layer and communication profile provided in EN 50325-4.

CANopen consist networks with more than one CANopen device capable of the NMT master mode (NMT flying master) are not in the scope of this specification. However such networks are not excluded.

NOTE With regard to an increased availability, within CANopen-based consist networks several CANopen devices with NMT master functionality may reside. As at a certain time, only one active NMT master is allowed, mechanisms are required, that enable overtaking of the NMT master functionality from one CANopen device to another. These services are part of the CANopen flying master functionality that is described in CiA 302.

9.2 CANopen NMT slave functionality

Devices operated in a CANopen-based consist network shall provide CANopen NMT-slave functionality and shall implement all mandatory communication services and protocols as well as all mandatory objects as specified in EN 50325-4. The minimum communication objects are defined in 7.6.

Optionally further communication objects may be supported. In addition the CANopen manager functionality may be supported.

9.3 CANopen manager functionality

9.3.1 General

This subclause provides the definition of the CANopen manager functionality for CANopen-based consist networks.

Besides the application process several different additional functionalities exist in a CANopen network. These functionalities are referred to by different terms. This subclause is intended to clarify these terms.

Within a distributed system the application process is divided into several parts running on different CANopen devices. From the applications point of view usually one CANopen device is responsible for the control of the system. This CANopen device is called *application master*.

From the network's point of view there are several additional functionalities, which not directly deal with the application but provide application-supporting functions. These additional functionalities are based on a master/slave, client/server or producer/consumer relationship.

As it is common to combine several of the additional functionalities in one CANopen device the term CANopen Manager is introduced.

A CANopen device is referred to as CANopen manager, if it provides the NMT master functionality and at least one of the functionalities SDO manager or Configuration manager.

9.3.1.1 NMT master

The network management (NMT) provides services for controlling the network behavior of CANopen devices as defined in EN 50325-4. All CANopen devices that are participating in a CANopen consist network, referred to as NMT slaves, are controlled by services provided by an NMT master. Usually the NMT master application is also part of the application master.

The device, in which the NMT master functionality resides is a fully CANopen device. In addition to the NMT master functionality it supports all functions and objects, which are indicated as mandatory in EN 50325-4.

9.3.1.2 Flying master

The flying master mechanism provides services for a hot stand-by NMT master within a CANopen network. The flying master is an optional functionality within a CANopen device. The CANopen device implementing the flying master implements the NMT master functionality. The definition of the responsibilities, functionalities, and services of the flying master is not in the scope of this document.

NOTE Definitions for the flying master functionality are provided in CiA 302.

9.3.1.3 SDO manager

The SDO manager is an optional functionality, responsible for handling of the dynamic establishment of SDO connections. If an SDO manager is present in a CANopen network it resides together with the NMT master on the same CANopen device.

NOTE Definitions for the SDO manager functionality are provided in CiA 302.

9.3.1.4 Configuration manager

The Configuration Manager is an optional functionality, which provides mechanisms for configuration of CANopen devices in a CANopen network during boot-up. The mechanisms are called Configuration Management (CMT). If the Configuration Manager is present in a CANopen network it resides together with the NMT master on the same CANopen device.

NOTE Definitions for the Configuration manager functionality are provided in CiA 302.

9.3.1.5 SYNC producer

The SYNC producer is an optional functionality, which is responsible for transmitting the SYNC object. It may reside on any CANopen device in the CANopen-based consist network. The relevant CANopen object dictionary entries are defined in EN 50325-4.

9.3.1.6 TIME producer

The TIME producer is an optional functionality, which is responsible for transmitting the TIME STAMP object. It may reside on any CANopen device in the CANopen-based consist network. The time stamp object is defined in EN 50325-4.

9.3.1.7 LSS master

The layer setting services (LSS) provide services for configuring layer 2 (bit timing) and NMT (CANopen Node-ID) via CAN. The LSS master configures LSS slaves by executing LSS services. The definition of the layer setting services is not in the scope of this document.

NOTE Definitions for the layer setting services are provided in CiA 305.

9.3.2 Object dictionary usage

Several objects relating to the configuration and validation of CANopen devices are of object type ARRAY. The sub-index of the entries of such an ARRAY corresponds to the CANopen Node-ID of the CANopen device. Those objects may have less than 127 entries. In that case all of these objects shall have the very same set of supported entries. Sub-index 00_h shall provide the highest sub-index supported at this index. Related object description in Clause 9 refer to this as *SupportedNodeID* condition.

NOTE For a CANopen manager to be universally usable it is recommended to support the full sub-index range of 01_h to $7F_h$.

9.3.3 Redundant networks

Redundant networks are required for high-availability applications. Redundant networks consist of at least two CAN lines. This allows communication between CANopen devices, in case of a single failure within the physical interconnection between the CANopen devices. The definition of the responsibilities, functionalities, and services for redundant networks is not in the scope of this document.

NOTE Definitions for setting up redundant CANopen networks are provided in CiA 302.

9.4 CANopen NMT start-up

9.4.1 NMT startup

CANopen managers shall behave according to the NMT slave state machine as defined in EN 50325-4. Before transition from NMT state *Pre-operational* to NMT state *Operational* of the CANopen manager, all assigned NMT slaves shall be booted. The main flow chart for the procedure is defined in Figure 12 and Figure 13. The simplest flow chart for the startup is defined in Figure 14.

