IEC 61158-4-13

Edition 2.0 2014-08

INTERNATIONAL STANDARD

NORME INTERNATIONALE

Industrial communication networks – Fieldbus specifications – Part 4-13: Data-link layer protocol specification – Type 13 elements

Réseaux de communication industriels – Spécifications des bus de terrain – Partie 4-13: Spécification du protocole de la couche liaison de données – Éléments de type 13

IEC 61158-4-13:2014-08(en-fr)

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.

IEC Catalogue - webstore.iec.ch/catalogue
The stand-alone application for consulting the entire bibliographical information on IEC International Standards, Technical Specifications, Technical Reports and other documents. Available for PC, Mac OS, Android Tablets and iPad.

IEC publications search - www.iec.ch/searchpub
The advanced search enables 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 It also gives information on projects, replaced and withdrawn publications.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 online 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 14 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

IEC Glossary - std.iec.ch/glossary
More than 55 000 electrotechnical terminology entries in English and French extracted from the Terms and Definitions clause of IEC publications issued since 2002. Some entries have been collected from earlier publications of IEC TC 37, 77, 86 and CISPR.

IEC 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 l'IEC
La Commission Electrotechnique Internationale (IEC) 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 IEC
Le contenu technique des publications IEC est constamment revu. Veuillez vous assurer que vous possédez l'édition la plus récente, un corrigendum ou amendement peut avoir été publié.

Catalogue IEC - webstore.iec.ch/catalogue
Application autonome pour consulter tous les renseignements bibliographiques sur les Normes internationales, Spécifications techniques, Rapports techniques et autres documents de l'IEC. Disponible pour PC, Mac OS, tablettes Android et iPad.

Recherche de publications IEC - www.iec.ch/searchpub
La recherche avancée permet de trouver des publications IEC 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.

IEC Just Published - webstore.iec.ch/justpublished
Restez informé sur les nouvelles publications IEC. 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 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 14 langues additionnelles. Egalement appelé Vocabulaire Electrotechnique International (IEV) en ligne.

Glossaire IEC - std.iec.ch/glossary
Plus de 55 000 entrées terminologiques électrotechniques, en anglais et en français, extraites des articles Termes et Définitions des publications IEC parues depuis 2002. Plus certaines entrées antérieures extraites des publications des CE 37, 77, 86 et CISPR de l'IEC.

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.

IEC 61158-4-13

Edition 2.0 2014-08

INTERNATIONAL STANDARD

NORME INTERNATIONALE

Industrial communication networks – Fieldbus specifications – Part 4-13: Data-link layer protocol specification – Type 13 elements

Réseaux de communication industriels – Spécifications des bus de terrain – Partie 4-13: Spécification du protocole de la couche liaison de données – Éléments de type 13

INTERNATIONAL ELECTROTECHNICAL COMMISSION

COMMISSION ELECTROTECHNIQUE INTERNATIONALE

PRICE CODE XC
CODE PRIX

ICS 25.040.40; 35.100.20; 35.110

ISBN 978-2-8322-1725-2

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

INDUSTRIAL COMMUNICATION NETWORKS – FIELDBUS SPECIFICATIONS –

Part 4-13: Data-link layer protocol specification – Type 13 elements

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.

Attention is drawn to the fact that the use of the associated protocol type is restricted by its intellectual-property-right holders. In all cases, the commitment to limited release of intellectual-property-rights made by the holders of those rights permits a layer protocol type to be used with other layer protocols of the same type, or in other type combinations explicitly authorized by its intellectual-property-right holders.

NOTE Combinations of protocol types are specified in IEC 61784-1 and IEC 61784-2.

International Standard IEC 61158-4-13 has been prepared by subcommittee 65C: Industrial networks, of IEC technical committee 65: Industrial-process measurement, control and automation.

This second edition cancels and replaces the first edition published in 2007. This edition constitutes a technical revision. The main changes with respect to the previous edition are listed below:

- addition of a new communication class,
- corrections and
- editorial improvements.

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

FDIS	Report on voting
65C/762/FDIS	65C/772/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 ISO/IEC Directives, Part 2.

A list of all the parts of the IEC 61158 series, under the general title *Industrial communication networks – Fieldbus specifications*, can be found on the IEC web site.

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

This part of IEC 61158 is one of a series produced to facilitate the interconnection of automation system components. It is related to other standards in the set as defined by the “three-layer” fieldbus reference model described in IEC 61158-1.

The data-link protocol provides the data-link service by making use of the services available from the physical layer. The primary aim of this standard is to provide a set of rules for communication expressed in terms of the procedures to be carried out by peer data-link entities (DLEs) at the time of communication. These rules for communication are intended to provide a sound basis for development in order to serve a variety of purposes:

a) as a guide for implementors and designers;

b) for use in the testing and procurement of equipment;

c) as part of an agreement for the admittance of systems into the open systems environment;

d) as a refinement to the understanding of time-critical communications within OSI.

This standard is concerned, in particular, with the communication and interworking of sensors, effectors and other automation devices. By using this standard together with other standards positioned within the OSI or fieldbus reference models, otherwise incompatible systems may work together in any combination.

INDUSTRIAL COMMUNICATION NETWORKS – FIELDBUS SPECIFICATIONS –

Part 4-13: Data-link layer protocol specification – Type 13 elements

1 Scope

1.1 General

The data-link layer provides basic time-critical messaging communications between devices in an automation environment.

This protocol provides communication opportunities to all participating data-link entities

a) in a synchronously-starting cyclic manner, according to a pre-established schedule, and

b) in a cyclic or acyclic asynchronous manner, as requested each cycle by each of those data-link entities.

Thus this protocol can be characterized as one which provides cyclic and acyclic access asynchronously but with a synchronous restart of each cycle.

1.2 Specifications

This standard specifies

a) procedures for the timely transfer of data and control information from one data-link user entity to a peer user entity, and among the data-link entities forming the distributed data-link service provider;

b) procedures for giving communications opportunities to all participating DL-entities, sequentially and in a cyclic manner for deterministic and synchronized transfer at cyclic intervals up to one millisecond;

c) procedures for giving communication opportunities available for time-critical data transmission together with non-time-critical data transmission without prejudice to the time-critical data transmission;

d) procedures for giving cyclic and acyclic communication opportunities for time-critical data transmission with prioritized access;

e) procedures for giving communication opportunities based on ISO/IEC 8802-3 medium access control, with provisions for nodes to be added or removed during normal operation;

f) the structure of the fieldbus DLPDUs used for the transfer of data and control information by the protocol of this standard, and their representation as physical interface data units.

1.3 Procedures

The procedures are defined in terms of

a) the interactions between peer DL-entities (DLEs) through the exchange of fieldbus DLPDUs;

b) the interactions between a DL-service (DLS) provider and a DLS-user in the same system through the exchange of DLS primitives;

c) the interactions between a DLS-provider and a Ph-service provider in the same system through the exchange of Ph-service primitives.

1.4 Applicability

These procedures are applicable to instances of communication between systems which support time-critical communications services within the data-link layer of the OSI or fieldbus reference models, and which require the ability to interconnect in an open systems interconnection environment.

Profiles provide a simple multi-attribute means of summarizing an implementation's capabilities, and thus its applicability to various time-critical communications needs.

1.5 Conformance

This standard also specifies conformance requirements for systems implementing these procedures. This standard does not contain tests to demonstrate compliance with such requirements.

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 61588, *Precision clock synchronization protocol for networked measurement and control systems*

ISO/IEC 7498-1, *Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model*

ISO/IEC 7498-3, *Information technology – Open Systems Interconnection – Basic Reference Model: Naming and addressing*

ISO/IEC 8802-3:2000, *Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications*

ISO/IEC 10731, *Information technology – Open Systems Interconnection – Basic Reference Model – Conventions for the definition of OSI services*

3 Terms, definitions, symbols, abbreviations and conventions

For the purposes of this document, the following terms, definitions, symbols, abbreviations and conventions apply.

3.1 Reference model terms and definitions

This standard is based in part on the concepts developed in ISO/IEC 7498-1 and ISO/IEC 7498-3, and makes use of the following terms defined therein:

3.1.1 DL-address [7498-3]

3.1.2 DL-address-mapping [7498-1]

3.1.3 called-DL-address [7498-3]

3.1.4 calling-DL-address [7498-3]

3.1.5 centralized multi-end-point-connection [7498-1]

3.1.6 DL-connection [7498-1]

3.1.7 DL-connection-end-point [7498-1]

3.1.8 DL-connection-end-point-identifier [7498-1]

3.1.9 DL-connection-mode transmission [7498-1]

3.1.10 DL-connectionless-mode transmission [7498-1]

3.1.11 correspondent (N)-entities [7498-1]
correspondent DL-entities (N=2)
correspondent Ph-entities (N=1)

3.1.12 DL-duplex-transmission [7498-1]

3.1.13 (N)-entity [7498-1]
DL-entity (N=2)
Ph-entity (N=1)

3.1.14 DL-facility [7498-1]

3.1.15 flow control [7498-1]

3.1.16 (N)-layer [7498-1]
DL-layer (N=2)
Ph-layer (N=1)

3.1.17 layer-management [7498-1]

3.1.18 DL-local-view [7498-3]

3.1.19 DL-name [7498-3]

3.1.20 naming-(addressing)-domain [7498-3]

3.1.21 peer-entities [7498-1]

3.1.22 primitive name [7498-3]

3.1.23 DL-protocol [7498-1]

3.1.24 DL-protocol-connection-identifier [7498-1]

3.1.25 DL-protocol-data-unit [7498-1]

3.1.26 DL-relay [7498-1]

3.1.27 reset [7498-1]

3.1.28 responding-DL-address [7498-3]

3.1.29 routing [7498-1]

3.1.30 segmenting [7498-1]

3.1.31 (N)-service [7498-1]
DL-service (N=2)
Ph-service (N=1)

3.1.32 (N)-service-access-point [7498-1]
DL-service-access-point (N=2)
Ph-service-access-point (N=1)

3.1.33 DL-service-access-point-address [7498-3]

3.1.34 DL-service-connection-identifier [7498-1]

3.1.35 DL-service-data-unit [7498-1]

3.1.36 DL-simplex-transmission [7498-1]

3.1.37 DL-subsystem [7498-1]

3.1.38 systems-management [7498-1]

3.1.39 DLS-user-data [7498-1]

3.2 Service convention terms and definitions

This standard also makes use of the following terms defined in ISO/IEC 10731 as they apply to the data-link layer:

3.2.1 acceptor

3.2.2 asymmetrical service

3.2.3 confirm (primitive);
requestor.deliver (primitive)

3.2.4 deliver (primitive)

3.2.5 DL-confirmed-facility

3.2.6 DL-facility

3.2.7 DL-local-view

3.2.8 DL-mandatory-facility

3.2.9 DL-non-confirmed-facility

3.2.10 DL-provider-initiated-facility

3.2.11 DL-provider-optional-facility

3.2.12 DL-service-primitive;
primitive

3.2.13 DL-service-provider

3.2.14 DL-service-user

3.2.15 DLS-user-optional-facility

3.2.16 indication (primitive);
acceptor.deliver (primitive)

3.2.17 multi-peer

3.2.18 request (primitive);
requestor.submit (primitive)

3.2.19 requestor

3.2.20 response (primitive);
acceptor.submit (primitive)

3.2.21 submit (primitive)

3.2.22 symmetrical service

3.3 Data-link service terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.3.1
async-only CN
CN that is accessed only by polling

3.3.2
asynchronous period
second part of the Type 13 cycle, starting with a start of asynchronous (SoA) frame

3.3.3
basic Ethernet mode
mode that provides legacy Ethernet communication

3.3.4
continuous-time-triggered
communication class where isochronous communication takes place every cycle

Note 1 to entry: The data sent from MN to various CNs are packed into a PollResponse. No PollRequest to these CNs is necessary. The CNs send their PollResponse time triggered. (An alternative to continuous).

Note 2 to entry: There are three node classes: *continuous, multiplexed* and *continuous-time-triggered*. Each node is a member of exactly one of these classes.

3.3.5
continuous
communication class where isochronous communication takes place every cycle (the opposite to multiplexed)

Note 1 to entry: There are three node classes: *continuous, multiplexed* and *continuous-time-triggered*. Each node is a member of exactly one of these classes.

3.3.6
controlled node (CN)
network node without the ability to manage the SCNM mechanism

3.3.7
cycle state machine
state machine that controls the data-link layer cycle and is itself controlled by the NMT state machine which determines the current operating mode

3.3.8
cycle time
time between two consecutive start of cyclic (SoC) frames

Note 1 to entry: The Cycle Time includes the time for data transmission and some idle time before the beginning of the next cycle.

3.3.9
DLCEP-address
DL-address which designates either

a) one peer DL-connection-end-point, or

b) one multi-peer publisher DL-connection-end-point and implicitly the corresponding set of subscriber DL-connection-end-points where each DL-connection-end-point exists within a distinct DLSAP and is associated with a corresponding distinct DLSAP-address

3.3.10
DL-segment,
link,
local link
single DL-subnetwork in which any of the connected DLEs may communicate directly, without any intervening DL-relaying, whenever all of those DLEs that are participating in an instance of communication are simultaneously attentive to the DL-subnetwork during the period(s) of attempted communication

3.3.11
DLSAP
distinctive point at which DL-services are provided by a single DL-entity to a single higher-layer entity

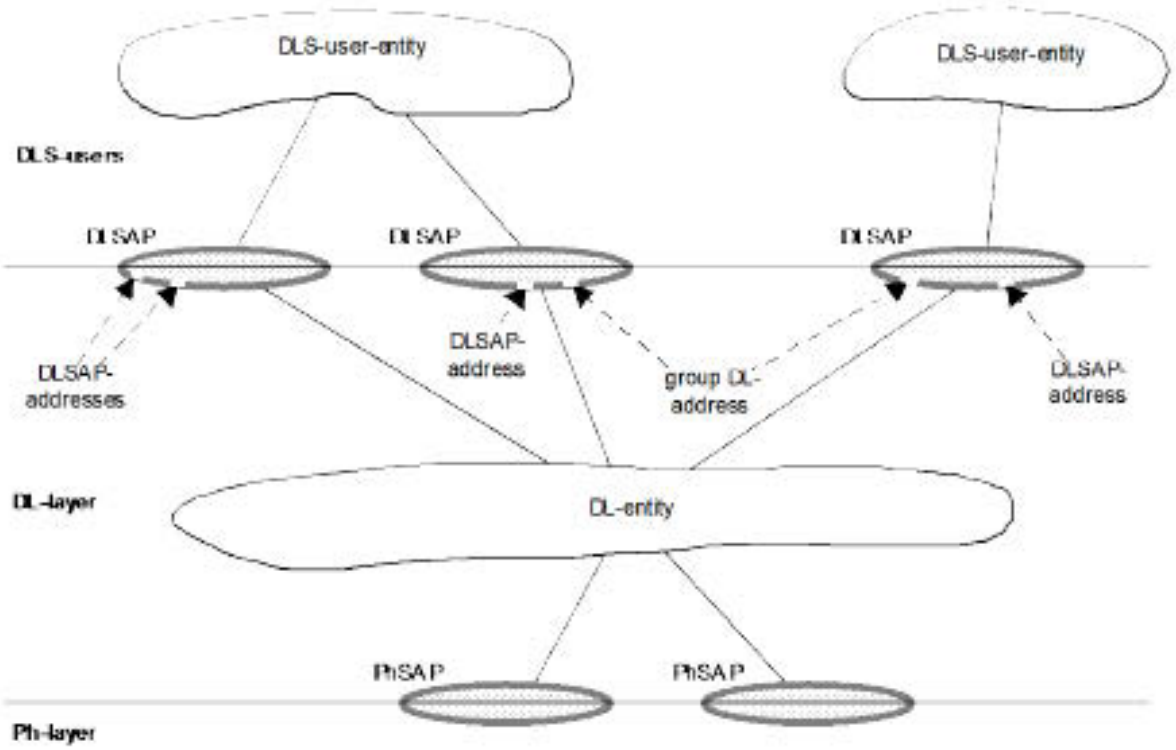
Note 1 to entry: This definition, derived from ISO/IEC 7498-1, is repeated here to facilitate understanding of the critical distinction between DLSAPs and their DL-addresses.

3.3.12
DL(SAP)-address
either an individual DLSAP-address, designating a single DLSAP of a single DLS-user, or a group DL-address potentially designating multiple DLSAPs, each of a single DLS-user

Note 1 to entry: This terminology is chosen because ISO/IEC 7498-3 does not permit the use of the term DLSAP-address to designate more than a single DLSAP at a single DLS-user.

3.3.13
(individual) DLSAP-address
DL-address that designates only one DLSAP within the extended link

Note 1 to entry: A single DL-entity may have multiple DLSAP-addresses associated with a single DLSAP (see Figure 1).



NOTE 1 DLSAPs and PhSAPs are depicted as ovals spanning the boundary between two adjacent layers.

NOTE 2 DL-addresses are depicted as designating small gaps (points of access) in the DLL portion of a DLSAP.

NOTE 3 A single DL-entity may have multiple DLSAP-addresses and group DL-addresses associated with a single DLSAP.

Figure 1 – Relationships of DLSAPs, DLSAP-addresses and group DL-addresses

3.3.14
Type 13 fieldbus cycle
fixed time interval consecutively repeated which is organized by the MN

3.3.15
Type 13 fieldbus node ID
unique number within one Type 13 network used to address a Type 13 fieldbus node

3.3.16
frame
denigrated synonym for DLPDU

3.3.17
isochronous data
data which is transmitted every cycle (or every nth cycle in case of multiplexed isochronous data)

3.3.18
isochronous period
period within each cycle that offers deterministic operation through being reserved for the exchange of (continuous or multiplexed) isochronous data

3.3.19
legacy Ethernet
Ethernet as standardized in ISO/IEC 8802-3 (non-deterministic operation in non-time-critical environments)

3.3.20
managing node
node that can manage the SCNM mechanism

3.3.21
multiplexed
communication class where cyclic communication takes place in such a way that m nodes are served in s cycles (an alternative to continuous)

3.3.22
multiplexed timeslot
timeslot assigned to multiplexed isochronous data and shared among multiple nodes

3.3.23
multipoint connection
connection from one node to many nodes

Note 1 to entry: Multipoint connection allows data transfer from a single publisher to many subscriber nodes.

3.3.24
multi-peer DLC
centralized multi-end-point DL-connection offering DL-duplex-transmission between a single distinguished DLS-user known as the publisher or publishing DLS-user, and a set of peer but undistinguished DLS-users known collectively as the subscribers or subscribing DLS-users, where the publishing DLS-user can send to the subscribing DLS-users as a group (but not individually), and the subscribing DLS-users can send to the publishing DLS-user (but not to each other)

3.3.25
NetTime
clock time of the MN as distributed to all CNs within the SoC frame

3.3.26
network management
management functions and services that perform network initialization, configuration and error handling

3.3.27
node
single DL-entity as it appears on one local link

3.3.28
PollRequest
frame which is used in the isochronous part of a communications cycle

3.3.29
PollResponse
frame which is used in the isochronous part of a communications cycle to respond to a PollRequest frame

3.3.30
process data object
object for isochronous data exchange between nodes

3.3.31
protocol
convention about the data formats, time sequences, and error correction in the data exchange of communication systems

3.3.32
receiving DLS-user
DL-service user that acts as a recipient of DLS-user-data

Note 1 to entry: A DL-service user can be concurrently both a sending and receiving DLS-user.

3.3.33
sending DLS-user
DL-service user that acts as a source of DLS-user-data

3.3.34
service data object
object for asynchronous data exchange between nodes

3.3.35
slot communication network management
mechanism which ensures that there are no collisions during physical network access of any of the networked nodes, thus providing deterministic communication via legacy Ethernet

3.4 Symbols and abbreviations

ASnd	Asynchronous send (Type 13 frame type)
ASS	Asynchronous slot scheduler
ATC	Asynchronous transmissionTX/RX control
CN	Controlled node
CSM	Cycle state machine
DL-	Data-link layer (as a prefix)
DLC	DL-connection
DLCEP	DL-connection-end-point
DLE	DL-entity (the local active instance of the data-link layer)
DLL	DL-layer
DLPDU	DL-protocol-data-unit
DLM	DL-management
DLMS	DL-management service
DLS	DL-service
DLSAP	DL-service-access-point
DLSDU	DL-service-data-unit
ES	Exception signaling
FIFO	First-in first-out (queuing method)
ITC	Isochronous transmission RX/TX control
MAC	Media access control
MN	Managing node
NMT	Network management
NS	NMT signaling
OSI	Open systems interconnection
PDO	Process data object
PDU	Protocol data unit
Ph-	Physical layer (as a prefix)
PhL	Ph-layer
PhPDU	Ph-protocol-data-unit
PhS	Ph-service

PReq	PollRequest (Type 13 frame type)
PRes	PollResponse (Type 13 frame type)
SCNM	Slot communication network management
SDO	Service data object
SoA	Start of asynchronous (Type 13 frame type)
SoC	Start of cyclic (Type 13 frame type)
UDT	Unspecified-data transfer

3.5 Common conventions

This standard uses the descriptive conventions given in ISO/IEC 10731.

The service model, service primitives, and time-sequence diagrams used are entirely abstract descriptions; they do not represent a specification for implementation.

Service primitives, used to represent service user/service provider interactions (see ISO/IEC 10731), convey parameters that indicate information available in the user/provider interaction.

This standard uses a tabular format to describe the component parameters of the DLS primitives. The parameters that apply to each group of DLS primitives are set out in tables throughout the remainder of this standard. Each table consists of up to six columns, containing the name of the service parameter, and a column each for those primitives and parameter-transfer directions used by the DLS:

- The request primitive's input parameters;
- The request primitive's output parameters;
- The indication primitive's output parameters;
- The response primitive's input parameters; and
- The confirm primitive's output parameters.

NOTE The request, indication, response and confirm primitives are also known as requestor.submit, acceptor.deliver, acceptor.submit, and requestor.deliver primitives, respectively (see ISO/IEC 10731).

One parameter (or part of it) is listed in each row of each table. Under the appropriate service primitive columns, a code is used to specify the type of usage of the parameter on the primitive and parameter direction specified in the column:

M Parameter: mandatory for the primitive.

U Parameter: a User option, and may or may not be provided depending on the dynamic usage of the DLS-user. When not provided, a default value for the parameter is assumed.

C Parameter is conditional upon other parameters or upon the environment of the DLS-user.

(Blank) Parameter is never present.

Some entries are further qualified by items in brackets. These may be

a) a parameter-specific constraint

(=) indicates that the parameter is semantically equivalent to the parameter in the service primitive to its immediate left in the table.

b) an indication that some note applies to the entry

(n) indicates that the following note n contains additional information pertaining to the parameter and its use.

In any particular interface, not all parameters need be explicitly stated. Some may be implicitly associated with the DLSAP at which the primitive is issued.

In the diagrams which illustrate these interfaces, dashed lines indicate cause-and-effect or time-sequence relationships, and wavy lines indicate that events are roughly contemporaneous.

3.6 Additional conventions

In the diagrams which illustrate the DLS and DLM interfaces, dashed lines indicate cause-and-effect or time-sequence relationships between actions at different stations, while solid lines with arrows indicate cause-and-effect time-sequence relationships which occur within the DLE-provider at a single station.

The following notation, a shortened form of the primitive classes defined in 3.5, is used in the figures and tables.

req	request primitive
ind	indication primitive
cnf	confirm primitive (confirmation)
res	Response primitive

4 Overview of the DL-protocol

4.1 Overview

A Type 13 fieldbus extends Ethernet according to ISO/IEC 8802-3 with mechanisms to transfer data with predictable timing and precise synchronization to meet timing demands typical for high-performance automation and motion applications. It does not change basic principles of the Fast Ethernet Standard ISO/IEC 8802-3 but extends it towards real-time Ethernet. Thus it is possible to leverage and continue to use any standard Ethernet silicon, infrastructure component or test and measurement equipment like a network analyzer.

4.2 General description

4.2.1 General

Type 13 fieldbus provides mechanisms to achieve the following.

- Transmit time-critical data in precise isochronous cycles. Data exchange is based on a publish/subscribe relationship.
- Synchronize networked nodes with high accuracy.
- Transmit less time-critical data on request asynchronously. Asynchronous data communication can be used to transfer IP-based protocols like TCP or UDP and higher layer protocols such as HTTP, FTP,...

Type 13 fieldbus manages the network traffic in a way that there are dedicated time-slots for isochronous and asynchronous data (see Figure 2). It takes care that always only one networked device gains access to the network media. Thus transmission of isochronous and asynchronous data will never interfere and precise communication timing is guaranteed. The mechanism is called slot communication network management (SCNM). SCNM is managed by one particular networked device – the Managing Node (MN) – which includes the MN functionality. All other nodes are called controlled nodes (CN).

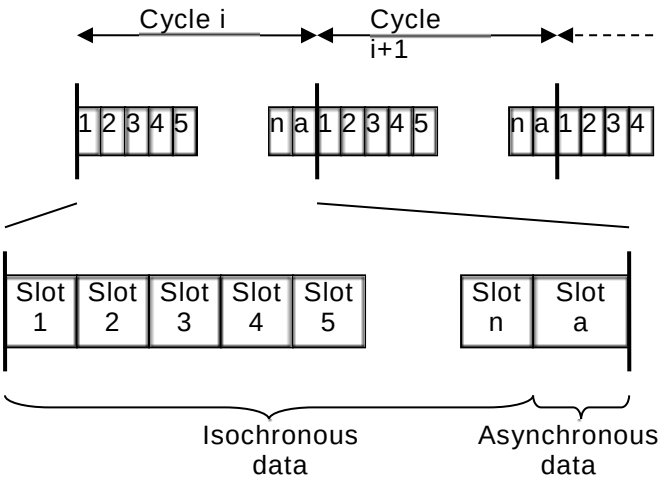


Figure 2 – Slot communication network management

Network access is managed by a master, the Type 13 fieldbus managing node (MN). A node is only being able to be granted the right to send data on the network via the MN. The central access rules preclude collisions; therefore a Type 13 network is therefore deterministic.

CNs are passive bus nodes; they only send when requested by the MN.

4.2.2 Overview of the medium access control and transmission protocol

Overview of the data frames flow on the medium is shown in Figure 3.

Data exchange between nodes occurs cyclically. It is required that the repetition occurs in a fixed interval, called Type 13 fieldbus cycle. The following time phases exist within one Type 13 fieldbus cycle.

- Isochronous phase
- Asynchronous phase
- Idle phase

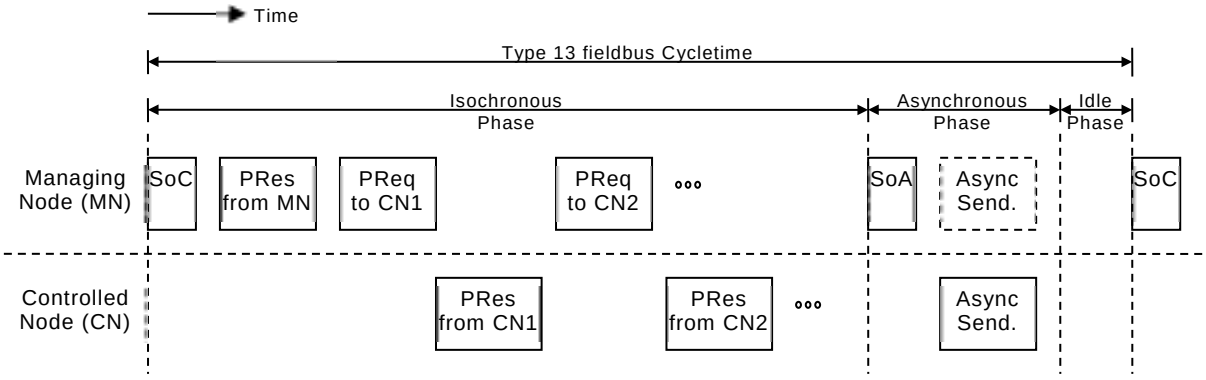


Figure 3 – Overall flow of data frames during one cycle

4.2.2.1 Isochronous phase

The DLL shall provide the opportunity of transferring data to each node in a sequential order and within a predetermined time period. At the beginning of a Type 13 fieldbus cycle, the MN shall send a SoC frame to all nodes via Ethernet multicast. The sending and receiving time of this frame is the basis for the common timing of all the nodes. Only the SoC frame is

generated on a periodic basis. The generation of all other frames is event controlled (with additional time monitoring per node).

The MN starts the isochronous data exchange after the SoC frame has been sent. A PReq frame is sent to every configured and active node. The accessed node responds by a PRes frame.

Both the PReq and the PRes frames shall transfer application data. The MN shall only send PReq data to one CN per frame. PReq transfer is dedicated to data relevant for the addressed CN only. In contrast, the PRes frame is received by all nodes. This makes communication relationships possible according to the producer/consumer model.

The PReq / PRes procedure is repeated for each configured and active isochronous CN. The MN may send a multicast PRes frame to all nodes. This frame is dedicated to transfer data relevant for groups of CNs. The order in which CNs are polled and the PRes frame of the MN is sent, is beyond the scope of this standard.

The isochronous phase is calculated from start of the SoC frame to start of the SoA frame.

The size and the length of the Type 13 fieldbus cycle is predominantly affected by the size of the isochronous phase. When configuring the Type 13 fieldbus cycle, the sum of the times required by the PReq / PRes accesses to each configured CN shall be taken into account. Use of the multiplexed or continuous-time-triggered access technology reduces the amount of time.

Continuous, continuous-time-triggered and multiplexed access operates in parallel during one Type 13 fieldbus cycle. The apportionment of the isochronous phase to continuous, continuous-time-triggered and multiplexed slots is configuration specific and not scope of this standard.

Although the multiplexed nodes are not processed in each cycle, they can monitor the entire data transfer of the continuous nodes.

In case of MN cycle loss, the multiplexed access sequence is continued on a per time base, after the cycle loss error phase is over.

4.2.2.2 Asynchronous phase

In the asynchronous phase of the Type 13 fieldbus cycle, access to the network is permitted to be granted to one CN or to the MN for the transfer of a single asynchronous message only.

There are two types of asynchronous frames available:

- the ASnd frame (Async Send);
- a Legacy Ethernet message.

In the SoA frame the MN assigns the permission to send an asynchronous frame to a CN or to itself. If no asynchronous message transmission request is pending at the MN scheduling queues, the MN shall issue a SoA without assignment of the right to send to any node. No ASnd frame and no legacy Ethernet message will follow the SoA frame in this case.

The SoA frame is the first frame in the asynchronous phase and is a signal to all CNs that all isochronous data have been exchanged during the isochronous phase.

In case of a Type 13 fieldbus ASnd frame, the user data may contain following data, depending upon MN request and application.

- Ident Data: The identification data of a CN.
- Status Data: The current status and detailed error information of a node.
- Sync Data: The synchronization data of a continuous-time-triggered CN.
- NMT Data: A NMTCommand from the MN to one or more CNs or a NMTRequest from a CN to the MN.
- Unspecified Data: Unspecified data communication may be used to transfer peer to peer communication with access to the object dictionary of a device, IP-based protocols like TCP or UDP and higher layer protocols such as HTTP, FTP, ...

In case of a legacy Ethernet message, the data could contain any legal Ethernet frame. The message is forwarded to the application without any further interpretation.

The asynchronous phase is calculated from the start of SoA to the end of the asynchronous response.

4.2.2.3 Idle phase

The idle phase is the remaining time interval between the end of the asynchronous phase and the beginning of the next cycle. During the idle Phase, all network components are “waiting" for the beginning of the following cycle. The duration of the idle phase may be 0.

4.3 Service assumed from the PhL

Subclause 4.3 describes the assumed physical service (PhS) and the constraints used by the DLE. The Physical Service is assumed to provide the following service primitives specified by ISO/IEC 8802-3:2000, Clause 2.

The assumed primitives of PhS are

- MA_DATA.request
- MA_DATA.indication

The temporal relationship of the primitives is shown in Figure 4.

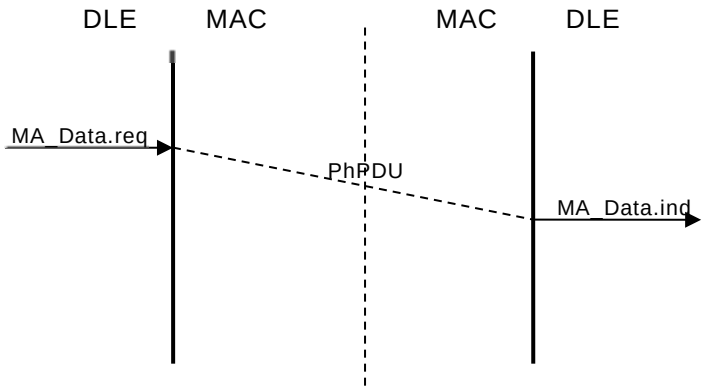


Figure 4 – Interaction of PhS primitives to DLE

The MA_DATA request primitive defines the transfer of data from a MAC client entity to a single peer entity or multiple peer entities in the case of group addresses.

The MA_DATA indication primitive defines the transfer of data from the MAC sublayer entity (through the optional MAC control sublayer, if implemented) to the MAC client entity or entities in the case of group addresses.

4.4 DLL architecture

The Type 13 fieldbus DLL is modeled as a combination of control components of cycle state machine (CSM), isochronous transmission TX/RX control (ITC), asynchronous transmission TX/RX control (ATC), asynchronous slot scheduler (ASS), exception signaling (ES), NMT signaling (NS) and DLL management interface.

The cycle state machine as the primary control component provides the function for deterministic medium access control cooperating for reliable and efficient support of higher-level data transfer services. According responsibility assignment for MN and CN the primary responsibility for CN is

- to handle communication within a Type 13 fieldbus cycle;
- to track the order of the frames received within a cycle and reacts according to the described dataflow;
- to generate an error event in case of a detected communication error;
- to hold up communication regardless to any errors.

The cycle state machine of the MN is responsible for:

- Managing the communication within a Type 13 fieldbus cycle.
- Generating the flow of the frames during a Type 13 fieldbus cycle.
- Monitoring the reaction of the CNs.
- Generating an error event in case of a detected communication error.

The data-link layer is comprised of the components listed in Table 1.

Table 1 – Data-link layer components

Components	Description
Cycle state machine (CSM)	The cycle state machine controls the Type 13 fieldbus cycle on the data-link layer, assembles and transmits the DLPDUs, receives and disassembles the DLPDUs with the control information, and determines the timing and duration of the transmissions
Isochronous transmission TX/RX control (ITC)	The ITC buffers and dispatches in time DLSDU received for the time-critical cyclic data transfer between the DLS-user and the SCM
Asynchronous transmission TX/RX control (ATC)	The ATC queues and dispatches in time DLSDU received for the asynchronous data transfer between DLS-user and the SCM
Asynchronous Slot Scheduler (ASS)	The MNs asynchronous slot scheduler decides when a requested asynchronous data transfer will happen
Exception Signaling (ES)	The exception signaling signals a detected CN exception to the MN NMT
NMT Signaling (NS)	The NS reports the current status of the NMT
DLL management interface	The DLL management interface holds the station management variables that belong to the DLL, and manages synchronized changes of the link parameters

The internal arrangement of these components, and their interfaces, are shown in Figure 5. The arrowheads illustrate the primary direction of the flow of data and control.