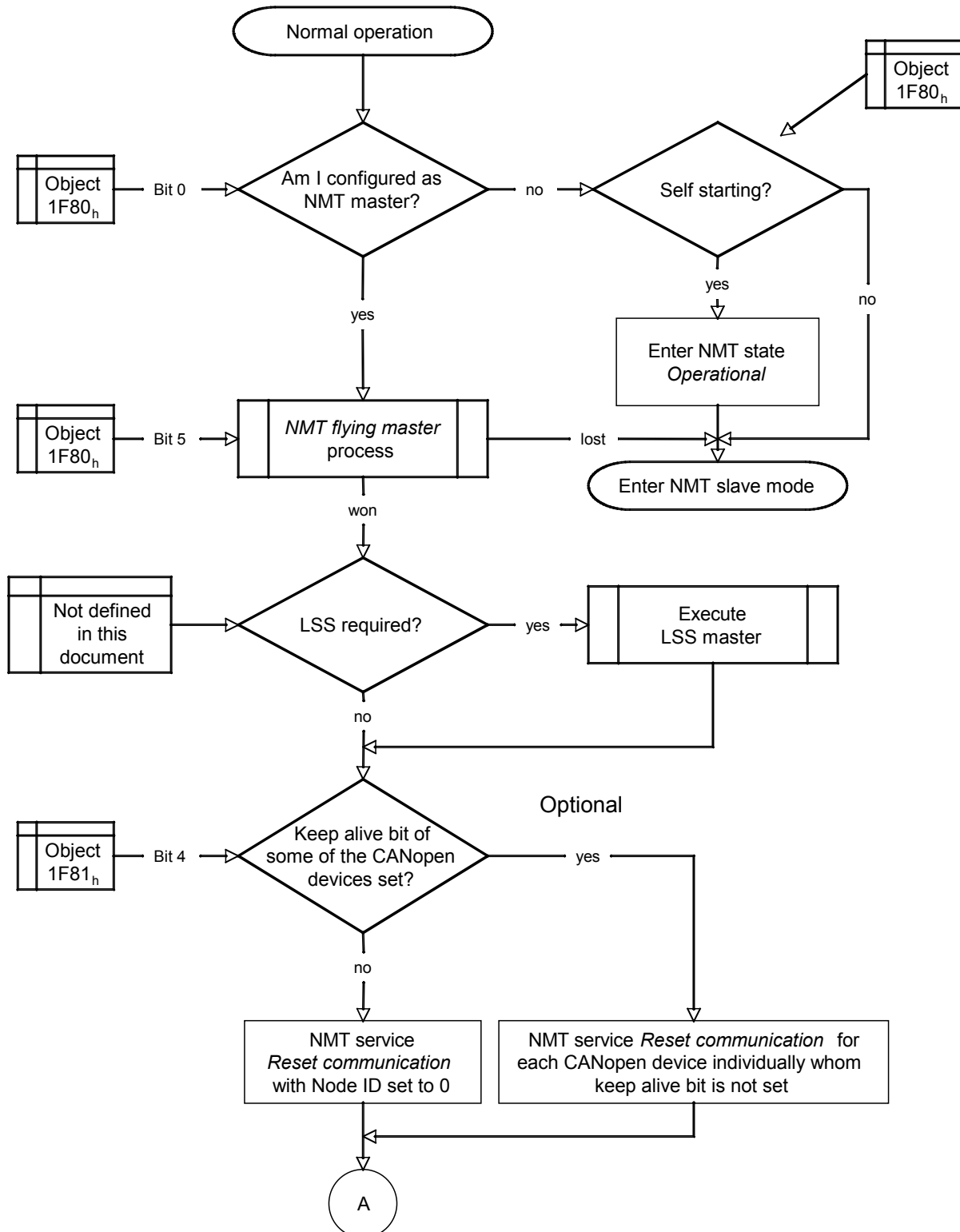


Figure 12 – NMT startup, part 1

The process NMT startup as shown in Figure 12 consists of the following basic steps:

NOTE 1 The process execute LSS master is not in the scope of this standard.

- a) Bit 0 of object 1F80_h (see clause 9.8.7) is used to decide whether this CANopen device shall be the NMT master. If the CANopen device is NMT master the process shall continue. If the CANopen device is configured as self-starting device, it shall enter the NMT state *Operational* automatically. In case the CANopen device is not the NMT master the process shall end.
- b) Bit 5 of object 1F80_h (see clause 9.8.7) is used to decide whether this CANopen device shall participate in the service *NMT flying master negotiation*. If the CANopen device shall participate in the service *NMT flying master negotiation* and the CANopen device lost the service *NMT flying master negotiation* the CANopen device shall not become NMT master. The definition of the NMT flying master negotiation is not in the scope of this document.

NOTE 2 The description of the NMT flying master negotiation is provided in CiA 302.

- c) If LSS is required to set the CANopen Node-ID and bit rate of other CANopen devices within the network the NMT master shall execute the LSS master services. The LSS master services may be executed at any time. The precise definition of the LSS master services is not in the scope of this document.

NOTE 3 The description of the layer setting services is provided in CiA 305.

- d) Bit 4 of all entries of object 1F81_h (see clause 9.8.8) is used to decide whether the NMT master shall perform the NMT service *Reset communication* with CANopen Node-ID set to 0 or if the NMT service *Reset communication* shall be performed for each CANopen device in the network individually. If at least for one entry of object 1F81_h bit 4 is set to 1_b and the corresponding CANopen device is in NMT state *Operational* the NMT master shall not issue the NMT service *Reset communication* with CANopen Node-ID set to 0. In this case each CANopen device shall be reset individually. This shall also include all CANopen Node-IDs that are not part of the slave list 1F81_h.

NOTE 4 This will force potentially existing CANopen devices that are not configured in the slave list to transmit a boot-up message. By this the CANopen Manager will recognize unconfigured CANopen devices via the boot-up handler (Figure 23). The NMT service *Reset communication* shall not apply to the NMT master itself.

If after step d) the NMT service *Reset communication* is necessary, for any reason, then the NMT service *Reset communication* shall be executed with the CANopen Node-ID of that NMT slave. This shall apply independently from the configuration for that NMT slave in object 1F81_h. The process NMT startup resumes in Figure 13.