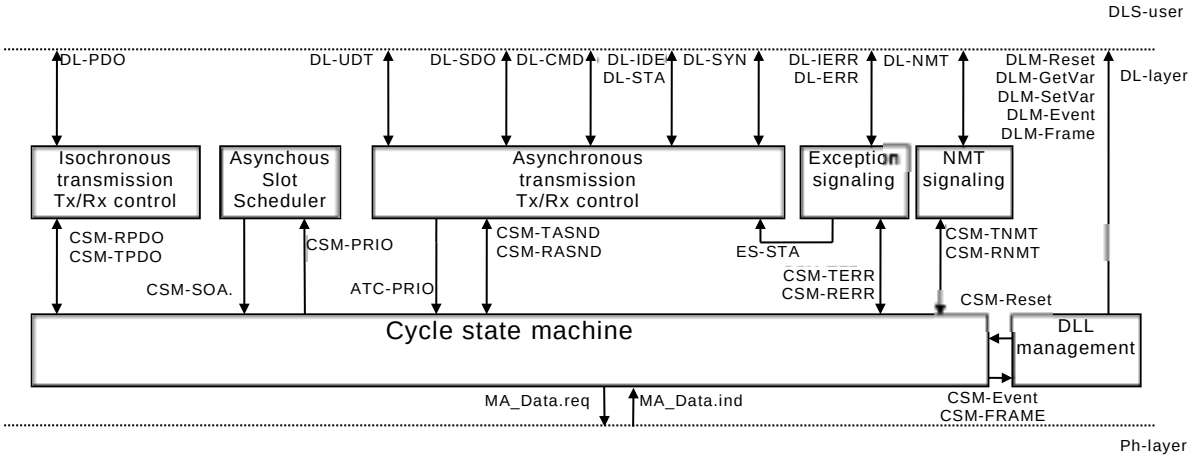


Figure 5 – Data-link layer internal architecture

4.5 Local parameters and variables

This specification uses DLS-user request parameters P(...) and local variables V(...) as a means of clarifying the effect of certain actions and the conditions under which those actions are valid, local timers T(...) as a means of monitoring actions of the distributed DLS-provider and of ensuring a local DLE response to the absence of those actions, and local counters C(...) for performing rate measurement functions.

Unless otherwise specified, at the moment of their creation or of DLE activation:

a) all variables shall be initialized to their default value, or to their minimum permitted value if no default is specified;

b) all counters shall be initialized to zero;

c) all timers shall be initialized to inactive;

DL-management may change the values of configuration variables.

4.5.1 Variables, parameter, counter and timer to support DLE function

4.5.1.1 T(ASND_TIME)

T(ASND_TIME) is used by the DLE to measure the time elapsed since last sending a SoA frame. The value is decremented in the range of V(AsyncSlotTimeout_U32) to 0.

4.5.1.2 V(AsyncSlotTimeout_U32)

This variable holds and designates the value of the timeout time for the ASnd frame in ns. An event is produced when the ASnd frame was not (or not completely) received within a preconfigured time. The initial value is 100 000 on power-up or reset of this node. The range of this value starts at 250.

4.5.1.3 P(C_DLL_ISOCHR_MAX_PAYL)

This parameter indicates the maximum size of PReq and PRes payload data in octets. The value is set to 1 490.

4.5.1.4 P(C_DLL_MAX_PAYL_OFFSET)

This parameter indicates the maximum offset of Ethernet frame payload in octets. The value is set to 1 499.

4.5.1.5 P(D_DLL_CNFeatureMultiplex_BOOL)

This parameter indicates the CN ability to perform control of multiplexed isochronous communication.

4.5.1.6 P(D_DLL_FeaturePResTimeTriggered_BOOL)

This parameter indicates the node ability to perform control of continuous-time-triggered isochronous communication.

4.5.1.7 P(D_DLL_MNFeatureMultiplex_Bool)

This parameter indicates the MNs ability to perform control of multiplexed isochronous communication.

4.5.1.8 P(D_NMT_CycleTimeMax_U32)

This parameter holds the maximum Type 13 fieldbus cycle time in μs.

4.5.1.9 P(D_NMT_CycleTimeMin_U32)

This parameter holds the minimum Type 13 fieldbus cycle time in μs.

4.5.1.10 P(D_NMT_NetTime_BOOL)

This parameter indicates the MNs ability to support of NetTime transmission via SoC DLPDU.

4.5.1.11 P(D_NMT_NetTimeIsRealTime_BOOL)

This parameter indicates the MNs ability to support of real time via the NetTime parameter in SoC DLPDU.

4.5.1.12 P(D_NMT_RelativeTime_BOOL)

This parameter indicates the MNs ability to support of RelativeTime transmission via SoC DLPDU.

4.5.1.13 T(FRAME_TIME)

T(FRAME_TIME) is used by the DLE to measure the time elapsed since last receiving a SoC, SoA and PReq frame . The value is decremented in the range of V(FRAME_TIMEOUT) to 0.

4.5.1.14 V(FRAME_TIMEOUT)

This variable holds and designates the value of the timeout time for the SoC, SoA and PReq frame. An event is produced when one of the frames were not (or not completely) received within a preconfigured time. This event signifies that a mandatory frame of the MN was lost.

4.5.1.15 V(MultiplCycleCnt_U8)

This variable describes the length of the multiplexed cycle in multiples of the Type 13 fieldbus cycle. The variable should be set by the system configuration and should be equal in all nodes of the segment. If this variable is zero, there is no support of multiplexed cycle on the network.

4.5.1.16 V(NMT_CycleLen_U32)

This variable defines the communication cycle time interval (synchronization interval) in μs. The variable should be set by the system configuration in the range of P(D_NMT_CycleTimeMin_U32) to P(D_NMT_CycleTimeMax_U32).

4.5.1.17 V(NMT_Version_U8)

The variable holds the Type 13 fieldbus communication profile version that is implemented by the device. The value shall be set by the device during system initialization.

4.5.1.18 V(NMT_IsochrSlotAssign_AU8)

This variable assigns CNs to a particular isochronous slot. The variable should be set by the system configuration in the range of 0 to 254.

4.5.1.19 V(NMT_MultiplCycleAssign_AU8)

This variable assigns CNs to the particular Type 13 fieldbus cycles of the multiplexed cycle period defined by V(MultiplCycleCnt_U8). It shall be equal in all nodes of the segment. The variable should be set by the system configuration in the range of 0 to V(MultiplCycleCnt_U8).

4.5.1.20 V(NodeID_U8)

The variable holds the device's actual Node ID. V(NodeID_U8) may be provided by hardware settings (dip switch etc.) or set up by software in the range of 1 to 240, 253 and 254.

4.5.1.21 T(PRES_TIME)

T(PRES_TIME) is used by the DLE to measure the time elapsed since last sending a PReq frame. The value is decremented in the range of V(PRES_TIMEOUT) to 0.

4.5.1.22 T(PRES_TT_TIME)

T(PRES_TT_TIME) is used by the DLE of a CN to measure the time elapsed since last receiving a PRes frame from the MN. The value is decremented in the range of V(PRES_TT_TIMEOUT) to 0.

4.5.1.23 V(PRES_TIMEOUT)

This variable holds and designates the value of the timeout time for the PRes frame in μs. An event is produced when the PRes frame was not (or not completely) received within a preconfigured time.

4.5.1.24 V(PRES_TT_TIMEOUT)

This variable holds and designates the value of the timeout time in μs for a continuous-time-triggered CN when to send its PRes frame.

4.5.1.25 V(Prescaler_U16)

This variable describes the toggle rate of the PS flag within the SoC DLPDU. The value provides the number of Type 13 fieldbus cycles that have to be completed to toggle the flag by the MN. The initial value is 2 on power-up or reset of this node. The range of this value is 0 to 1 000.

If this variable is 0, there shall be no toggling of the PS flag. V(Prescaler_U16) shall be equal in all nodes of the segment.

5 General structure and encoding of PhPDUs and DLPDU and related elements of procedure

5.1 Overview

The DLL and its procedures are necessary to provide the services offered to the DLS user by using the services available from the PhL. Clause 5 describes the structure and semantics of MA_PDU, DLPDU and the procedure, commonly used in this specification. Nevertheless this portion is identical to and fully compliant with ISO/IEC 8802-3.

NOTE Within Clause 5, any reference to bit *k* of an octet is a reference to the bit whose weight in a one-octet unsigned is 2^k, and this is sometimes referred to as "little endian" bit numbering.

5.2 MA_PDU structure and encoding

The local MAC sublayer uses the service primitives provided by the PhS sublayer specified by ISO/IEC 8802-3 Clause 2. All of the service primitives provided by the PhS sublayer are as follows and are considered mandatory:

a) MA_DATA request;

b) MA_DATA indication;

5.3 Common MAC frame structure, encoding and elements of procedure

5.3.1 MAC frame structure

5.3.1.1 MAC frame format for the Type 13 fieldbus DLPDU

The DLPDU for the Type 13 fieldbus is encapsulated in the data field of a MAC frame as specified by ISO/IEC 8802-3, Clause 3. The value of the Length/Type field is designated to $88AB_H$, which is authorized and registered as the protocol identification number by the IEEE Registration Authority, to be identified as the Type 13 fieldbus frame. Figure 6 shows the Type 13 fieldbus DLPDU.

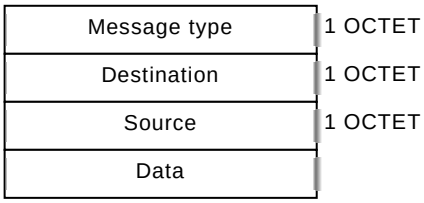


Figure 6 – Type 13 fieldbus DLPDU

The length of the frame shall be restricted to the configured size. Otherwise the cycle time cannot be guaranteed. Ethernet frames shall not be shorter than the specified minimum of 64 octets.

5.3.1.2 MAC frame format non-Type 13 fieldbus DLPDU

Any MAC frame as specified by ISO/IEC 8802-3, Clause 3 applies, with the exception of MAC frames with Length/Type field designated to $88AB_H$.

5.3.2 Elements of the MAC frame

5.3.2.1 General

This subclause specifies the Type 13 fieldbus usage of ISO/IEC 8802-3 MAC frame elements.

5.3.2.2 Destination address field (DA)

The destination address (DA) field is specified in ISO/IEC 8802-3:2000, Clause 3. The destination address field specifies the station(s) for which the frame is intended. It may be an individual or multicast (including broadcast) address.

In case of Type 13 fieldbus DLPDU Table 2 shows the MAC multicast addresses, when destination is set to “Type 13 fieldbus broadcast” (see 5.3.3.2). Otherwise the DA is set to the corresponding node ID, resolved by the application layer.

Table 2 – MAC multicast addresses

Frame type	ID / Abbr.	MAC-multicast address
Start of cycle (SoC)	SoC	01-11-1E-00-00-01 (C_DLL_MULTICAST_SOC)
PollResponse (PRes)	PRes	01-11-1E-00-00-02 (C_DLL_MULTICAST_PRES)
Start of Asynchronous (SoA)	SoA	01-11-1E-00-00-03 (C_DLL_MULTICAST_SOA)
AsynchronousSend (ASnd)	ASnd	01-11-1E-00-00-04 (C_DLL_MULTICAST_ASND)

5.3.2.3 Source address field (SA)

The source address (SA) field is specified in ISO/IEC 8802-3:2000, Clause 3. The source address field specifies the station sending the frame.

5.3.2.4 Length/type field

The Length/type field is specified in ISO/IEC 8802-3:2000, Clause 3. In order to be identified as Type 13 fieldbus frame, the value of the length/type field is set to $88AB_H$, which is authorized and registered as protocol identification number for Type 13 fieldbus by the IEEE registration authority. Every frame with the value not equal to $88AB_H$ is processed as a legacy Ethernet frame inside an asynchronous slot.

5.3.3 Elements of the Type 13 fieldbus DLPDU

This subclause specifies the elements of a Type 13 fieldbus DLPDU as depicted in Figure 6.

5.3.3.1 Message type

The message type field is used by the DLE to identify and designate the frame type of the Type 13 fieldbus for medium access control. Table 3 shows the list of message types for Type 13 fieldbus.

Table 3 – Message types

Message Type	Value	ID / Abbr.
Start of Cycle	01h	SoC
PollRequest	03h	PReq
PollResponse	04h	PRes
Start of Asynchronous	05h	SoA
Asynchronous Send	06h	ASnd

5.3.3.2 Destination

Node ID of the addressed node(s). The destination address field specifies the station(s) for which the frame is intended. It may be an individual or a broadcast address.

Table 4 shows the list of node ID for destination address(es).

Table 4 – Node ID assignment

Node ID	Description
0	Invalid
1..239	Regular Type 13 fieldbus CNs
240	Type 13 fieldbus MN
241..250	Reserved
251	Pseudo node ID to be used by a node to address itself
252	Dummy node ID
253	Type 13 CN (Diagnostic device)
254	Type 13 CN (Routing device)
255	Type 13 fieldbus broadcast

5.3.3.3 Source

Node ID of the transmitting node. The source address field specifies the station sending the frame. In case of a transmitted multicast / broadcast message, the source address is interpreted by the DLE. The value of the source address is always set to V(NodeID_U8).

Table 4 shows the list of node IDs for the source address. The node IDs 251, 252 and 255 (broadcast) shall not be used as a source address.

5.3.3.4 Data

The data field contains a sequence of N octets which provides full data transparency in the sense that any arbitrary sequence of octet values may appear in the data field up to a maximum number specified by ISO/IEC 8802-3. A minimum frame size is required, that is minFrameSize by ISO/IEC 8802-3, and if a frame size is less than minFrameSize, then the data field is extended by appending extra bits in units of octets.

The structure of this field for the Type 13 fieldbus DLPDU is described in Clause 6.

5.4 Invalid DLPDU

An invalid DLPDU shall be defined as one that meets at least one of the following conditions.

a) The frame length is inconsistent with a length value specified in the Length/Type field. If the Length/Type field contains a type value as defined by ISO/IEC 8802-3:2000, subclause 3.2.6, then the frame length is assumed consistent with this field and should not be considered an invalid DLPDU on this basis.

b) It is not an integral number of octets in length.

c) The bits of the incoming DLPDU (exclusive of the FCS field itself) do not generate a CRC value identical to the one received.

d) It is inconsistent with a Message Type value of Type 13 fieldbus DLPDU.

The contents of invalid DLPDU shall not be passed to the DLS-user or DLE. The occurrence of invalid DLPDU may be communicated to network management.

NOTE Invalid DLPDU may be ignored, discarded, or used in a private manner by DLS-users. The use of such DLPDUs is beyond the scope of this specification.

6 DLPDU-specific structure, encoding and elements of procedure

6.1 General

Clause 6 defines the structure, contents and encoding of each type and format of the DLPDU, and specifies elements of procedure for the DLPDU.

Within each subclause, the structure, contents, parameters and encoding of the DLPDU are described, and the Type 13 fieldbus specific part of the DLPDU structure, which is shown in Figure 6, is specified. The aspects relating to the sending and receiving of DLS-users and their DLEs are further described. All data format and encoding is described in little endian format throughout Clause 6.

NOTE Within Clause 6, any reference to bit *k* of an octet is a reference to the bit whose weight in a one-octet unsigned is 2^k, and this is sometimes referred to as "little endian" bit numbering.

6.2 Overview

Type 13 fieldbus collects more than one function into one frame. It is therefore not usually possible to apply a single communication relationship to the complete frame, but only to particular services inside the frame.

The PRes DLPDU for example transmitted by the CN includes several services.

- Transmission of the current NMT status of the CN is the response part of an unconfirmed master/slave relationship triggered by the MN.
- Request of the asynchronous slot is the request part of a client/server relationship.
- Transmission of PDO data occurs in conformance to a push model producer/consumer relationship.

6.3 Start of synchronization (SoC)

6.3.1 General

The SoC DLPDU is used for synchronisation of all nodes. Only the MN shall initiate the SoC DLPDU.

At the beginning of a Type 13 fieldbus cycle, the MN shall send a SoC DLPDU to all nodes. The sending and receiving time of this frame is the basis for the common timing of all the nodes. Only the SoC DLPDU is generated on a periodic basis. The generation of all other frames shall be event controlled (with additional time monitoring per node).

The MN starts the isochronous data exchange after the SoC frame has been sent.

6.3.2 Structure of the SoC DLPDU

The structure of SoC DLPDU is shown in Table 5.

Table 5 – Structure of SoC DLPDU

DLPDU field	Data type	Value/description
Message type	OCTET[1]	01h / Identify the SoC DLPDU
Destination	OCTET[1]	Node ID of the addressed node
Source	OCTET[1]	Node ID of the transmitting node
reserved	OCTET[1]	
SOC-flag	OCTET[1]	
reserved	OCTET[1]	
NetTime	OCTET[8]	Starting time of the Type 13 fieldbus cycle
RelativeTime	OCTET[8]	Relative time
reserved	OCTET[24]	

6.3.2.1 Parameters of SoC DLPDU

6.3.2.1.1 Destination (dest)

This parameter indicates the Node ID of the addressed node (see 5.3.3.2). For SoC DLPDU the destination is set to 255 (C_ADDR_BROADCAST).

6.3.2.1.2 Source (src)

This parameter indicates the Node ID of the transmitting node (see 5.3.3.3). For SoC DLPDU, the parameter is set to 240 (C_ADDR_MN_DEF_NODE_ID).

6.3.2.1.3 SOC-Flag

6.3.2.1.3.1 General

The structure of the SoC-Flag is shown in Table 6.

Table 6 – Structure of SoC-Flag

Bit number	Value	Description
7		Multiplexed cycle complete (MC)
6		Prescaled slot (PS)
5-0		Reserved

6.3.2.1.3.2 Multiplexed cycle complete (MC)

This parameter is toggled by the MN when the final multiplexed cycle has ended. The parameter is generated within the ITC and signaled to the application within the DLMS frame status.

6.3.2.1.3.3 Prescaled slot (PS)

This parameter is toggled by the MN every n-th cycle (n is configurable by V(Prescaler_U16))

The parameter is signaled to the application within the DLMS frame status.

NOTE This prescaled signal is useful for “slow” nodes, which cannot react every cycle.

6.3.2.1.4 NetTime (time)

This parameter is distributed by the MN and indicates the starting time of the Type 13 fieldbus cycle.

NetTime transmission is optional. Support is indicated by P(D_NMT_NetTime_BOOL).

IEC 61588 conform distribution via the parameter NetTime is indicated by P(D_NMT_NetTimeIsRealTime_BOOL).

6.3.2.1.5 RelativeTime (reltime)

This parameter is distributed by the MN and indicates the relative time, which is incremented by the Type 13 fieldbus cycle time when a SoC DLPDU is generated. The unit of RelativeTime is µs.

RelativeTime transmission is optional. Support is indicated by P(D_NMT_RelativeTime_BOOL).