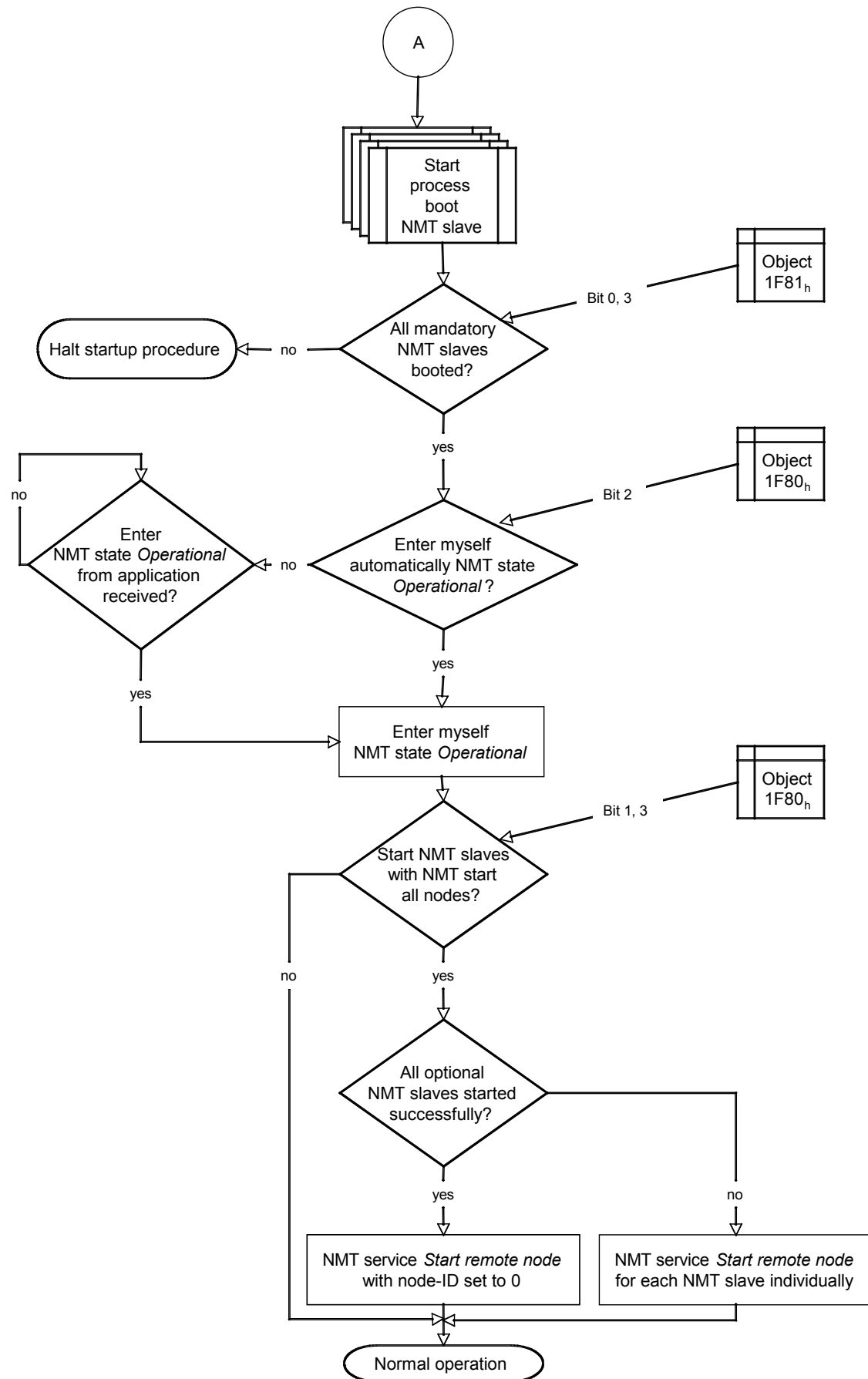


Figure 13 – NMT startup, part 2

- e) The NMT master shall start the process *start process boot NMT slave* for all NMT slaves as shown in Figure 15. For all NMT slaves that are marked as mandatory (bit 0 and bit 3 of object 1F81_h; see 9.8.8) the process *start process boot NMT slave* shall terminate successfully.
- f) If an error is detected during process *boot slave* for the NMT slaves that are marked as mandatory the process *NMT startup* shall be stopped.
- g) Bit 2 of object 1F80_h (see 9.8.7) is used to decide whether the NMT master shall enter the NMT state *Operational* automatically by itself or shall wait until it is requested by the application running on the very same CANopen device.
- h) Under the following conditions
 - Bit 3 of object 1F80_h is set to 0_b,
 - Bit 1 of object 1F80_h is set to 1_b,
 - and all NMT slaves listed in 1F81_h booted successfully
 the NMT service *Start remote node* shall be performed with CANopen Node-ID set to 0.
 Under the following conditions
 - Bit 3 of object 1F80_h is set to 0_b,
 - Bit 1 of object 1F80_h is set to 1_b,
 - and not all NMT slaves listed in 1F81_h booted successfully
 the NMT service *Start remote node* shall be performed for each NMT slave individually.
- i) The process *NMT startup* ended successfully and the NMT master shall proceed with normal operation.

Occurrence and detection of NMT slaves not listed in 1F81_h falls into the responsibility of the application.

9.4.2 NMT startup simple

Since nearly all objects and features are optional it is possible to implement a basic NMT master, which may make sense for some applications. Removing all optional parts in the definitions above results in the process *NMT startup simple* as shown in Figure 14. Even a simple NMT master supports all mandatory objects, which are specified in EN 50325-4 and which are defined in this standard.

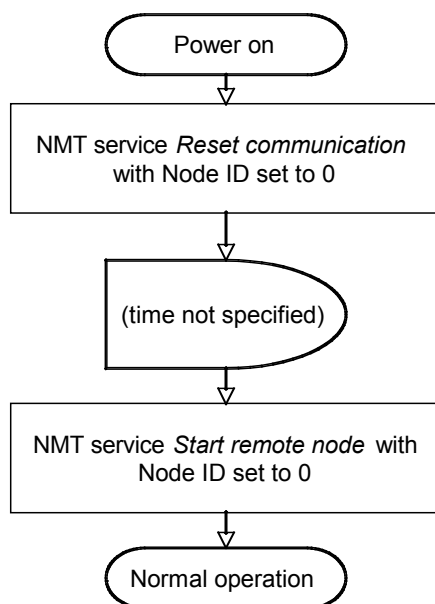


Figure 14 – NMT startup simple

9.4.3 Start process boot NMT slave

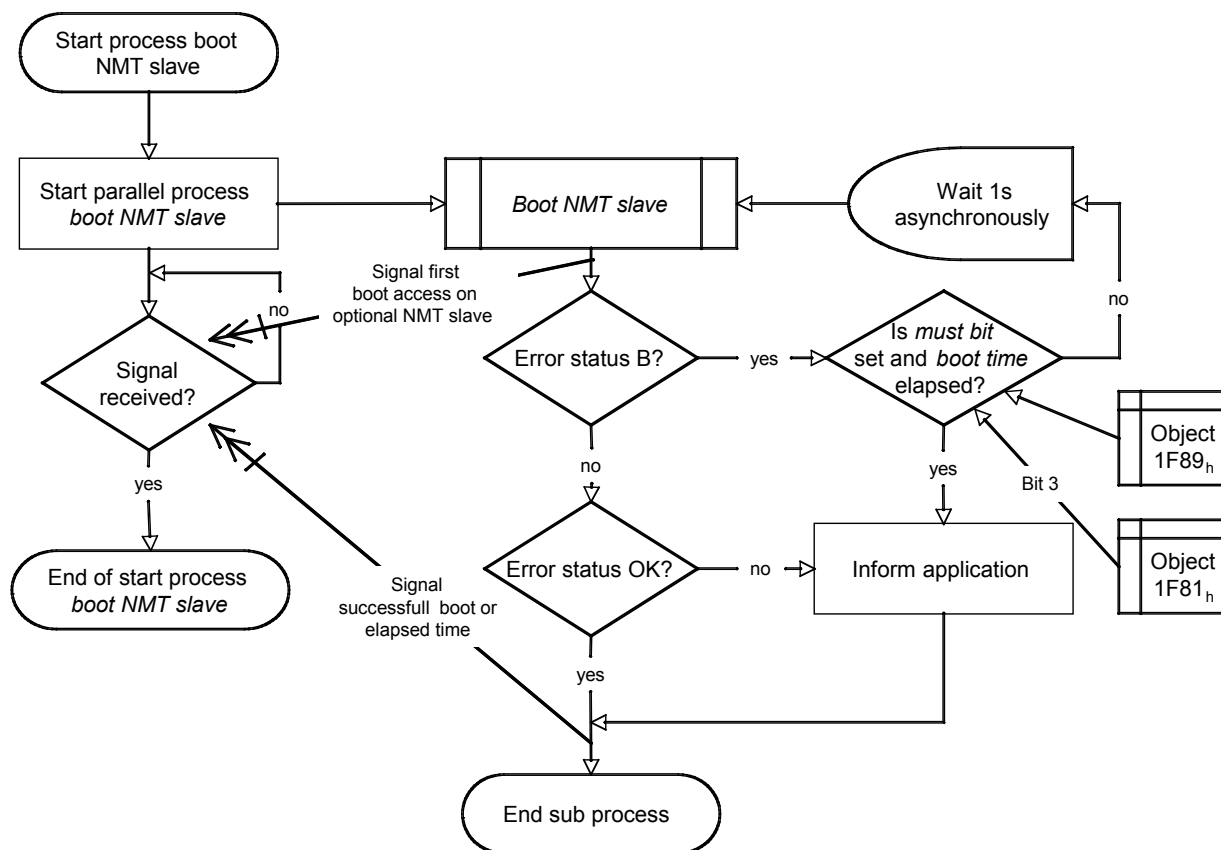


Figure 15 – Start process boot NMT slave

The process *start process boot NMT slave* as shown in Figure 15 shall include the following steps:

- Start parallel process *boot NMT slave*.
- Mandatory NMT slaves: wait for completion of the process *boot NMT slave*.
Optional NMT slaves: wait for signal that the process *Boot NMT slave* was performed.

The parallel process shall

- Perform the process boot NMT slave (see 9.5).
- Create signal for every try of the process *boot NMT slave*.

If the process *boot NMT slave* returned with status OK the process shall terminate. This process shall run endlessly for any optional NMT slave until the process *Boot NMT slave* finishes with status OK.

NOTE The recommended cycle time is 1 s for bit rate higher than 125 kbit/s.

If the process *boot NMT slave* returned with error status B for mandatory NMT slaves and the elapsed time is greater than the configured value of object 1F89_h then the application shall be informed and this sub-process shall end.

The sub-process of the process *start process boot NMT slave* shall run asynchronously to other processes.