6.3.2.2 User data

No user data is conveyed by the SoC DLPDU.

6.3.3 Sending SoC DLPDU

The CSM of the MN sends out the SoC DLPDU with a set of parameters, as stated above. The time interval of the periodic basis is determined by V(NMT_CycleLen_U32).

6.3.4 Receiving the SoC DLPDU

Each node receives this DLPDU to synchronize his behavior on this DLPDU. The CSM of the CNs use this DLPDU for:

- Indicating the beginning of a Type 13 fieldbus cycle.
- Exception recognition.
- Cycle time overrun.

The reception and the parameter of this DLPDU is signaled by the DLMS frame status for synchronization purpose.

The MN shall observe network traffic in order to ensure, that there is no other MN active on the network. Reception of a SoC or a SoA DLPDU indicates that there is another MN active.

6.4 PollRequest (PReq)

6.4.1 General

The real-time data transfer is performed by means of process data objects (PDO). PDO communication in Type 13 fieldbus is always performed isochronously by PReq and PRes DLPDUs.

The MN only sends PReq DLPDU to one CN per frame. PReq transfer is dedicated to data relevant for the addressed CN only. The addressed CN response its process data object to all nodes. This makes communication relationships possible according to the producer/consumer model. The PReq / PRes procedure shall be repeated for each configured and active isochronous CN during each cycle. The MN may send an PRes DLPDU to all CN.

A further assignment of the PReq DLPDU is the exception signaling. A flag inside the DLPDU is used to acknowledge an exception signal received by a particular CN to the particular CN.

6.4.2 Structure of the PReq DLPDU

6.4.2.1 General

The structure of PReq DLPDU is shown in Table 7.

Table 7 – Structure of PReq DLPDU

DLPDU field	Data type	Value/description
Message type	OCTET[1]	03h / identify the PReq DLPDU
Destination	OCTET[1]	Node ID of the addressed node
Source	OCTET[1]	Node ID of the transmitting node
reserved	OCTET[1]	
PReq-flag	OCTET[1]	
reserved	OCTET[1]	
Payload	OCTET[4 to P(C_DLL_MAX_PAYL_OFFSET)-10]	DLSDU

6.4.2.2 Parameters of PReq DLPDU

6.4.2.2.1 Destination (dest)

This parameter indicates the Node ID of the addressed node (see 5.3.3.2). For PReq DLPDU only individual CN Node IDs are allowed.

This parameter is inserted/evaluated to/from the DLS-User as the D_addr parameter within the DLS DL-PDO.

6.4.2.2.2 Source (src)

This parameter indicates the node ID of the transmitting node (see 5.3.3.3). For PReq DLPDU, the parameter is set to 240 (C_ADDR_MN_DEF_NODE_ID).

This parameter is passed to the DLS-User as the S_addr parameter within the DLS DL-PDO.

6.4.2.2.3 PRreq-Flag

6.4.2.2.3.1 General

The structure of PReq-Flag is shown in Table 8.

Table 8 – Structure of PReq-Flag

Bit number	Value	Description
7-6		Reserved
5		Multiplexed slot (MS)
	0	Isochronous slot is handled in a non-multiplexed slot
	1	Isochronous slot is handled in a multiplexed slot
4-3		Reserved
2		Exception acknowledge (EA)
1		Reserved
0		Ready (RD)
	0	DLSDU is not valid
	1	DLSDU is valid

6.4.2.2.3.2 Multiplexed slot (MS)

This parameter is set by the MN when the addressed CN is handled in a multiplexed timeslot. The parameter is generated within the ITC and signaled to the application within DLS DL-PDO.

6.4.2.2.3.3 Exception acknowledge (EA)

This parameter is toggled by the MN to acknowledge a requested exception signal from the addressed CN. The parameter is generated / evaluated within the ES.

For async-only CNs, the parameter is also signaled within the DLSDU of DL-STA. The CSM is responsible to insert/remove this parameter into the DL-STA.

6.4.2.2.3.4 Ready (RD)

This parameter indicates that the transferred payload data are valid. The parameter is passed/received to/from the DLS-User within the DLS DL-PDO primitive.

6.4.2.2.4 Payload (pl)

PReq DLPDU can transmit user data up to the length of P(C_DLL_ISOCHR_MAX_PAYL). The parameter is passed/received to/from the DLS-User within the DLS DL-PDO primitive.

6.4.3 Sending PReq DLPDU

A PReq DLPDU shall be sent to every configured and active node every Type 13 fieldbus cycle under consideration of the multiplexed slot. A PReq DLPDU shall not be sent to a CN configured as a continuous-time-triggered node.

The CSM of the MN administrates access to all CNs via PReq DLPDU during a multiplexed cycle. For each slot, the MNs CSM requests the transmitting DLSDU via ITC and assembles them to the PReq DLPDU including the exception signaling information.

The MN uses a timeout after sending a PReq DLPDU for receiving the corresponding PRes DLPDU response to detect transmission errors and node failures.

6.4.4 Receiving the PReq DLPDU

Each isochronous continuous or multiplexed CN shall receive a PReq DLPDU from the MN in the Type 13 fieldbus cycle and shall send a PRes DLPDU to the MN. CNs may be accessed every cycle or every n^{th} cycle (multiplexed nodes, n > 1).

The received data is unpacked and passed directly to the ITC via CSM-RPDO. The exception acknowledge parameter is passed to the ES via CSM-ERR for further processing.

6.5 Poll response (PRes)

6.5.1 General

Each isochronous continuous or multiplexed CN shall receive an unicast PReq DLPDUs from the MN in the Type 13 fieldbus cycle and shall response with a PRes DLPDU to the MN. Each isochronous continuous-time-triggered CN shall receive the PRes DLPDU from the MN in the Type 13 fieldbus cycle and shall send a PRes DLPDU to the MN on a time triggered basis. PReq and PRes DLPDU may transport isochronous data.

PReq can only be received by the specifically addressed CN. However, PRes DLPDU shall be sent by the CN as multicast messages, allowing all other CNs to monitor the data being sent.

Additional data from the MN may be received by a multicast PRes DLPDU transmitted by the MN.

CNs participating in the isochronous slot shall request the right to transmit asynchronous data from the MN via the PR / PS parameter of the PRes DLPDU.

New exception conditions and the current NMT status of the CN is also transmitted within the PRes DLPDU.

6.5.2 Structure of the PRes DLPDU

6.5.2.1 General

The structure of PRes DLPDU is shown in Table 9.

Table 9 – Structure of PRes DLPDU

DLPDU field	Data type	Value/description
Message type	OCTET[1]	04h / identify the PRes DLPDU
Destination	OCTET[1]	Node ID of the addressed node
Source	OCTET[1]	Node ID of the transmitting node
NMTStatus	OCTET[1]	Status of the CNs NMT state machine
PRes-flag	OCTET[2]	
Payload	OCTET[4 to P(C_DLL_MAX_PAYL_OFFSET)-10]	DLSDU

6.5.2.2 Parameters of PRes DLPDU

6.5.2.2.1 Destination (dest)

This parameter indicates the node ID of the addressed node (see 5.3.3.2). For PRes DLPDU the destination is set to 255 (C_ADDR_BROADCAST).

This parameter is inserted/evaluated to/from the DLS-User as the D_addr parameter within the DLS DL-PDO.

6.5.2.2.2 Source (src)

This parameter indicates the node ID of the transmitting node (see 5.3.3.3).

This parameter is passed to the DLS-User as the S_addr parameter within the DLS DL-PDO.

6.5.2.2.3 NMTStatus (stat)

This parameter reports the current status of the CNs NMT state machine and is inserted/evaluated to/from the DLS-User with DLS DL-NMT.

6.5.2.2.4 PRes-Flag

6.5.2.2.4.1 General

The structure of PRes-Flag is shown in Table 10.

Table 10 – Structure of PRes-Flag

Bit number	Value	Description
15-14		Reserved
13		Multiplexed slot (MS)
	0	Isochronous slot is handled in a non-multiplexed slot
	1	Isochronous slot is handled in a multiplexed slot
12		EN
11-9		Reserved
8		Ready (RD)
	0	DLSDU is not valid
	1	DLSDU is valid
7-6		Reserved
5-3		Priority (PR)
	000	PRIO_0
	001	PRIO_1
	010	PRIO_2
	011	PRIO_GENERIC_REQUEST
	100	PRIO_4
	101	PRIO_5
	110	PRIO_6
	111	PRIO_NMT_REQUEST
2-0		RequestToSend (RS)
	000	no pending request
	001	1 pending request
	010	2 pending requests
	011	3 pending requests
	100	4 pending requests
	101	5 pending requests
	110	5 pending requests
	111	7 and more pending requests

6.5.2.2.4.2 Multiplexed slot (MS)

This parameter is done by the corresponding PReq DLPDU when the addressed CN is handled in a multiplexed timeslot. The parameter is signaled to the application within the DLS DL-PDO primitive.

Based on this information, other CNs can identify that the transmitting CN is served by a multiplexed slot.

6.5.2.2.4.3 Exception new (EN)

This parameter is toggled by the CN to indicate an exception signal to the MN. The parameter is generated / evaluated within ES.

For async-only CNs, the parameter is also signaled within the DLSDU of DL-STA. The CSM is responsible to insert/remove this parameter into the DL-STA.

6.5.2.2.4.4 Ready (RD)

This parameter indicates that the transferred payload data are valid. The parameter is passed/received to/from the DLS-User within the DLS DL-PDO primitive.

6.5.2.2.4.5 Priority (PR)

This parameter indicates the priority of the frame(s) in the asynchronous send queue with the highest priority. The range of this value is 0 (lowest priority) to 7 (highest priority). Two of these levels are dedicated to Type 13 fieldbus purpose:

- PRIO_NMT_REQUEST: This is the highest priority that shall be exclusively applied if a CN requests an NMT command to be issued by the MN.
- PRIO_GENERIC_REQUEST: Medium priority which is the standard priority level for non-NMT command requests.

The remaining priority levels above and below PRIO_GENERIC_REQUEST are available for application purpose.

Before transmitting the PRes DLPDU, ATC is requested for the current status of the asynchronous send queues.

6.5.2.2.4.6 RequestToSend (RS)

This parameter indicates the number of pending frames in the asynchronous send queue with the highest priority. The range of this value is 0 (no pending request) to 7 (7 and more pending requests).

Before transmitting the PRes DLPDU, ATC is requested for the current status of the asynchronous send queues.

6.5.2.2.5 Payload (pl)

PRes DLPDU can transmit user data up to the length of P(C_DLL_ISOCHR_MAX_PAYL). The parameter is passed/received to/from the DLS-User within the DLS DL-PDO primitive.

6.5.3 Sending PRes DLPDU

A PRes DLPDU shall be sent as the response to a received PReq DLPDU in case of a continuous or multiplexed CN, as the response to a received PRes DLPDU from the MN in case of a continuous-time-triggered CN on a time triggered basis or by the MN.

The DLPDU is assembled by requesting the ITC for the DLSDU transmitting to all nodes, requesting the ATC for the current status of the asynchronous sending queues, requesting the ES for the actual exception new parameter and requesting the NS for the actual CNs NMTStatus.

6.5.4 Receiving the PRes DLPDU

The PRes DLPDU is received by all isochronous CNs as well as the MN.

The received data is unpacked and passed directly to the ITC via the CSM-RPDO service. The EA parameter is passed to the ES via CSM-ERR and the NMTStatus is passed to the NS for further processing.

The MN uses a timeout after sending a PReq DLPDU resp. after sending its PRes DLPDU in case of continuous-time-triggered nodes for receiving the corresponding PRes DLPDU response to detect transmission errors and node failures.

6.6 Start of asynchronous (SoA)

6.6.1 General

The SoA DLPDU shall be used to

- identify CNs,
- request status information of a CN,
- send NMT commands,
- poll async-only CNs and
- grant the asynchronous transmit right to one CN,
- synchronize continuous-time-triggered CNs.

The SoA DLPDU is the first frame in the asynchronous phase and is a signal to all CNs that all isochronous data have been exchanged during the isochronous phase. If no asynchronous message transmission request is pending the SoA DLPDU without assignment of the right to send any message is transmitted.

The SoA DLPDU is handled by the MN. The MN knows its own asynchronous send queue and the queues of all active CNs, reported via PRes DLPDU. The ASS of the MN shall determine in which cycle the right to send the asynchronous frame will be granted. It shall guarantee that no send request will be delayed for an indefinite amount of time, even if network load is high.

Assignment of the asynchronous slot to the MN itself shall be indicated in the same way as assignments to CNs.

6.6.2 Structure of the SoA DLPDU

6.6.2.1 General

The structure of SoA DLPDU is shown in Table 11.

Table 11 – Structure of SoA DLPDU

DLPDU field	Data type	Value/description
Message Type	OCTET[1]	05h / identify the SoA DLPDU
Destination	OCTET[1]	Node ID of the addressed node
Source	OCTET[1]	Node ID of the transmitting node
NMTStatus	OCTET[1]	Status of the MNs NMT state machine
SoA-Flag	OCTET[1]	
reserved	OCTET[1]	
svid	OCTET[1]	Requested serviceID (svid)
svtg	OCTET[1]	Requested service target (svtg)
FieldbusVersion	OCTET[1]	Type 13 fieldbus version of the MN
reserved	OCTET[1]	
Payload	OCTET[36]	DLSDU

6.6.2.2 Parameters of SoA DLPDU

6.6.2.2.1 Destination (dest)

This parameter indicates the node ID of the addressed node (see 5.3.3.2). For SoA DLPDU the destination is set to 255 (C_ADDR_BROADCAST).

NOTE The target of this DLPDU is indicated within the RequestedServiceTarget parameter (see 6.6.2.2.6). Destination set to 255 (C_ADDR_BROADCAST) is necessary for synchronization purpose of the CSM.

6.6.2.2.2 Source (src)

This parameter indicates the Node ID of the transmitting node (see 15.4.8). For SoA DLPDU, the parameter is set to 240 (C_ADDR_MN_DEF_NODE_ID).

6.6.2.2.3 NMTStatus (stat)

This parameter reports the current status of the MNs NMT state machine and is inserted/evaluated to/from the DLS-User with the DLS DL-NMT.

6.6.2.2.4 SoA-Flag

6.6.2.2.4.1 General

The structure of SoA-Flag is shown in Table 12.

Table 12 – Structure of SoA-Flag

Bit number	Value	Description
7-3		Reserved
2		Exception acknowledge (EA)
1		Exception reset (ER)
	0	No request
	1	Request initialization of the exception system
0		Reserved

6.6.2.2.4.2 Exception acknowledge (EA)

This parameter is toggled by the MN to acknowledge a requested exception signal from the addressed CN. The parameter is generated / evaluated within the ES. The parameter is valid only, if RequestedServiceID equals StatusRequest.

For isochronous CNs, the parameter is also signaled within the PReq DLPDU. The CSM is responsible to insert/remove this parameter into the DL-STA.

6.6.2.2.4.3 Exception reset (ER)

This parameter is set by the MN to reset the internal exception flag from the addressed CN. The parameter is generated / evaluated within the ES. The parameter is valid only, if RequestedServiceID equals StatusRequest.

6.6.2.2.5 RequestedServiceID (svid)

This parameter indicates the asynchronous service ID dedicated to the following asynchronous slot.

Table 13 shows the definitions of the RequestServiceID entries.

Table 13 – Definition of the RequestedServiceID in the SoA DLPDU

Description	Value (ID)	Description
NoService	00_h NO_SERVICE	Shall be used if the asynchronous slot is not assigned to any node. RequestedServiceTarget shall be C_ADDR_INVALID
IdentRequest	01_h IDENT_REQUEST	Shall be used to identify inactive CNs and/or to query the identification data of a CN. The addressed CN shall answer immediately after the reception of the SoA with the node specific IdentResponse ASnd DLPDU
StatusRequest	02_h STATUS_REQUEST	Shall be used to request the current status and detailed error information of a node. Async-only CNs shall be cyclically queried by StatusRequest to supervise their status and to query their requests for the asynchronous slot. The addressed node shall answer immediately after the reception of the SoA, with the node specific StatusResponse ASnd DLPDU frame
NMTRequestInvite	03_h NMT_REQUEST_INVITE	Shall be used to assign the asynchronous slot to a node that has indicated a pending NMTCommand / NMTRequest. The addressed node shall answer immediately after the reception of the SoA with the NMTCommand / NMTRequest ASnd DLPDU
SyncRequest	06_h SYNC_REQUEST	Shall be used to configure/synchronize continuous-time-triggered CNs and/or to query the synchronization data of a CN. The addressed CN shall answer immediately after the reception of the SoA with the node specific SyncResponse ASnd DLPDU
UnspecifiedInvite	FF_h UNSPECIFIED_INVITE	Shall be used to assign the asynchronous slot to a node that has indicated a pending transmit request. The addressed node shall answer immediately after the reception of the SoA, with any kind of a Type 13 fieldbus ASnd or a legacy Ethernet

Description	Value (ID)	Description
		DLPDU
NOTE All ServiceIDs without NoService are handled within the ATC.		

The MNs ASS decides within the known send queue status, which node with which status gets the next asynchronous slot.

6.6.2.2.6 RequestedServiceTarget (svtg)

This parameter indicates the address (DL-address) of the node, which shall send within the asynchronous slot.

C_ADDR_INVALID shall indicate the asynchronous slot is not assigned.

6.6.2.2.7 Type 13 fieldbus Version (eplv)

This parameter indicates the current Type 13 fieldbus version of the MN.

6.6.2.3 Payload

6.6.2.3.1 General

Only in the SyncRequest user data is conveyed by the SoA DLPDU. In all other cases no user data is conveyed by the SoA DLPDU.

6.6.2.3.2 SyncRequest

The parameter is passed/received to/from the DLS-User as the DLSDU parameter of the DLS DL-SYN.

6.6.3 Sending SoA DLPDU

A SoA DLPDU is sent by the MN at the beginning of the asynchronous phase and assigns the asynchronous phase to a specific node, if necessary. The MNs ASS decides in a fair way, which ServiceID of which node shall be served next.

The DLPDU is assembled by requesting the ASS for the next ServiceID and ServiceTarget. The EA and the ER parameters are gathered from the ES. The MNs NMTStatus is requested from the NS.

6.6.4 Receiving the SoA DLPDU

After reception of a PReq DLPDU the CSM waits for the reception of a SoA DLPDU. Reception of a SoA DLPDU confirms the end of the isochronous phase.

The RequestedServiceTarget parameter of the received SoA DLPDU is analyzed. Dependend on the requested ServiceID, the transmission of an ASnd DLPDU shall occur immediately after the transmission / reception of a SoA DLPDU. The ATC is requested for the requested ServiceID.

The EA and the EN parameters are passed to the ES via CSM-ERR and the NMTStatus is passed to the NS for further processing.

The MN shall observe network traffic in order to ensure, that there is no other MN active on the network. Reception of a SoC or a SoA DLPDU indicates that there is another MN active.

Because of the distinct MAC address of each node it is not possible that two or more nodes own the same MAC address but it is still possible that two or more nodes own the same Type 13 fieldbus address. Only in the asynchronous communication multiple CNs can answer on a SoA DLPDU. Since the MN sends a MAC Broadcast (SoA DLPDU) to a Unicast Type 13 fieldbus address, several CNs with the same Type 13 fieldbus address are able to respond. The MN shall detect multiple used Type 13 fieldbus addresses in a network by counting the responses on an IdentRequest SoA DLPDU.

6.7 Asynchronous send (ASnd)

6.7.1 General

There are two types of asynchronous DLPDUs available:

- a non Type 13 fieldbus DLPDU may be sent,
- a Type 13 fieldbus DLPDU in the form of a ASnd DLPDU.

Although only one asynchronous DLPDU per Type 13 fieldbus cycle is allowed, the CSM of the CN does not limit the amount of received DLPDUs within the asynchronous phase of the cycle.

Upon assignment of the asynchronous slot via the SoA DLPDU, the requested node arranges the requested information and sends it via a non Type 13 fieldbus DLPDU or via an ASnd DLPDU. Different ServiceIDs in an ASnd DLPDU are possible.

- StatusResponse: Is used to distribute the current status and detailed error information of a node.
- IdentResponse: Is used to distribute identification data of inactive CNs waiting to be included into the network and/or to query the identification data of a CN.
- NMTCommand: Is used by the MN NMT to distribute common NMT information and to force specific commands to a single, to specific or to all nodes on the Type 13 fieldbus network.
- NMTRequest: Is used to give CNs the possibility also to request the above functions. The requested function is decoded in the payload and distributed by the MN in one of the next asynchronous slots via ASnd-DLPDU.
- SDO: Is used for peer to peer communication with access to the object dictionary of a device.
- SyncResponse: Is used to distribute the current synchronization data of a continuous-time-triggered CN.

The addressed node shall answer immediately after the reception of the SoA.

6.7.2 Structure of the ASnd DLPDU

6.7.2.1 General

The structure of ASnd DLPDU is shown in Table 14.

Table 14 – Structure of ASnd DLPDU

DLPDU field	Data type	Value/description
Message Type	OCTET[1]	06h / identify the ASnd DLPDU
Destination	OCTET[1]	Node ID of the addressed node
Source	OCTET[1]	Node ID of the transmitting node
ServiceID	OCTET[1]	Service ID
Payload	OCTET[1 to P(C_DLL_MAX_PAYL_OFFSET)-4]	DLSDU

6.7.2.2 Parameters of ASnd DLPDU

6.7.2.2.1 Destination (dest)

This parameter indicates the node ID of the addressed node (see 5.3.3.2). For ASnd DLPDU the destination depends on the used ServiceID (see 6.7.2.2.3).

6.7.2.2.2 Source (src)

This parameter indicates the node ID of the transmitting node (see 5.3.3.3).

6.7.2.2.3 ServiceID (svid)

This parameter indicates the ServiceID dedicated to the asynchronous slot.

Table 15 shows the definitions of the ServiceID entries.

Table 15 – Definition of the ServiceID in the ASnd DLPDU

Description	Value (ID)	Destin-ation	Description
IdentResponse	01_h (IDENT_RESPONSE)	Type 13 fieldbus broadcast	Within the IdentResponse ServiceID the CN distributes identification data. This ServiceID shall be issued by a node that received an IdentRequest via SoA
StatusResponse	02_h (STATUS_RESPONSE)	Type 13 fieldbus broadcast	Within the StatusResponse ServiceID the CN distributes the current status and detailed error information. This ServiceID shall be issued by a node that received a StatusRequest via SoA
NMTRequest	03_h (NMT_REQUEST)	Type 13 fieldbus MN	Within the NMTRequest ServiceID the CN asks the MN to request the specified ServiceID within the next asynchronous phase. This ServiceID shall be issued by a CN that received a NMTRequestInvite via SoA
NMTCommand	04_h (NMT_COMMAND)	(see Note 2)	Within the NMTCommand ServiceID the MN NMT distributes common NMT information and forces specific commands to a single, to specific or to all nodes on the Type 13 fieldbus network. This ServiceID shall be issued by the MN upon an internal request or upon an external request via NMTRequest
SDO	05_h (SDO)	Type 13 fieldbus CNs and Type 13 fieldbus	Within the SDO ServiceID, a peer to peer communication with access to the object dictionary of a device is issued. This ServiceID may be issued by a node that received an UnspecifiedInvite via SoA.

Description	Value (ID)	Destin-ation	Description
		MN	
SyncResponse	06_h (SYNC_RESPONSE)	Type 13 fieldbus broadcast	Within the SyncResponse ServiceID the CN distributes the current synchronization data. This ServiceID shall be issued by a node that received a SyncRequest via SoA
Manufacturer specific	$A0_h$ to FE_h	Type 13 fieldbus CNs and Type 13 fieldbus MN	Within the manufacturer specific Service ID the DLS-User may distribute any kind of data. The use of these service IDs is beyond the scope of this document. This ServiceID may be issued by a node that received an UnspecifiedInvite via SoA.
NOTE 1 ServiceIDs not listed are reserved. NOTE 2 The destination is depending on the underlying NMT command. Type 13 fieldbus CNs as well as Type 13 fieldbus broadcast is possible.			

6.7.2.2.4 Payload (pl)

6.7.2.2.4.1 General

ASnd DLPDU can transmit user data up to the length of P(C_DLL_ISOCHR_MAX_PAYL).

The following user data are defined for the ASnd DLPDU payload parameter:

6.7.2.2.4.2 IdentResponse

The parameter is passed/received to/from the DLS-User as the DLSDU parameter of the DLS DL-IDE.

6.7.2.2.4.3 StatusResponse

The parameter is passed/received to/from the DLS-User as the DLSDU parameter of the DLS DL-STA.

6.7.2.2.4.4 NMTCommand

The parameter is passed/received to/from the DLS-User as the DLSDU parameter of the DLS DL-CMD.

6.7.2.2.4.5 NMTRequest

CNs requesting a NMTCommand, an IdentResponse and a StatusResponse handle this request by inviting the MN with a NMTRequest to accomplish the requested service.

As shown in Table 16 the NMTRequest user data shall contain the NMTRequested-CommandID, NMTRequestedCommandTarget and NMTRequestedData.

Table 16 – Structure of NMTRequest user data

DLPDU field	Data type	Value/description
NMTRequestCommandID	OCTET[1]	
NMTRequestedCommandTarget	OCTET[1]	
NMTRequestedCommandData	OCTET[1 to P(C_DLL_MAX_PAYL_OFFSET)-4]	

where:

- NMTRequestedCommandID (rcid): The NMT service shall be issued by the MN by its NMTCommandID value. StatusResponse and IdentResponse services are indicated by the SoA RequestedServiceID values STATUS_REQUEST and IDENT_REQUEST.
- NMTRequestedCommandTarget (rct): Indicates the target node of the requested NMT command.
- NMTRequestedCommandData (rcd): NMT command specific data to be issued by the MN.

6.7.2.2.4.6 SDO

The parameter is passed/received to/from the DLS-User as the DLSDU parameter of the DLS DL-SDO.

6.7.2.2.4.7 SyncResponse

The parameter is passed/received to/from the DLS-User as the DLSDU parameter of the DLS DL-SYN.

6.7.2.2.4.8 Manufacturer specific

The parameter is passed/received to/from the DLS-User as the DLSDU parameter of the DLS DL-UDT.

6.7.3 Sending ASnd DLPDU

After invited by a SoA DLPDU, the requested CN sends an ASnd DLPDU, with ServiceID set to:

- IdentResponse: Within the ATC, the user data is requested with the DLS DL-IDE. The responded user data is assembled with the ServiceID=IdentResponse to the ASnd DLPDU.
- StatusResponse: Within the ATC, the user data is requested with the DLS DL-STA. The responded user data is assembled with the ServiceID=StatusResponse to the ASnd DLPDU.
- NMTCommand: Requested NMTCommands from the DLS-User are queued in the ATC. Within the ATC, the oldest queue content is responded and this user data is assembled with the ServiceID=NMTCommand to the ASnd DLPDU.
- NMTRequest: Requested NMTCommands, StatusRequest and IdentRequests from CNs are queued in the ATC. Within the ATC, the oldest queue content is translated into a NMTRequest user data and this is assembled with the ServiceID=NMTReqeust to the ASnd DLPDU to be executed at the MN.
- SDO: Requested SDO DLSDUs from the DLS-User are queued in the ATC. Within the ATC, the oldest queue content is responded and this user data is assembled with the ServiceID=SDO to the ASnd DLPDU.
- SyncResponse: Within the ATC, the user data is requested with the DLS DL-SYN. The responded user data is assembled with the ServiceID=SyncResponse to the ASnd DLPDU.
- Manufacturer specific: Requested UDT DLSDUs from the DLS-User are queued in the ATC. Within the ATC this user data is assembled with the ServiceID=”Manufacturer specific” to the ASnd DLPDU.

6.7.4 Receiving the ASnd DLPDU

If an ASnd frame has been received it shall be processed depending on the ServiceID as follow:

- IdentResponse: Received IdentResponse ASnd DLPDUs are passed to the DLS-user via the DL-IDE primitive of the ATC. The MN shall receive the IdentResponse. CNs may receive IdentResponse if configured to do so.
- StatusResponse: Received StatusResponse ASnd DLPDUs are passed to the DLS-user via the DL-STA primitive of the ATC. The MN shall receive the StatusResponse. CNs may receive StatusResponse if configured to do so.
- NMTCommand: Received NMTCommand ASnd DLPDUs are passed to the DLS-user via the DL-CMD primitive of the ATC.
- NMTRequest: Received NMTRequest ASnd DLPDUs are interpreted by the MNs ATC and pushed into the Status, Ident or the NMTCommand queue, depending on the requesting NMTRequestedCommandID. CNs do not receive this DLPDU.
- SDO: Received SDO ASnd DLPDUs are passed to the DLS-user via the DL-SDO primitive of the ATC.
- SyncResponse: Received SyncResponse ASnd DLPDUs are passed to the DLS-user via the DL-SYN primitive of the ATC. The MN shall receive the SyncResponse. CNs shall receive SyncResponse if supporting continuous-time-triggered communication.
- Manufacture specific: Received UDT ASnd DLPDUs are passed to the DLS-user via the DL-UDT primitive of the ATC.

7 DLE elements of procedure

7.1 Overall structure

The DLL is composed of control elements of isochronous transmission TX/RX control (ITC), asynchronous transmission TX/RX Control (ATC), cycle state machine (CSM), asynchronous slot scheduler (ASS), exception signaling (ES), NMT signaling (NS) and DLL Management Interface.

The CSM as the primary control element provides the function for deterministic medium access control cooperating with the ITC, the ATC, the ES and the NS for reliable and efficient support both of isochronous and asynchronous data transfer services.

The DLL management interface provides DLL management functions.

7.2 Cycle state machine (CSM)

7.2.1 Overview

7.2.1.1 Cycle state machine of the MN (MN-CSM)

The cycle state machine of the MN (MN-CSM) shall manage the communication within a Type 13 fieldbus cycle. The MN-CSM generates the flow of the frames during a Type 13 fieldbus cycle and monitors the reaction of the CNs. The flow order is dependent on the state of the NMT (signaled via the primitive CSM-TNMT).

7.2.1.2 Cycle state machine of the CN (CN-CSM)

The cycle state machine of the CN (CN-CSM) handles communication within a Type 13 fieldbus Cycle. The CN-CSM tracks the order of the frames received within a cycle and reacts as described below. The expected order of frame reception is dependent on the state of the NMT (signaled via the primitive CSM-TNMT).

7.2.2 Primitive definitions

7.2.2.1 Primitive definitions between CSM and ITC

Table 17 summarizes all primitives exchanged between the CSM and the ITC.

Table 17 – Primitives exchanged between CSM and ITC

Primitive name	Source	Associated parameters	Description
CSM-PDO.ind	ITC	Isochr Isochr-time-triggered Isochr-Out	Indicates the assignment of the next isochronous slot.
CSM-TPDO.ind	ITC	D_addr DLSDU	Indication of a PDO element with its associated parameters for transmission at the next isochronous slot. MN: The requested PDO element is determined by the ITC internal slot counter. CN: Only the CN dependent PDO is requested once per Type 13 fieldbus cycle.
CSM-RPDO.ind	CSM	S_addr D_addr DLSDU	Carries forward a received PDO element with its associated parameters to the ITC.

The parameters used with the primitives exchange between the CSM and the ITC are described in Table 18.

Table 18 – Parameters used with primitives exchanged between CSM and ITC

Parameter name	Description
S_addr	The S_addr parameter specifies the DL-address of the publisher.
D_addr	The D_addr parameter specifies the DL-address of the subscriber.
DLSDU	This parameter specifies the information that is transferred by buffer transfer from the local DLE as a publisher to the remote multi-peer DLEs as subscribers.
Isochr	This parameter specifies with a value unequal zero that there are nodes in the isochronous list of the current cycle.
Isochr-time-triggered	This parameter specifies with a value unequal zero that there are nodes in the isochronous list of the current cycle which are of communication class *continuous-time-triggered*.
Isochr-Out	This parameter specifies with a value unequal zero that the MN is configured to send a PRes.

7.2.2.2 Primitive definitions between CSM and ATC

Table 19 summarizes all primitives exchanged between the CSM and the ATC.

Table 19 – Primitives exchanged between CSM and ATC

Primitive name	Source	Associated parameters	Description
CSM-TASND.ind	ATC	Destination ServiceID Payload	Indicates an ASnd DLPDU with its associated parameters from the ATC internal queue for transmission at the next asynchronous slot.
CSM-RASND.ind	CSM	Destination Source ServiceID Payload	Carries forward a received ASnd DLPDU with its associated parameters to the ATC.
ATC-Prio.req	CSM	(none)	Requests the priority and the number of pending frames in the asynchronous send queue with the highest priority from the ATC. CN: This primitive is called just before sending the PRes DLPDU and StatusResponse ASnd DLPDU. MN: This primitive is called just before sending the SoA DLPDU.
ATC-Prio.ind	ATC	Priority RequestToSend	Indicates the priority and the number of pending frames in the asynchronous send queue with the highest priority. CN: The parameters are transmitted via the PRes and StatusResponse ASnd DLPDU. The MNs CSM carries this parameter forward to the ASS via CSM-Prio. MN: The MNs internal CSM carries forward this parameter to the ASS via CSM-Prio.

The parameters used with the primitives exchanged between the CSM and the ATC are described in Table 20.

Table 20 – Parameters used with primitives exchanged between CSM and ATC

Parameter name	Description
Destination	This parameter indicates the Node ID of the addressed node.
Source	This parameter indicates the Node ID of the transmitting node.
ServiceID	This parameter indicates the ServiceID dedicated to the asynchronous slot.
Payload	This parameter indicates the user data.
Priority	This parameter indicates the priority of the user data in the asynchronous send queue with the highest priority.
RequestToSend	This parameter indicates the number of pending user data in the asynchronous send queue with the highest priority.

7.2.2.3 Primitive definitions between CSM and ASS

Table 21 summarizes all primitives exchanged between the CSM and the ASS.

Table 21 – Primitives exchanged between CSM and ASS

Primitive name	Source	Associated parameters	Description
CSM-CSM.ind	CSM	Resp-Expected Async-In Asycn-Out	Indicates the assignment of the next asynchronous slot
CSM-Prio.ind	CSM	Source Priority RequestToSend	Indicates the received priority and Request to Send parameter from the specified source
CSM-SoA.req	CSM	(none)	Request the service and node, which shall transmit the ASnd frame inside the next asynchronous slot
CSM-SoA.ind	ASS	Requested-ServiceID Requested-ServiceTarget	Indicates the type of user data with the ServiceID parameter and the addressed node with the ServiceTarged parameter, who shall send on the next asynchronous slot. The parameters are transmitted via the pending SoA DLPDU

The parameters used with the primitives exchange between the CSM and the ASS are described in Table 22.

Table 22 – Parameters used with primitives exchanged between CSM and ASS

Parameter name	Description
RequestedServiceID	This parameter indicates the asynchronous service ID dedicated to the following asynchronous slot
RequestedServiceTarget	This parameter indicates the Node ID of the node, which shall send within the following asynchronous slot
Resp-Expected	Resp-expected = “TRUE” means that there is an ASnd DLPDU expected upon the next asynchronous slot
Async-In	This indicates, that a SoA DLPDU must be sent in this cycle and an ASnd DLPDU or a non Type 13 fieldbus frame could be received
Async-Out	This indicates, that a SoA DLPDU must be sent in this cycle and an ASnd DLPDU must be sent in this cycle
Priority	This parameter indicates the priority of the user data in the asynchronous send queue with the highest priority
RequestToSend	This parameter indicates the number of pending user data in the asynchronous send queue with the highest priority

7.2.2.4 Primitive definitions between CSM and ES

Table 23 summarizes all primitives exchanged between the CSM and the ES.

Table 23 – Primitives exchanged between CSM and ES

Primitive name	Source	Associated parameters	Description
CSM-TERR.ind	ES	Destination ER EC EN EA	Indication of the exception flags. MN: Only the flags ER and EA are passed to the CSM. CN: Only the flags EC and EN are passed to the CSM
CSM-RERR.ind	CSM	Source ER (SoA) EC (Status) EN (PRes/Stat) EA (PReq/Stat)	Carries forward the received exception flags to the ES every time a new flag has been received. MN: Only the flags EC and EN are passed to the ES. CN: Only the flags ER and EA are passed to the ES

The parameters used with the primitives exchange between the CSM and the ES are described in Table 24.