9.5 Boot NMT slave

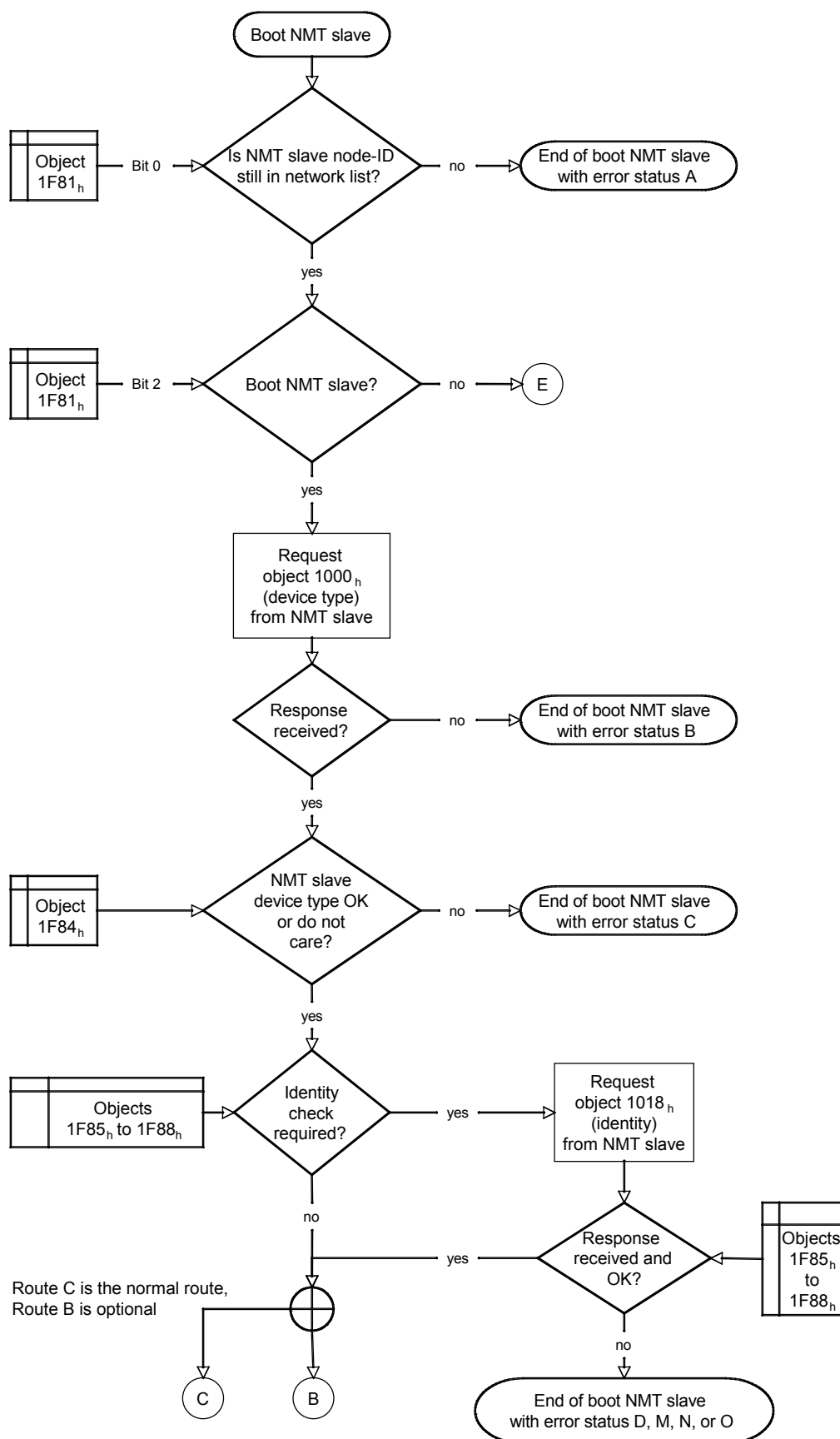


Figure 16 – Boot NMT slave, part 1

The process *boot NMT slave* as shown in Figure 16 consists of the following steps:

- a) Bit 0 of object 1F81_h is used to decide upon whether the NMT slave shall be processed or if the process shall terminate with error status.
- b) Bit 2 of object 1F81_h is used to decide upon whether the NMT slave shall be configured and started.
- c) Upload object 1000_h from the NMT slave. In case no response is received the process shall terminate with error status.
- d) In case the value of object 1F84_h for the NMT slave is unequal 0 the value shall be checked against object 1000_h. In case both values are different the process shall terminate with error status.
- e) In case the values of objects 1F85_h to 1F88_h are unequal 0, the particular object values shall be checked against their corresponding values of the object 1018_h from the NMT slave. In case one of the values of object 1F85_h to 1F88_h is different to the corresponding values of the object 1018_h from the NMT slave the process shall terminate with error status.

The process *boot NMT slave* may continue with an optional part 2 (see Figure 17), which introduces two optional features:

- keeping alive CANopen devices initially in NMT state *Operational*, and
- managing application software versions including automatic update of the application software.

If the keep alive bit of a CANopen device is set the NMT master shall not issue the NMT services *Reset node* and *Reset communication* for this CANopen device.

NOTE 1 Such situations may occur, in case the NMT master encounters a failure with a subsequent re-start, e.g. power fail.

Software version control may be used in systems where the version of the application software running on the CANopen device is checked for correctness. The process check and update software version may be used to automatically download the most current version of the application software. CANopen devices that detect an error in their application software, for example by calculating a checksum during startup, may force a download of the most current application software by responding with wrong version information.

Both or only one of the two features may be implemented.