Table 24 – Parameters used with primitives exchanged between CSM and ES

Parameter name	Description
Source	This parameter indicates the node ID of the transmitting node
Destination	This parameter indicates the node ID of the dedicated node
ER	Exception reset: Initialization of the exception signaling This parameter is set by the MN to reset the internal exception flag from the addressed CN
EC	Exception clear: acknowledge of the exception signaling initialization. This parameter mirrors the last received ER flag from the MN to indicate the MN that the initialization of the exception signaling was done
EN	Exception new: signaling a new exception This parameter is toggled by the CN to indicate an exception to the MN
EA	Exception acknowledge: This parameter is toggled by the MN to acknowledge a requested exception signal from the addressed CN

7.2.2.5 Primitive definitions between CSM and NS

Table 25 summarizes all primitives exchanged between the CSM and the NS.

Table 25 – Primitives exchanged between CSM and NS

Primitive name	Source	Associated parameters	Description
CSM-TNMT.ind	NS	NMTStatus	Indication of the own NMTStatus parameter.
CSM-RNMT.ind	CSM	Source NMTStatus	Carries forward the received NMTStatus parameter to the NS every time a new NMTstatus has received. MN: The MNs NMTStatus parameter is transmitted via the SoA DLPDU. CN: The CNs NMTStatus parameter is transmitted via the PRes DLPDU, StatusResponse ASnd DLPDU and IdentResponse ASnd DLPDU

The parameters used with the primitives exchange between the CSM and the NS are described in Table 26.

Table 26 – Parameters used with primitives exchanged between CSM and NS

Parameter name	Description
Source	This parameter indicates the node ID of the transmitting node
NMTStatus	Current status of the NMT state machine

7.2.2.6 Primitive definitions between CSM and DLM

Table 27 summarizes all primitives exchanged between the CSM and the DLM.

Table 27 – Primitives exchanged between CSM and DLM

Primitive name	Source	Associated parameters	Description
CSM-Reset.ind	DLM	(none)	Indicates a reset of the CSM from the DLS-user
CSM-Frame.ind	CSM	DLM-frame-identifier MC PS time reltime	Indicates the actual received DLPDU type to the DLS-user
CSM-Event.ind	CSM	DLM-event-identifier EntryType TimeStamp Additional-Information	Indicates the new error status to the DLS-user

The parameters used with the primitives exchange between the CSM and the DLM are described in Table 28.

Table 28 – Parameters used with primitives exchanged between CSM and DLM

Parameter name	Description
DLM-frame-identifier	This parameter specifies the primitive within the DLE whose occurrence is being announced. The possible values are defined in the corresponding part of this part of IEC 61158
MC	This parameter is toggled by the MN when the final multiplexed cycle has ended
PS	This parameter is toggled by the MN every n-th cycle
time	This parameter is distributed by the MN and indicates the starting time of the Type 13 fieldbus cycle
reltime	This parameter is distributed by the MN and indicates the relative time, which is incremented by the Type 13 fieldbus cycle time when a SoC DLPDU is generated
DLM-event-identifier	Detailed error description about the occurred error
EntryType	Mode and profile information about the occurred error
TimeStamp	Nettime from the Type 13 fieldbus cycle when the error/event was detected
AdditionalInformation	This field contains device profile or vendor specific additional error information

7.2.3 CSM state table

7.2.3.1 CSM state table at MN

7.2.3.1.1 Overview

The cycle state machine of the MN shall manage the communication within a Type 13 fieldbus cycle.

The MNs CSM generates the flow of the frames during a Type 13 fieldbus cycle and monitors the reaction of the CNs. The flow order is dependent of the nodes NMTStatus parameter.

Usually the CNs are synchronized by the reception of the SoC DLPDU. This means the most significant parameter for the synchronization of the Type 13 fieldbus network is the timing accuracy of the function ME_SOC_TRIG.

If an error in the communication is detected by the CSM, an error primitive will be generated.

7.2.3.1.2 Overall state table

The transitions of CSM could be displayed within a single diagram where the NMTStatus parameter, received via the DL-NMT primitive, are conditions for the transitions. Because of comprehension and clarity purposes, the relevant transitions of single NMTStatus values are filtered out and displayed within a separate diagram as an “operation mode” of the CSM. The state transition diagram in Figure 7 shows the reaction on NMTStatus and the activation of the underlying state machines.

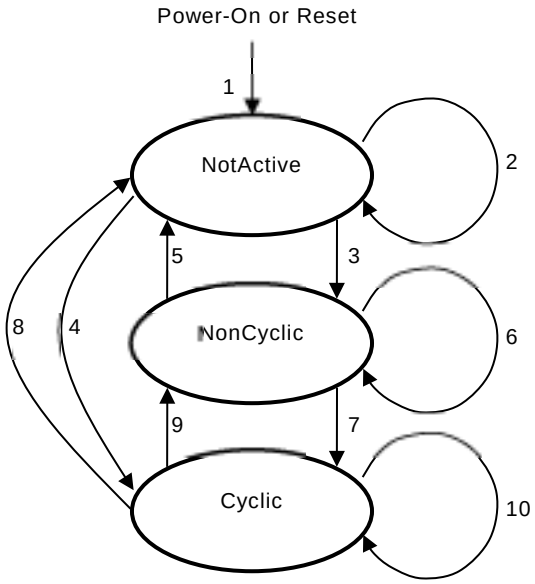


Figure 7 – State transition diagram of the MNs CSM

The state “NotActive” means that no underlying state machine is active and no reaction on received frames occurs.

The state “NonCyclic” means, the MNs CSM generates the reduced Type 13 fieldbus Cycle and observes the behavior of the CNs. The MNs CSM is in the DLL_MS_NON_CYCLIC mode.

The state “Cyclic” means, the MNs CSM generates the Type 13 fieldbus Cycle and observes the behavior of the CNs.

The transitions are described in Table 29.

Table 29 – Transitions of the MNs CSM

#	Current state	Event /condition ⇒actions	Next state
1	Any states	POWER-ON or RESET =>	NotActive
2	NotActive	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_GS_INITIALISATION || NMTStatus = NMT_MS_NOT_ACTIVE || NMTStatus = NMT_MS_BASIC_ETHERNET => -- Deactivate CSM_MS_NON_CYCLIC State Machine -- Deactivate CSM_MS_CYCLIC State Machine	NotActive
3	Not Active	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_MS_PRE_OPERATIONAL_1 => -- Activate CSM_MS_NON_CYCLIC State Machine -- Deactivate CSM_MS_CYCLIC State Machine	NonCyclic
4	Not Active	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_MS_OPERATIONAL || NMTStatus =NMT_MS_READY_TO_OPERATE || NMTStatus =NMT_MS_PRE_OPERATIONAL_2 => -- Deactivate CSM_MS_NON_CYCLIC State Machine -- Activate CSM_MS_CYCLIC State Machine	Cyclic
5	NonCyclic	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_GS_INITIALISATION || NMTStatus = NMT_MS_NOT_ACTIVE || NMTStatus = NMT_MS_BASIC_ETHERNET => -- Deactivate CSM_MS_NON_CYCLIC State Machine -- Deactivate CSM_MS_CYCLIC State Machine	NotActive
6	NonCyclic	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_MS_PRE_OPERATIONAL_1 => -- Activate CSM_MS_NON_CYCLIC State Machine -- Deactivate CSM_MS_CYCLIC State Machine	NonCyclic
7	NonCyclic	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_MS_OPERATIONAL || NMTStatus =NMT_MS_READY_TO_OPERATE || NMTStatus =NMT_MS_PRE_OPERATIONAL_2 => -- Deactivate CSM_MS_NON_CYCLIC State Machine -- Activate CSM_MS_CYCLIC State Machine	Cyclic
8	Cyclic	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_GS_INITIALISATION || NMTStatus = NMT_MS_NOT_ACTIVE || NMTStatus = NMT_MS_BASIC_ETHERNET => -- Deactivate CSM_MS_NON_CYCLIC State Machine -- Deactivate CSM_MS_CYCLIC State Machine	NotActive
9	Cyclic	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_MS_PRE_OPERATIONAL_1 => -- Activate CSM_MS_NON_CYCLIC State Machine -- Deactivate CSM_MS_CYCLIC State Machine	NonCyclic
10	Cyclic	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_MS_OPERATIONAL || NMTStatus =NMT_MS_READY_TO_OPERATE || NMTStatus =NMT_MS_PRE_OPERATIONAL_2 => -- Deactivate CSM_MS_NON_CYCLIC State Machine -- Activate CSM_MS_CYCLIC State Machine	Cyclic

7.2.3.1.3 CSM_MS_NON_CYLIC

The state transition diagram is shown in Figure 8.

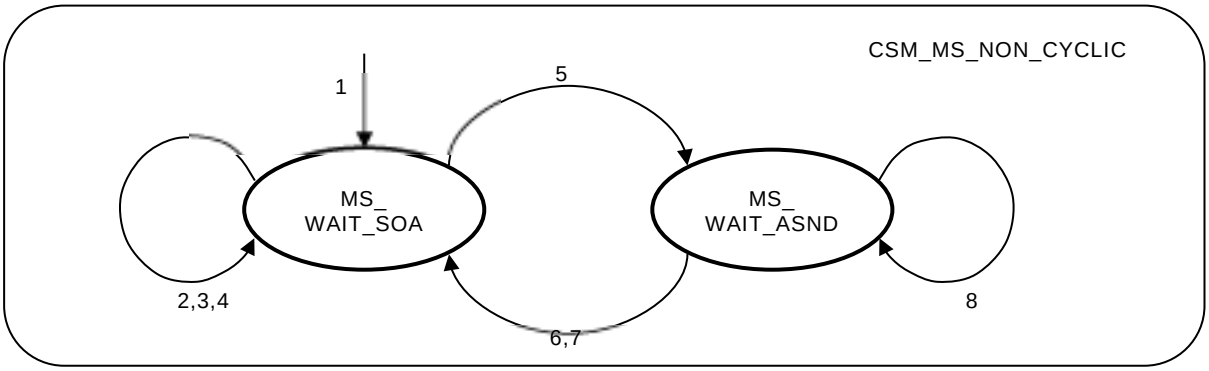


Figure 8 – State transition diagram of MNs CSM at CSM_MS_NON_CYCLIC

The state “NON_CYCLIC” means that the cyclic communication has not started yet or was stopped by the NMTStatus parameter (NMTStatus = NMT_MS_PRE_OPERATIONAL_1). The state machine waits here until the NMTStatus changes to NMT_MS_PRE_OPERATIONAL_2. It depends on the current NMTStatus parameter, which events will be processed and which will be ignored.

The CSM remains in the state “MS_WAIT_ASND” until any Ethernet frame is received, the timeout of the asynchronous phase has elapsed or the next reduced cycle begins with the ME_SOA_TRIG() = “TRUE”.

The transitions are described in Table 30.

Table 30 – Transitions of MNs CSM at CSM_MS_NON_CYCLIC

#	Current state	Event /condition ⇒actions	Next state
1	Any states	Activation / =>	MS_WAIT_SOA
2	MS_WAIT_SOA	CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \ ME_SOA_TRIG ( ) = ”TRUE“ && Async-In <> 0 && Resp-Expected = “FALSE” => SOADU := BUILD_SOA ( ) MA_Data.req { SOADU }	MS_WAIT_SOA
3	MS_WAIT_SOA	CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \ ME_SOA_TRIG ( ) = ”TRUE“ && Async-In = 0 && Async-Out <> 0 => SOADU := BUILD_SOA ( ) MA_Data.req { SOADU } ASNDDU := BUILD_ASND (C_ADDR_MN_DEF_NODE_ID) MA_Data.req { ASNDDU }	MS_WAIT_SOA
4	MS_WAIT_SOA	CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \ ME_SOA_TRIG ( ) = ”TRUE“ && Async-In = 0 && Asynch-Out = 0 => SOADU := BUILD_SOA ( ) MA_Data.req { SOADU }	MS_WAIT_SOA

#	Current state	Event /condition ⇒actions	Next state
5	MS_WAIT_SOA	CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \ ME_SOA_TRIG ( ) = ”TRUE“ && Async-In <> 0 && Resp-Expected = “TRUE” => SOADU := BUILD_SOA ( ) MA_Data.req { SOADU } START_TIMER(T(ASND_TIME), V(AsyncSlotTimeout_U32))	MS_WAIT_ASND
6	MS_WAIT_ASND	CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \ (ME_SOA_TRIG () = “TRUE” || EXPIRED_TIMER (T(ASND_TIME)) = "True") && Async-In = 0 => SOADU := BUILD_SOA ( ) MA_Data.req { SOADU } ASNDDU := BUILD_ASND (C_ADDR_MN_DEF_NODE_ID) MA_Data.req { ASNDDU }	MS_WAIT_SOA
7	MS_WAIT_ASND	CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \ (ME_SOA_TRIG () = “TRUE” || EXPIRED_TIMER (T(ASND_TIME)) = "True") && Async-In <> 0 && Resp-Expected = “FALSE” => SOADU := BUILD_SOA ( ) MA_Data.req { SOADU }	MS_WAIT_SOA
8	MS_WAIT_ASND	CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \ ME_SOA_TRIG () = “TRUE” || EXPIRED_TIMER (T(ASND_TIME)) =”TRUE) Async-In <> 0 && Resp-Expected = “TRUE” => SOADU := BUILD_SOA ( ) MA_Data.req { SOADU }	MS_WAIT_ASND

7.2.3.1.4 CSM_MS_CYCLIC

The state transition diagram is shown in Figure 9.

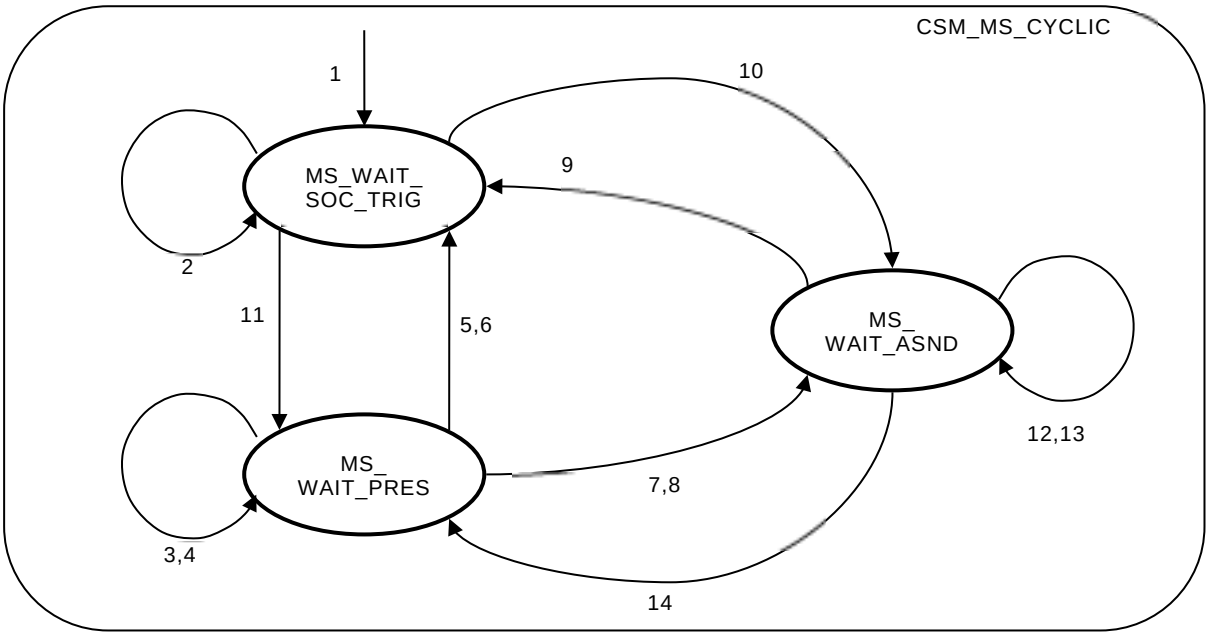


Figure 9 – State transition diagram of MNs CSM at CSM_MS_CYCLIC

The state “CYCLIC” means that the cyclic communication is in progress. (NMTStatus = NMT_MS_PRE_OPERATIONAL_2, NMT_MS_READY_TO_OPERATE or NMT_MS_OPERATIONAL).

After the sending of a PReq frame the state machine waits in the state “MS_WAIT_PRES” for a response. The waiting time is limited by a timeout.

If a SoA frame with an invite is sent, the state machine waits the state “MS_WAIT_ASND” until the asynchronous phase ends with the ME_SOC_TRIG( ) = “TRUE”.

The transitions are described in Table 31.

Table 31 – Transitions of MNs CSM at CSM_MS_CYCLIC

#	Current state	Event /condition ⇒actions	Next state
1	Any states	Activation \ =>	MS_WAIT_SOC_TRIG

2	MS_WAIT_SOC_TRIG	CSM-PDO.ind { Isochr, Isochr-time-triggered, Isochr-Out} /
		Isochr = 0 &&
		Async-In = 0 &&
		ME_SOC_TRIG ( ) = “TRUE”
		=>
		SOCDU := BUILD_SOC ( )
		MA_Data.req { SOCDU }
		IF Isochr-Out <> 0 THEN
		PRESDU := BUILD_PRES ( )
		MA_DATA.req { PRESDU }
		END IF
		SOADU := BUILD_SOA ( )
		MA_DATA.req { SOADU }
		IF Async-Out <> 0 THEN
		ASNDDU := BUILD_ASND ( )
		MA_DATA.req { ASNDDU }
		ENDIF
			MS_WAIT_SOC_TRIG

3	MS_WAIT_PRES	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} /
		Isochr <> 0 &&
		ME_PRES ( ) = “TRUE”
		=>
		IF Isochr-time-triggered = 0 THEN
		PREQDU := BUILD_PREQ ( )
		MA_DATAreq { PREQDU }
		ENDIF
			MS_WAIT_PRES

4	MS_WAIT_PRES	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} /
		Isochr <> 0 &&
		EXPIRED_TIMER (T(PRES_TIME)) = "TRUE")
		=>
		IF Isochr-time-triggered = 0 THEN
		PREQDU := BUILD_PREQ ( )
		MA_DATA.req { PREQDU }
		ENDIF
		CSM-Event.ind
		{E_DLL_LOSS_PRES,3002h,time,AdditionalInfo}
			MS_WAIT_PRES

5	MS_WAIT_PRES	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} /
		CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \
		ME_PRES ( ) = “TRUE” &&
		Isochr = 0 &&
		Async-In = 0
		=>
		SOADU := BUILD_SOA ( )
		MA_DATA.req { SOADU }
		IF Async-Out <> 0 THEN
		ASNDDU := BUILD_ASND ( )
		MA_DATA.req { ASNDDU }
		ENDIF
			MS_WAIT_SOC_TRIG

6	MS_WAIT_PRES	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} / CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \ EXPIRED_TIMER (T(PRES_TIME)) = "True") && Isochr = 0 && Async-In = 0 => SOADU := BUILD_SOA ( ) MA_DATA.req { SOADU } IF Async-Out <> 0 THEN ASNDDU := BUILD_ASND ( ) MA_DATA.req { ASNDDU }	MS_WAIT_SOC_TRIG

#	Current state	Event /condition ⇒actions	Next state
		ENDIF CSM-Event.ind {E_DLL_LOSS_PRES,3002h,time,AdditionalInfo}	
7	MS_WAIT_PRES	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} / CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \ ME_PRES ( ) = “TRUE” && Isochr = 0 && Async-In <> 0 => SOADU := BUILD_SOA ( ) MA_DATA.req { SOADU }	MS_WAIT_ASND

8	MS_WAIT_PRES	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} /
		CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \
		EXPIRED_TIMER (T(PRES_TIME)) = "True") &&
		Isochr = 0 &&
		Async-In <> 0
		=>
		IF Isochr-Out <> 0 THEN
		PRESDU := BUILD_PRES ( )
		MA_DATA.req { PRESDU }
		ENDIF
		SOADU := BUILD_SOA ( )
		MA_DATA.req { SOADU }
		CSM-Event.ind
		{E_DLL_LOSS_PRES,3002h,time,AdditionalInfo}
			MS_WAIT_ASND

9	MS_WAIT_ASND	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} /
		CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \
		ME_SOC_TRIG ( ) = “TRUE” &&
		Isochr = 0 &&
		Async-In = 0
		=>
		SOCDU := BUILD_SOC ( )
		MA_DATA.req { SOCDU }
		IF Isochr-Out <> 0 THEN
		PRESDU := BUILD_PRES ( )
		MA_DATA.req { PRESDU }
		ENDIF
		SOADU := BUILD_SOA ( )
		MA_DATA.req { SOADU }
		IF Async-Out <> 0 THEN
		ASNDDU := BUILD_ASND ( )
		MA_DATA.req { ASNDDU }
		ENDIF
			MS_WAIT_SOC_TRIG

10	MS_WAIT_SOC_TRIG	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} /
		CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \
		ME_SOC_TRIG ( ) = “TRUE” &&
		Isochr = 0 &&
		Async-In <> 0
		=>
		SOCDU := BUILD_SOC ( )
		MA_DATA.req { SOCDU }
		IF Isochr-Out <> 0 THEN
		PRESDU := BUILD_PRES ( )
		MA_DATA.req { PRESDU }
		ENDIF
		SOADU := BUILD_SOA ( )
		MA_DATA.req { SOADU }
			MS_WAIT_ASND

11	MS_WAIT_SOC_TRIG	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} / ME_SOC_TRIG ( ) = “TRUE” && Isochr <> 0 && => SOCDU := BUILD_SOC ( ) MA_DATA.req { SOCDU } IF Isochr-Out <> 0 THEN	MS_WAIT_PRES

#	Current state	Event /condition ⇒actions	Next state

		PRESDU := BUILD_PRES ( )
		MA_DATA.req { PRESDU }
		END IF
		IF Isochr-time-triggered = 0 THEN
		PREQDU := BUILD_PREQ ( )
		MA_DATAreq { PREQDU }
		ENDIF	

12	MS_WAIT_ASND	\ ME_ASND ( ) = “TRUE” => CSM-RASND.ind { }	MS_WAIT_ASND

13	MS_WAIT_ASND	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} /
		CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \
		ME_SOC_TRIG ( ) = “TRUE” &&
		Isochr = 0 &&
		Async-In <> 0
		=>
		SOCDU := BUILD_SOC ( )
		MA_DATA.req { SOCDU }
		IF Isochr-Out <> 0 THEN
		PRESDU := BUILD_PRES ( )
		MA_DATA.req { PRESDU }
		ENDIF
		SOADU := BUILD_SOA ( )
		MA_DATA.req { SOADU }
			MS_WAIT_ASND

14	MS_WAIT_ASND	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} /
		CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \
		ME_SOC_TRIG ( ) = “TRUE” &&
		Isochr <> 0
		=>
		SOCDU := BUILD_SOC ( )
		MA_DATA.req { SOCDU }
		IF Isochr-Out <> 0 THEN
		PRESDU := BUILD_PRES ( )
		MA_DATA.req { PRESDU }
		END IF
		IF Isochr-time-triggered = 0 THEN
		PREQDU := BUILD_PREQ ( )
		MA_DATA.req { PREQDU }
		ENDIF
			MS_WAIT_PRES

7.2.3.2 CSM state table at CN

7.2.3.2.1 Overview

The cycle state machine of the CN handles communication within a Type 13 fieldbus cycle. The CSM tracks the order of the frames received within a cycle and reacts as described below. The expected order of frame reception is dependent of the NMTStatus parameter.

If an error in the communication is detected by the CSM, an error primitive will be generated. The CSM will attempt to uphold communication regardless of any errors.

7.2.3.2.2 Overall state table

The transitions of CSM could be displayed within a single diagram where the NMTStatus parameter, received via the DL-NMT primitive, are conditions for the transitions. Because of comprehension and clarity purposes, the relevant transitions of single NMTStatus value are filtered out and displayed within a separate diagram as an “operation mode” of the CSM. The state transition diagram in Figure 10 shows the reaction on NMTStatus and the activation of the underlying state machines.

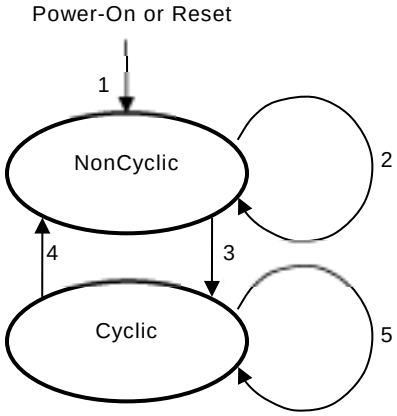


Figure 10 – State transition diagram of the CNs CSM

The state “NonCyclic” means that the isochronous communication hasn’t started yet or the connection was lost.

The state “Cyclic” means, the CNs CSM is locked into the MNs frame sequence.

The transitions are described in Table 32.

Table 32 – Transitions of the CNs CSM

#	Current state	Event /condition ⇒actions	Next state
1	Any states	POWER-ON or RESET =>	NonCyclic
2	NonCyclic	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_GS_INITIALISATION || NMTStatus = NMT_CS_NOT_ACTIVE || NMTStatus = NMT_CS_BASIC_ETHERNET || NMTStatus = NMT_CS_PRE_OPERATIONAL_1 => -- Activate CSM_CS_NON_CYCLIC State Machine -- Deactivate CSM_CS_CYCLIC State Machine	NonCyclic
3	NonCyclic	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_CS_PRE_OPERATIONAL_2 || NMTStatus = NMT_CS_READY_TO_OPERATE || NMTStatus = NMT_CS_OPERATIONAL => -- Deactivate CSM_CS_NON_CYCLIC State Machine -- Activate CSM_CS_CYCLIC State Machine	Cyclic
4	Cyclic	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_GS_INITIALISATION || NMTStatus = NMT_CS_NOT_ACTIVE || NMTStatus = NMT_CS_BASIC_ETHERNET || NMTStatus = NMT_CS_PRE_OPERATIONAL_1 => -- Activate CSM_CS_NON_CYCLIC State Machine -- Deactivate CSM_CS_CYCLIC State Machine	NonCyclic
5	Cyclic	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_CS_PRE_OPERATIONAL_2 || NMTStatus = NMT_CS_READY_TO_OPERATE || NMTStatus = NMT_CS_OPERATIONAL => -- Deactivate CSM_CS_NON_CYCLIC State Machine -- Activate CSM_CS_CYCLIC State Machine	Cyclic

7.2.3.2.3 CSM_CS_NON_CYCLIC

The state transition diagram is shown in Figure 11.

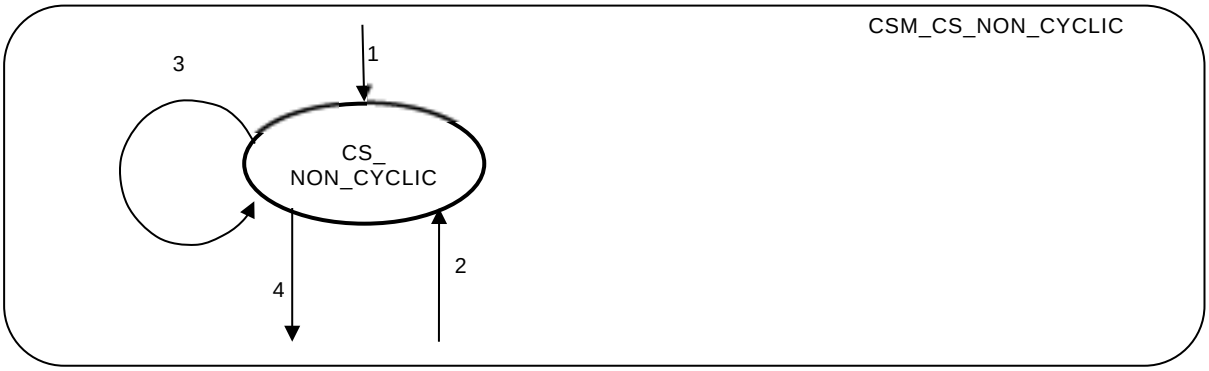


Figure 11 – State transition diagram of CNs CSM at CSM_CS_NON_CYCLIC

The transitions are described in Table 33.

Table 33 – Transitions of CNs CSM at CSM_CS_NON_CYCLIC

#	Current state	Event /condition ⇒actions	Next state
1	Any states	Activation =>	CS_NON_CYCLIC
2	CS_WAIT_SOC, CS_WAIT_PREQ, CS_WAIT_SOA	/(CE_SOC ( ) = “TRUE” || CE_PREQ ( ) = “TRUE” || CE_SOA ( ) = “TRUE”) && EXPIRED_TIMER (T(FRAME_TIME)) = "True") => -- NMT state specific reaction --	CS_NON_CYCLIC
3	CS_NON_CYCLIC	/CE_SOA ( ) = “TRUE” || CE_ASND ( ) = “TRUE” => -- NMT state specific reaction --	CS_NON_CYCLIC
4	CS_NON_CYCLIC	/CE_SOC ( ) = “TRUE => -- NMT state specific reaction --	CS_WAIT_PREQ

7.2.3.2.4 CSM_CS_CYCLIC

For NMTStatus = NMT_CS_OPERATIONAL and NMT_CS_READY_TO_OPERATE, there are three mandatory frames for a continuous node, which shall occur each cycle in the specified order: SoC, PReq and SoA. If a node is accessed multiplexed, only the SoC and SoA frames are mandatory for every cycle. The PReq frame is only mandatory in the multiplexed cycle the node was configured for.

For NMTStatus = NMT_CS_PRE_OPERATIONAL_2, there are two mandatory frames, which shall occur each cycle in the order SoC and SoA. The PReq frame may occur between SoC and SoA. In the NMT_CS_PRE_OPERATIONAL_2 state the timeout detection of the SoC is disabled because the node may not be configured yet.

The cycle state machine keeps track of all received frames to detect frame losses. Receive frames shall be accepted by the CN independent of the current CSM state.

The state transition diagram is shown in Figure 12.

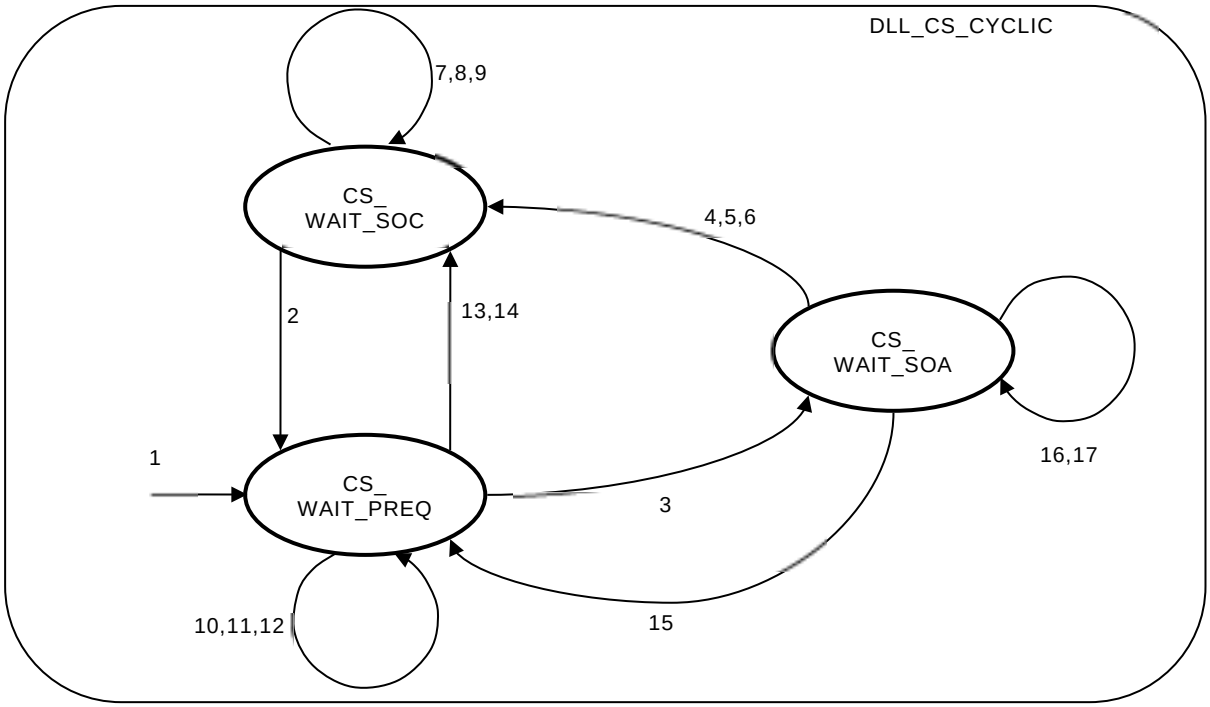


Figure 12 – State transition diagram of CNs CSM at CSM_CS_CYCLIC

The state machine waits in the state “CS_WAIT_SOC” after receiving the SoA frame until the beginning of the next cycle (triggered by a SoC frame from the MN). Ethernet frames of any type may be received between the SoA and the SoC frames (asynchronous phase).

After the beginning of the cycle, the state machine waits in the state “CS_WAIT_PREQ” for a PReq frame. After PReq reception the CN shall respond with a PRes Frame. The CN may receive and process PRes Frames from other CN whilst in this state.

After reception of a PReq frame the state machine waits in the state “CS_WAIT_SOA” for the reception of a SoA frame. Reception of a SoA frame confirms the end of the isochronous phase. The CN may receive and process PRes frames from other nodes whilst in this state.

The transitions are described in Table 34.