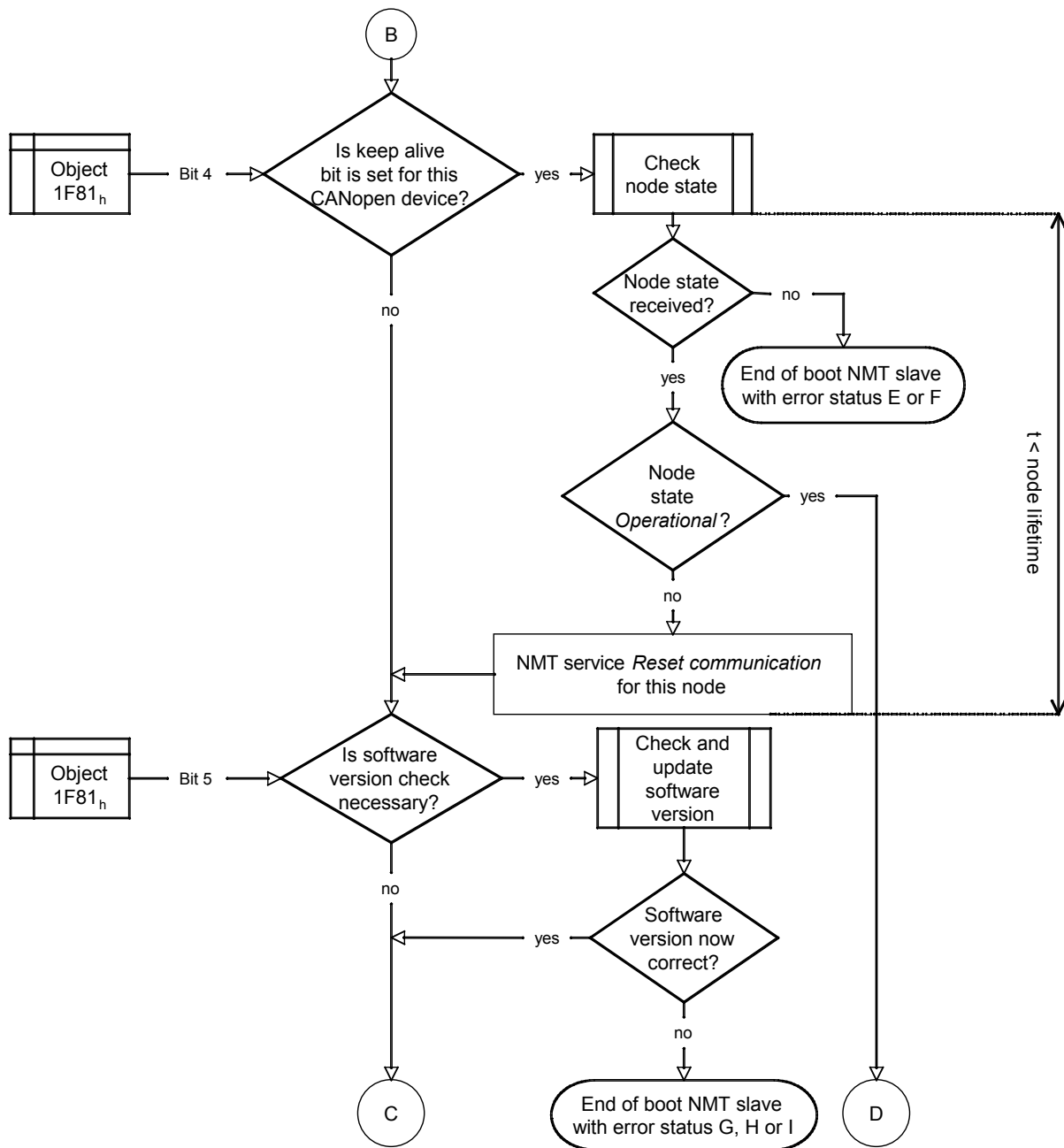


Figure 17 – Boot NMT slave, part 2

- f) Bit 4 of object 1F81_h is used to decide whether keep alive is requested.

In case keep alive is requested the NMT master shall request the current NMT state of the NMT slave for which the *boot NMT slave* is performed. In case no current NMT state is received the process shall finish with error status. In case the current NMT state is *Operational* the process shall continue with D. In case the current NMT state is not the NMT state *Operational* the NMT master shall perform the NMT service *Reset communication* for that NMT slave.

The process check NMT state is defined in 9.5.2.

NOTE 2 In case the NMT slave supports Node and Life guarding only the NMT master takes care that the time from requesting the current NMT state from the NMT slave to issuing the NMT service *Reset communication* is less than the life time as set in the NMT slave.

- g) Bit 5 of object 1F81_h is used to decide whether the application software verification shall be performed.

In case application software verification is requested, the process check and update software version shall be performed.

In case the software version stays incorrect the process Boot NMT slave shall end with error status.

The process *boot NMT slave* continues in part 3 (see Figure 18) and is mandatory.

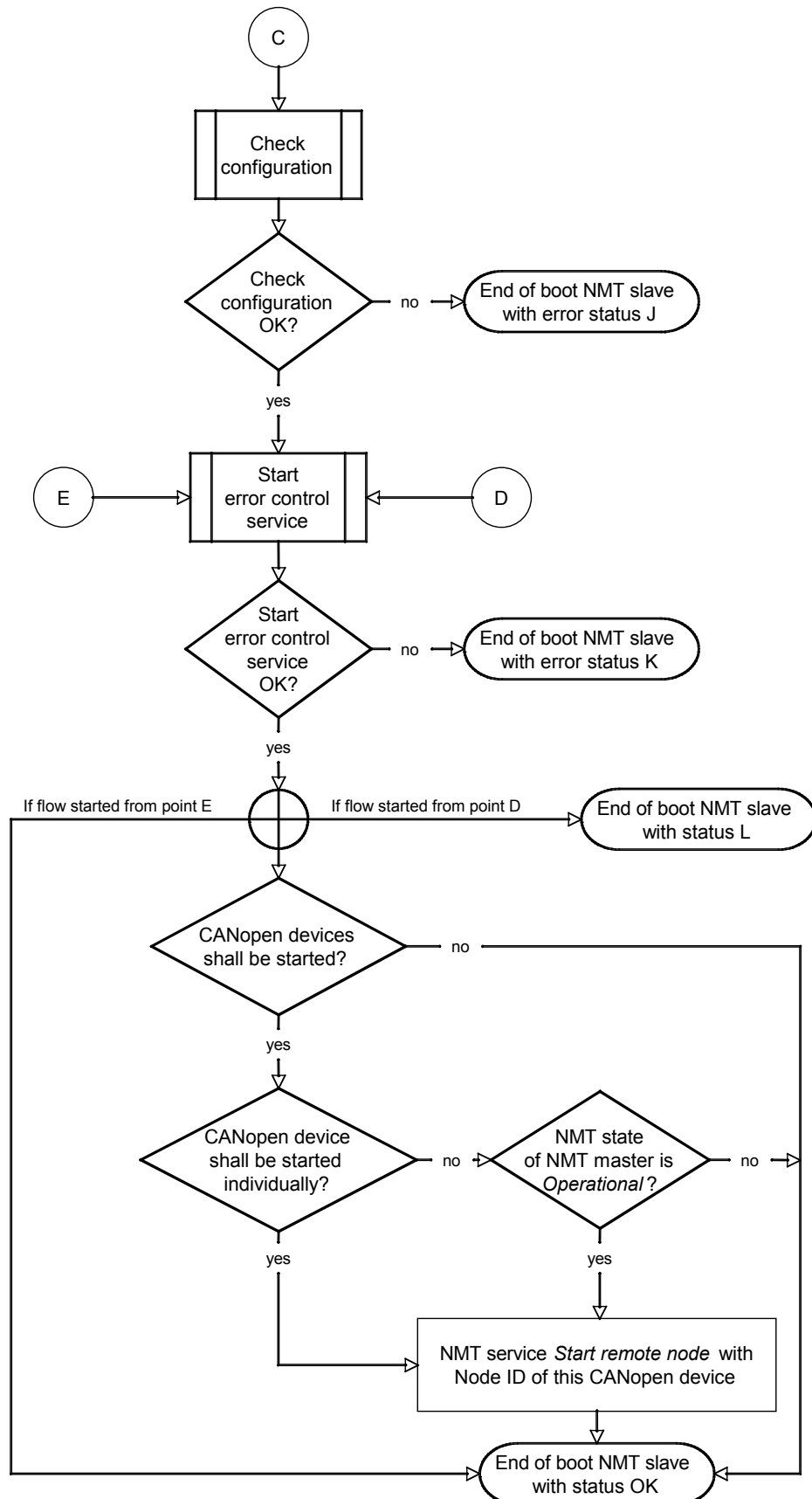


Figure 18 – Boot NMT slave, part 3

- h) The process check configuration (see 9.5.1) shall be performed.
 - i) In case the process check configuration finished unsuccessfully the process *boot NMT slave* shall end with an error status.
 - j) The process *start error control* (see 9.6.1) shall be performed.
 - k) In case the process *start error control* finished unsuccessfully the process *boot NMT slave* shall end with an error status.
 - l) In case the NMT slave is in NMT state *Operational* the process *boot NMT slave* shall finish successfully.
 - m) Bit 3 of object 1F80_h is used to decide whether the NMT master shall execute the NMT service *Start remote node*.
 - n) In case the NMT master shall not execute the NMT service *Start remote node* the process *boot NMT slave* shall end successfully with status.
 - o) Bit 1 of object 1F80_h is used to decide upon whether the NMT master shall execute the NMT service *Start remote node* with CANopen Node-ID set to 0 or for each NMT slave in the network individually.
 - p) In case the NMT master shall execute the NMT service *Start remote node* with CANopen Node-ID set to 0 and the NMT master is not in NMT state *Operational* the process *boot NMT slave* shall finish successfully with status OK.
- NOTE 3 If the NMT master is in NMT state *Operational* it is regarded as an indication that the initial NMT startup has been completed prior to the startup of this NMT slave.
- q) In case the NMT master executes the NMT service *Start remote node* for each NMT slave in the network individually or in case the NMT master executes the NMT service *Start remote node* with CANopen Node-ID set to 0 and the NMT master is in NMT state *Operational* the NMT master shall execute the NMT service *Start remote node* with CANopen Node-ID set to the appropriate value.
 - r) The process *boot NMT slave* shall finish successfully with status.

9.5.1 Check configuration

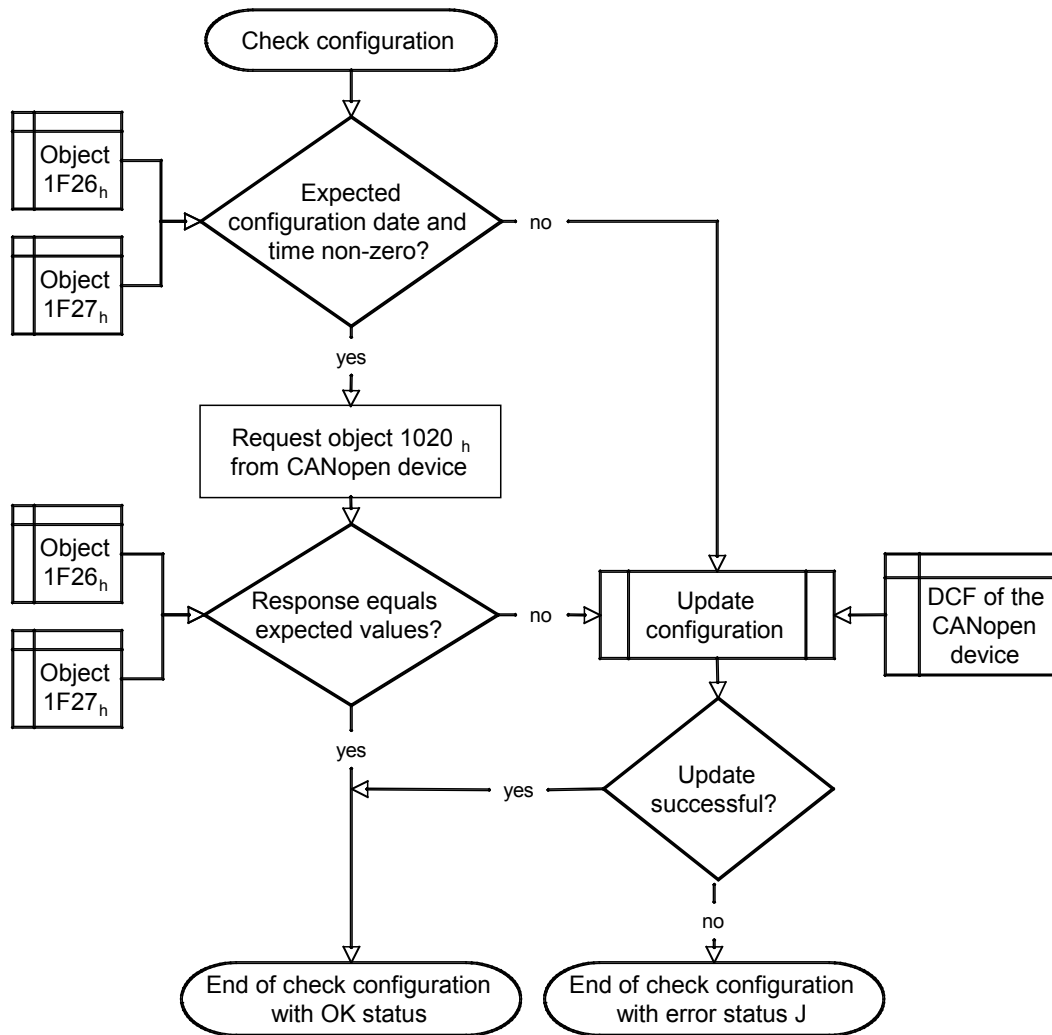


Figure 19 – Check configuration

The process *check configuration* as shown in Figure 19 consists of the following steps:

- The corresponding entries (sub-index shall equal CANopen Node-ID) of object 1F26_h and object 1F27_h are used to decide whether the CANopen devices configuration is verified. If the values are equal to 0 the process shall go to step d). If the values are not equal to 0 the process shall go to step b).
- Object 1020_h shall be requested from the CANopen device.
- The values of the corresponding entries (sub-index shall equal CANopen Node-ID) of object 1F26_h and object 1F27_h shall be compared to the values of the received object 1020_h. If the values are equivalent the process shall finish with status OK. If the values are not equivalent the process shall go to step d).
- The configuration shall be updated on the CANopen device.
Depending on the settings of 1F81_h and 1F8A_h the configured restore operation shall be executed. Only after the restore operation the NMT service reset communication or NMT service reset node for the corresponding NMT slave shall be issued.
The configuration may be derived from any kind of DCF provided for that CANopen device. The DCF may be read from a local file system, object 1F20_h, or object 1F22_h on the configuration manager.
- If the download is successful then the process shall finish with status OK. If the download is not successful then the process shall finish with error status.

The process *check configuration* may be skipped if the keep-alive is enabled for that CANopen device and the CANopen device is in the NMT state *Operational*. If this occurs the process boot NMT slave shall finish with error status (L — "NMT slave was initially operational").

9.5.2 Check NMT state

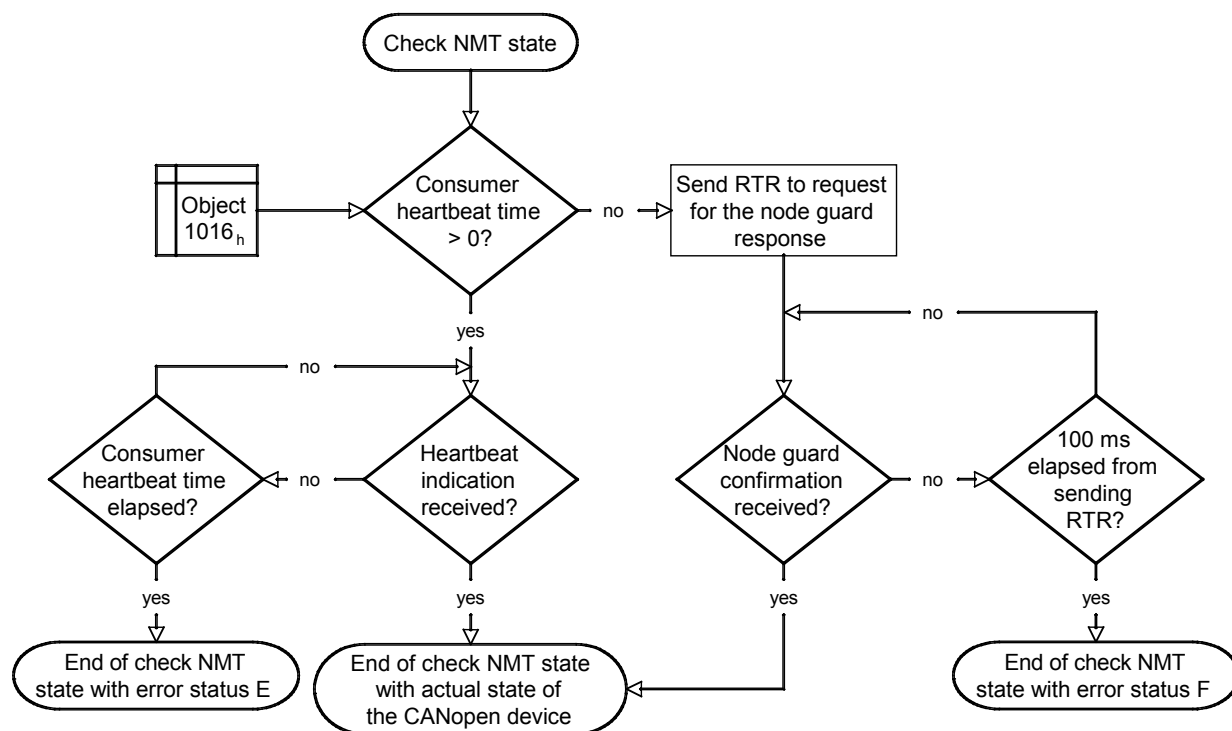


Figure 20 – Check NMT state¹

The process *check NMT state* as shown in Figure 20 consists of the following steps:

- Object 1016_h (see EN 50325-4) shall be used to decide whether the CANopen device is set up for heartbeat.
- In case the CANopen device is set up for heartbeat the NMT master shall check if it received a heartbeat indication in time (heartbeat consumer time is not elapsed). If the heartbeat indication is not received in time the process shall end with an error status. If the heartbeat indication is received the process shall end with the actual NMT state of the CANopen device that is checked.
- In case the CANopen device that is checked is not set up for heartbeat it is assumed that the CANopen device is set up for CANopen device guarding. In this case the NMT master shall request the actual NMT state of the CANopen device that is checked, the NMT service CANopen device guarding shall be used. In case no confirmation is received and more than 100 ms are elapsed the process shall end with an error status. In case a confirmation is received the process shall end with the actual NMT state of the CANopen device that is checked.

9.5.3 NMT flying master start up

The definition of the flying NMT master is not in the scope of this standard.

NOTE Definitions for the flying NMT master functionality are provided in CiA 302.

¹ Required, if bit 4 of object 1F81_h is set (keep alive supported).

9.5.4 Error status

The errors as shown in Table 14 shall be signaled within the NMT master during startup.

Table 14 – Error status

Error status	Description
A	The CANopen device is not listed in object 1F81 _h .
B	No response received for upload request of object 1000 _h .
C	Value of object 1000 _h from CANopen device is different to value in object 1F84 _h (Device type).
D	Value of object 1018 _h sub-index 01 _h from CANopen device is different to value in object 1F85 _h (Vendor ID).
E	Heartbeat event. No heartbeat message received from CANopen device.
F	Node guarding event. No confirmation for guarding request received from CANopen device.
G	Object 1F53 _h or object 1F54 _h not configured for that CANopen device.
H	Values of object 1F52 _h from that CANopen device are different to value of object 1F53 _h or value of object 1F54 _h and bit 6 of object 1F81 _h is not set.
I	Values of object 1F52 _h from that CANopen device are different to value of object 1F53 _h or value of object 1F54 _h and download failed.
J	Configuration download failed.
K	Heartbeat event during start error control service. No heartbeat message received from CANopen device during start error control service.
L	NMT slave was initially operational. (CANopen manager may resume operation with other CANopen devices)
M	Value of object 1018 _h sub-index 02 _h from CANopen device is different to value in object 1F86 _h (Product code).
N	Value of object 1018 _h sub-index 03 _h from CANopen device is different to value in object 1F87 _h (Revision number).
O	Value of object 1018 _h sub-index 04 _h from CANopen device is different to value in object 1F88 _h (Serial number).

9.6 Error control

9.6.1 Start error control

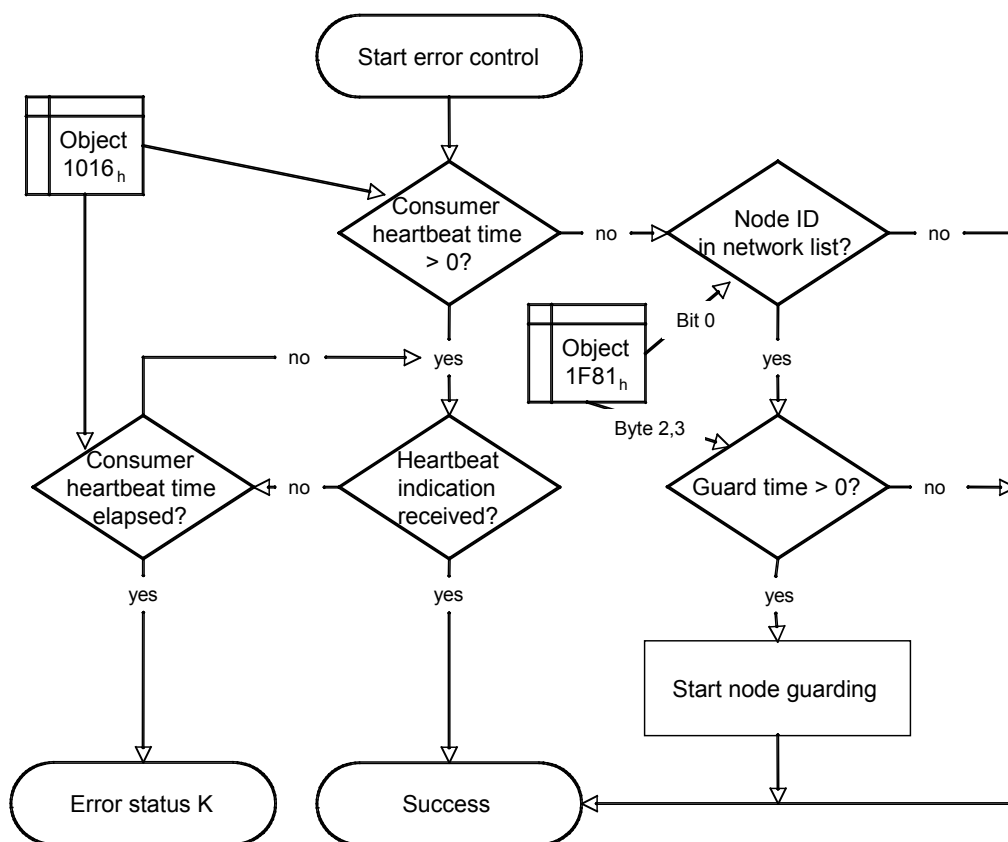


Figure 21 – Start error control

The process *start error control* as shown in Figure 21 consists of the following steps:

- Object 1016_h (see EN 50325-4) shall be checked to decide whether the CANopen device is set up for heartbeat.
- In case the CANopen device is set up for heartbeat and no heartbeat indication is received in time the process shall end with an error status. The timer for timeout checking shall be started immediately with the process itself.
- In case the CANopen device is set up for heartbeat and a heartbeat indication is received in time the process shall end successfully.
- In case the CANopen device is not set up for heartbeat bit 0 of object 1F81_h is used to decide whether the CANopen device shall be guarded.
- In case the CANopen device is not in the network list the process shall end successfully.
- In case the CANopen device is in the network list the value of byte 2 and 3 of object 1F81_h is used to decide whether the CANopen device shall be guarded.
- In case the value is greater than 0 guarding shall be started for that CANopen device and the process shall end successfully.
- In case the value equals 0 the process shall end successfully.

9.6.2 Error handler

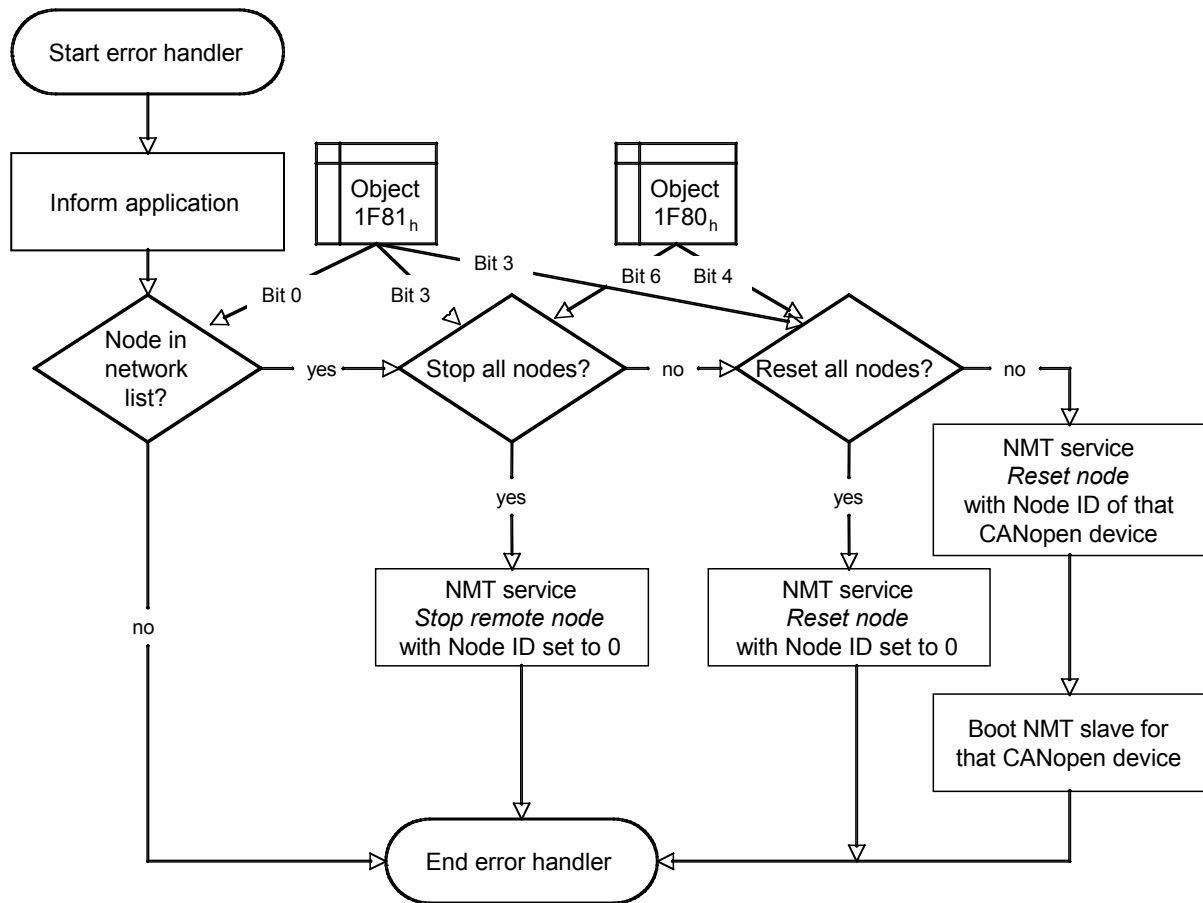


Figure 22 – Error handler

The process *error handler* as defined in Figure 22 shall be initiated any time an NMT error event occurs.

9.6.3 Bootup handler

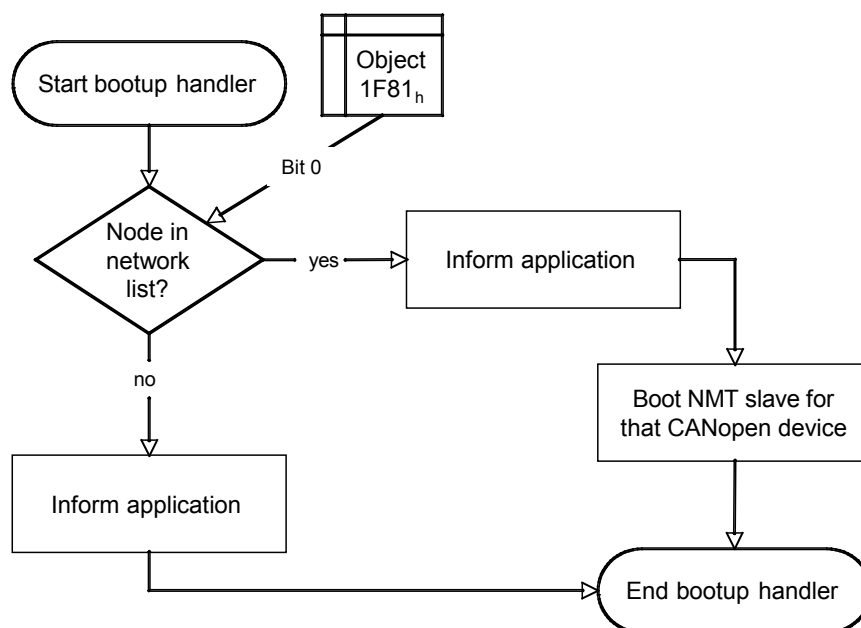


Figure 23 – Bootup handler

The process *bootup handler* as defined in Figure 23 shall be initiated any time a bootup event occurs (see EN 50325-4).

9.7 Additional NMT master services and protocols

The definition of additional NMT master services and protocols is not in the scope of this specification.

NOTE Additional NMT master services and protocols such as NMT master negotiation or NMT master detection services are provided in CiA 302.

9.8 Object dictionary entries

9.8.1 Object 1020_h: Verify configuration

This object shall indicate the downloaded configuration date and time. If a CANopen device supports the saving of parameters in non-volatile memory, a network configuration tool or a CANopen manager uses this object to verify the configuration after a CANopen device reset and to check if a reconfiguration is necessary. The configuration tool stores the date and time in that object and stores the same values in the DCF. Now the configuration tool lets the CANopen device save its configuration by writing to index 1010_h sub-index 01_h the signature "save". After a reset the CANopen device shall restore the last configuration and the signature automatically or by request. If any other command changes boot-up configuration values, the CANopen device shall reset the object Verify Configuration to 0.

The Configuration Manager compares signature and configuration with the value from the DCF and decides if a reconfiguration is necessary or not.

NOTE The usage of this object allows a significant speed-up of the boot-up process. If it is used, the system integrator considers that an user changes a configuration value and afterwards activate the command store configuration 1010_h without changing the value of 1020_h. So the system integrator ensures a 100 % consequent usage of this feature.

Sub-index 01_h (configuration date) shall contain the number of days since January 1, 1984. Sub-index 02_h (configuration time) shall be the number of ms after midnight. Table 15 and Table 16 provide the object description and the entry description.

Table 15 – Object description

Attribute	Value
Index	1020 _h
Name	Verify configuration
Object code	ARRAY
Data type	UNSIGNED32
Category	Optional

Table 16 – Entry description

Attribute	Value
Sub-index	00 _h
Description	Highest sub-index supported
Entry category	Mandatory
Access	const
PDO mapping	No
Value range	02 _h
Default value	02 _h
Sub-index	01 _h
Description	Configuration date
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	UNSIGNED32
Default value	Manufacturer-specific
to	
Sub-index	02 _h
Description	Configuration time
Entry category	Mandatory
Access	rw
PDO mapping	No
Value range	UNSIGNED32
Default value	Manufacturer-specific

9.8.2 Object 102A_h: NMT inhibit time

This object shall indicate the configured inhibit time between two subsequent NMT messages. The outstanding NMT services shall be queued and shall be issued in order of their occurrence respecting the configured inhibit time. Table 17 and Table 18 define the object description and the entry description.

The value shall be given in multiples of 100 µs. The value 0 shall disable the inhibit time.