Table 34 – Transitions of CNs CSM at CSM_CS_CYCLIC

#	Current state	Event /condition ⇒actions	Next state
1	Any states	Activation =>	CS_WAIT_PREQ
2	CS_WAIT_SOC	/CE_SOC ( ) = “TRUE” => -- synchronize the start of cycle -- DLM-Frame.ind { SOC }	CS_WAIT_PREQ
3	CS_WAIT_PREQ	/CE_PREQ ( ) = “TRUE” || EXPIRED_TIMER (T(PRES_TT_TIME)) = “TRUE” => IF CE_PREQ ( ) CSM-RPDO.ind { } ENDIF PRESDU := BUILD_PRES ( ) MA_DATA.req { PRESDU }	CS_WAIT_SOA
4	CS_WAIT_SOA	/CE_SOA ( ) = “TRUE” =>	CS_WAIT_SOC

#	Current state	Event /condition ⇒actions	Next state
		CSM-TASND.ind { } -- process SoA -- ASNDDU := BUILD_ASND ( ) MA_DATA.req { ASNDDU}	
5	CS_WAIT_SOA	/EXPIRED_TIMER (T(FRAME_TIME)) = "True") => -- Synchronize to the next SoC CSM-Event.ind {E_DLL_LOSS_SOC,3002h,time,AdditionalInfo} CSM-Event.ind {E_DLL_LOSS_SOA,3002h,time,AdditionalInfo}	CS_WAIT_SOC
6	CS_WAIT_SOA	/CE_PREQ ( ) = “TRUE” || EXPIRED_TIMER (T(PRES_TT_TIME)) = “TRUE” => IF CE_PREQ ( ) CSM-RPDO.ind { } -- accept the PReq frame -- ENDIF PRESDU := BUILD_PRES ( ) MA_DATA.req { PRESDU } CSM-Event.ind {E_DLL_LOSS_SOC,3002h,time,AdditionalInfo} CSM-Event.ind {E_DLL_LOSS_SOA,3002h,time,AdditionalInfo}	CS_WAIT_SOC
7	CS_WAIT_SOC	/CE_ASND ( ) = “TRUE” => CSM-RASND.ind { }	CS_WAIT_SOC
8	CS_WAIT_SOC	/CE_PREQ ( ) = “TRUE” || EXPIRED_TIMER (T(PRES_TT_TIME)) = “TRUE”) => PRESDU := BUILD_PRES ( ) MA_DATA.req { PRESDU } CSM-Event.ind {E_DLL_LOSS_SOC,3002h,time,AdditionalInfo}	CS_WAIT_SOC
9	CS_WAIT_SOC	/CE_PRES ( ) = “TRUE” || /CE_SOA ( ) = “TRUE” || /EXPIRED_TIMER (T(FRAME_TIME)) = "True") => CSM-Event.ind {E_DLL_LOSS_SOC,3002h,time,AdditionalInfo}	CS_WAIT_SOC
10	CS_WAIT_PREQ	/CE_PRES ( ) = “TRUE” => CSM-RPDO.ind { } -- process PRes (cross traffic) --	CS_WAIT_PREQ
11	CS_WAIT_PREQ	/CE_SOC ( ) = “TRUE” => -- synchronize to the cycle begin -- CSM-Event.ind {E_DLL_LOSS_SOA,3002h,time,AdditionalInfo}	CS_WAIT_PREQ
12	CS_WAIT_PREQ	/CE_ASND ( ) = “TRUE” => CSM-Event.ind {E_DLL_LOSS_SOA,3002h,time,AdditionalInfo}	CS_WAIT_PREQ

13	CS_WAIT_PREQ	/CE_SOA ( ) = “TRUE”
		=>
		CSM-TASND.ind { } -- process SoA --
		ASNDDU := BUILD_ASND ( )
		MA_DATA.req { ASNDDU }
		IF CN <> MULTIPLEXED THEN
		CSM-Event.ind
		{E_DLL_LOSS_PREQ,3002h,time,AdditionalInfo}
		ENDIF
			CS_WAIT_SOC

14	CS_WAIT_PREQ	/EXPIRED_TIMER (T(FRAME_TIME)) = "True") => -- Synchronize on the next SoC CSM-Event.ind {E_DLL_LOSS_SOC,3002h,time,AdditionalInfo} CSM-Event.ind	CS_WAIT_SOC

#	Current state	Event /condition ⇒actions	Next state
		{E_DLL_LOSS_SOA,3002h,time,AdditionalInfo}	
15	CS_WAIT_SOA	/CE_SOC ( ) = “TRUE” => -- Synchronize on the next SoC CSM-Event.ind {E_DLL_LOSS_SOA,3002h,time,AdditionalInfo}	CS_WAIT_PREQ
16	CS_WAIT_SOA	/CE_PRES ( ) = “TRUE” => CSM-RPDO.ind { } -- process PRes (cross traffic) --	CS_WAIT_SOA
17	CS_WAIT_SOA	/CE_ASND ( ) = “TRUE” => CSM-Event.ind {E_DLL_LOSS_SOA,3002h,time,AdditionalInfo}	CS_WAIT_SOA

7.2.4 Functions of CSM

All the functions used by the CSM are summarized in Table 35.

Table 35 – CSM function table

Function name	Inputs	Outputs	Description and operation

BUILD_ASND	Source	ASNDDU	ASNDDU is assembled as follows
			ASNDDU.Destination coming from CSM-TASND.ind
			ASNDDU.Source := source
			ASNDDU.ServiceID coming from CSM-TASND.ind
			ASNDDU.Payload coming from CSM-TASND.ind
			
			IF ASNDDU.ServiceID = IDENT THEN
			ASNDDU.Payload.PR coming from ATC-Prio.ind
			ASNDDU.Payload.RS coming from ATC-Prio.ind
			ASNDDU.Payload.NMTStatus coming from CSM-TNMT.ind
			ASNDDU.Payload.Type 13 fieldbusVersion :=
			V(NMT_Type 13 fieldbusVersion_U8)
			ENDIF
			IF ASNDDU.ServiceID = STATUS then
			ASNDDU.Payload.PR coming from ATC-Prio.ind
			ASNDDU.Payload.RS coming from ATC-Prio.ind
			ASNDDU.Payload.NMTStatus coming from CSM-TNMT.ind
			ASNDDU.Payload.EN coming from CSM-TERR.ind
			ASNDDU.Payload.EC coming from CSM-TERR.ind
			ENDIF

BUILD_PREQ	(none)	PREQDU	PREQDU is assembled as follows PREQDU.Destination := coming from CSM-TPDO.ind PREQDU.Source := C_ADDR_MN_DEF_NODE_ID PREQDU.MS coming from ME_SOC_TRIG ( ) PREQDU.EA coming from CSM-TERR.ind PREQDU.RD coming from CSM-TPDO.ind PREQDU.Payload coming from CSM-TPDO.ind
BUILD_PRES	(none)	PRESDU	PRESDU is assembled as follows PRESDU.Destination C_ADDR_BROADCAST PRESDU.Source := NodeID PRESDU.NMTStatus coming from CSM-TNMT.ind PRESDU.MS coming from ME_SOC_TRIG ( ) PRESDU.EN coming from CSM-TERR.ind PRESDU.RD coming from CSM-TPDO.ind PRESDU.PR coming from ATC-Prio.ind PRESDU.RS coming from ATC-Prio.ind PRESDU.Payload coming from CSM-TPDO.ind
BUILD_SOA	(none)	SOADU	SOADU is assembled as follows SOADU.Destination := C_ADDR_BROADCAST SOADU.Source := C_ADDR_MN_DEF_NODE_ID SOADU.NMTStatus coming from CSM-TNMT.ind SOADU.EA coming from CSM-TERR.ind SOADU.ER coming from CSM-TERR.ind

Function name	Inputs	Outputs	Description and operation
			SOADU.RequestedServiceID coming from CSM-SoA.ind SOADU.RequestedServiceTarget coming from CSM-SoA.ind SOADU.Type 13 fieldbusVersion = V(NMT_Type 13 fieldbusVersion_U8) SOADU.ER := ERRDU.ER
BUILD_SOC	(none)	SOCDU	SOCDU is assembled as follows SOCDU.Destination := C_ADDR_BROADCAST SOCDU.Source := C_ADDR_MN_DEF_NODE_ID SOCDU.MC coming from ME_SOC_TRIG ( ) SOCDU.PS coming from ME_SOC_TRIG ( ) SOCDU.NetTime coming from ME_SOC_TRIG ( ) SOCDU.RelativeTime coming from ME_SOC_TRIG ( )
CE_ASND	(none)	True/False	This function signifies that an ASnd frame or a non Type 13 fieldbus frame has been received. Since the frame types during the asynchronous phase are not limited to Type 13 fieldbus types, this event is generated on reception of all legal Ethernet frames
CE_PREQ	(none)	True/False	This function signifies that a PReq frame was received from the MN
CE_PRES	(none)	True/False	The CN may be configured to process the PRes frames from other CNs or from the MN (cross traffic). Every time a PRes frame is received, a DLL_CE_PRES event is generated
CE_SOA	(none)	True/False	This function signifies that a SoA frame was received from the MN. It marks the end of the isochronous phase of the cycle and the beginning of the asynchronous phase
CE_SOC	(none)	True/False	This function signifies that a Type 13 fieldbus SoC frame was received from the MN. It marks the beginning of a new cycle and simultaneously the beginning of the isochronous phase of the cycle
ME_ASND	(none)	True/False	This function means that an ASnd frame or a non Type 13 fieldbus frame was received
ME_PRES	(none)	True/False	This function signifies that a PRes frame was received
ME_SOA_TRIG	(none)	True/False	This function means that a new reduced Type 13 fieldbus cycle shall start. The function can either be generated cyclically or directly after the reception of a requested ASnd frame to continue the reduced Type 13 fieldbus cycle as fast as possible
ME_SOC_TRIG	(none)	True/False time reltime	This function triggers emission of the SoC frame and starts a new Type 13 fieldbus cycle. The timing accuracy determines the synchronization accuracy of the Type 13 fieldbus network
EXPIRED_TIMER	T(x)	True/False	When requested timer x has expired, “True” is returned, otherwise False is returned
START_TIMER	T(x), V(y)	(none)	Timer x is set by value of y, and is activated

7.3 Isochronous transmission TX/RX control (ITC)

7.3.1 Overview

The real-time data transfer is performed by means of process data objects (PDO). PDO communication in Type 13 fieldbus is always performed isochronously by PReq and/or PRes frames. The PRes frames are sent as broadcasts or multicasts following the producer/consumer scheme. The PReq frames with unicast addresses comply with the master/slave relationship.

The transmission type of PDO is continuous. There is no “on event” or “on change” transmission type provided.

7.3.2 Multiplexed timeslots

7.3.2.1 General

Type 13 fieldbus supports CN communication classes that determine the cycles in which nodes are to be addressed during the isochronous phase:

- Continuous: continuous data shall be exchanged in every Type 13 fieldbus cycle.
- Continuous-time-triggered: Continuous-time-triggered data shall be exchanged in every Type 13 fieldbus cycle.
- Multiplexed: multiplexed data to and from one CN shall not be exchanged in every Type 13 fieldbus cycle.

The accesses to the multiplexed CNs shall be dispersed to the multiplexed cycle that consists of a number of Type 13 fieldbus cycles. The dispersion allows the isochronous access to a large number of CNs without elongating the Type 13 fieldbus cycle to an unacceptable amount. However, multiplexed CN access reduces the poll frequency to the particular CN.

The order in which CNs are polled is implementation specific. An implementation should pack the performed PReq / PRes DLPDUs to the beginning of the isochronous phase. It should provide means to rearrange the poll order, to avoid location of the nodes having the worst SoC latency time value at the slot following SoC.

7.3.2.2 Distribution of multiplexed timeslots

Continuous, continuous-time-triggered and multiplexed access may be operated in parallel during one Type 13 fieldbus cycle.

Figure 13 demonstrates the assignment of the multiplexed slots to CNs. The multiplexed slots are identified by “Mux 1” and “Mux 2”.

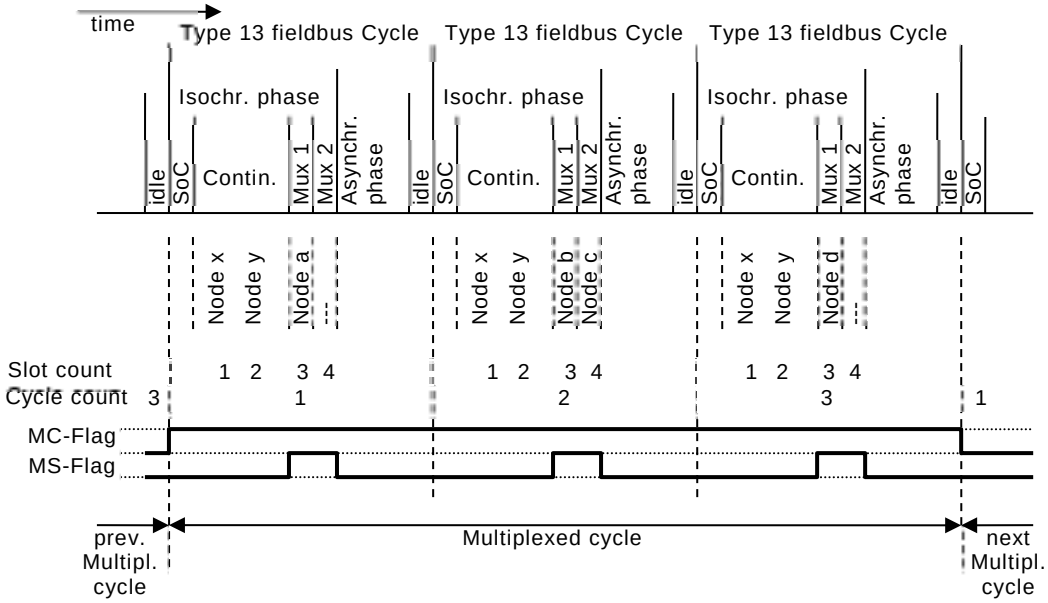


Figure 13 – Multiple slot assignment example

The ability of a MN enabled node to perform control of multiplexed isochronous operation shall be indicated by the device description item P(D_DLL_MNFeatureMultiplex_BOOL). The ability of a CN enabled node to be isochronously accessed in a multiplexed way shall be indicated by the device description item P(D_DLL_CNFeatureMultiplex_BOOL).

The assignment is controlled by V(MultiplCycleCnt_U8) and V(NMT_MultiplCycle-Assign_AU8). V(MultiplCycleCnt_U8) defines the length of the multiplexed cycle in Type 13 fieldbus cycle counts. If V(MultiplCycleCnt_U8) is zero, the multiplexed access method shall not be applied, e.g. all CNs shall be accessed continuously.

V(NMT_MultiplCycleAssign_AU8) defines the cycle count inside the multiplexed cycle, when the respective CN shall be polled by the MN. If the sub-index is zero, the CN shall be accessed continuously.

The number of isochronous slots per isochronous cycle is derived from the maximum number of CN assignments that are cumulated to a cycle.

V(NMT_IsochrSlotAssign_AU8) assigns CNs to a particular isochronous slot. The isochronous Type 13 fieldbus cycle can be divided into communication slots each consisting of PReq and PRes message for a particular node.

Each sub-index in the array corresponds to an individual communication slot which is equal to the sub-index. Value 0 indicates that there is no particular node assigned to the communication slot.

The sub-indices can also be used for the slot assignment of multiplexed nodes. If multiplexed and non-multiplexed nodes shall be assigned to particular communication slots, it must be ensured that all the isochronous non-multiplexed stations are configured on the lower subindices of V(NMT_IsochrSlotAssign_AU8). Care has also to be taken that the multiplex slots are mapped to the communication slots in ascending order. That means the first multiplexed node assigned to V(NMT_IsochrSlotAssign_AU8) must be configured in the first multiplex slot in V(NMT_MultiplCycleAssign_AU8) and so on.

Gaps in the NMT_IsochrSlotAssign_AU8 are allowed as unused communication slots are skipped.

The following table shows the parameter values that control the system in Figure 13:

Table 36 – Example of isochronous slot assignment

Variable	Value		Variable	Value
MultiplCycleCnt_U8	3		NMT_IsochrSlotAssign_AU8[0]	6
NMT_MultiplCycleAssign_AU8[$NodeID_a$]	1		NMT_IsochrSlotAssign_AU8[1 (Slot 1)]	$NodeID_x$
NMT_MultiplCycleAssign_AU8[$NodeID_b$]	2		NMT_IsochrSlotAssign_AU8[2 (Slot 2)]	$NodeID_y$
NMT_MultiplCycleAssign_AU8[$NodeID_c$]	2		NMT_IsochrSlotAssign_AU8[3 (Slot 3)]	$NodeID_a$
NMT_MultiplCycleAssign_AU8[$NodeID_d$]	3		NMT_IsochrSlotAssign_AU8[4 (Slot 4)]	$NodeID_b$
NMT_MultiplCycleAssign_AU8[$NodeID_x$]	0		NMT_IsochrSlotAssign_AU8[5 (Slot 5)]	$NodeID_c$
NMT_MultiplCycleAssign_AU8[$NodeID_y$]	0		NMT_IsochrSlotAssign_AU8[6 (Slot 6)]	$NodeID_d$

7.3.3 Time-triggered PRes

Instead of sending the PRes in response to a PReq, nodes of the communication class continuous-time-triggered send their PRes on a time triggered basis.

Therefore the MN shall send its PRes right after the SoC. This PRes from the MN is the reference for all nodes to start their send timer.

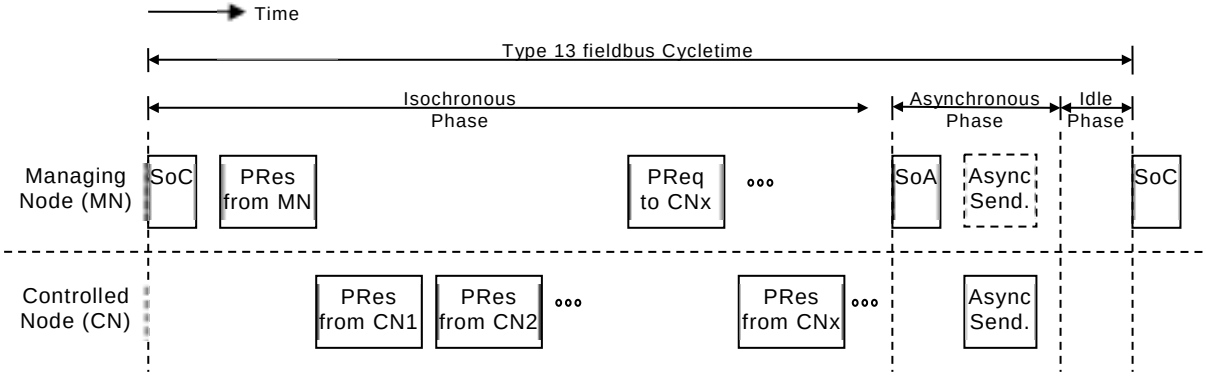


Figure 14 – Time triggered PRes example

The ability of a node to perform control of time triggered sending of PRes shall be indicated by the device description item P(D_DLL_FeaturePResTimeTriggered_BOOL).

The time when to send the PRes is controlled by V(PRES_TT_TIMEOUT). V(PRES_TT_TIMEOUT) defines the time between the end of the reception of the PRes of the MN and the start of sending the PRes.

So after reception of the PRes from the MN the timer T(PRES_TT_TIME) is started and when it expires the CN sends its PRes.

7.3.4 Primitive definitions

7.3.4.1 Primitive exchanged between ITC and DLS-user

Table 37 summarizes all primitives exchanged between the ITC and the DLS-user.

Table 37 – Primitives exchanged between ITC and DLS-user

Primitive name	Source	Associated parameters	Description
DL-PDO.req	DLS-user	D_addr DLSDU	Transmits (publish) a PDO element from the DLS-user. The PDO element is buffered in a D_addr dependent local buffer, until requested via the CSM-TPDO primitive from the CSM. MN: A buffer for every active node is available. CN: Only one buffer is available. D_addr is set to 255 (Broadcast)
DL-PDO.ind	ITC	S_addr D_addr DLSDU	Carries forward a received PDO element with its associated parameters to the DLS-user. Neither buffering nor processing is done with these elements

The parameters used with the primitives exchanged between the ITC and the DLS-user are described in Table 38.

Table 38 – Parameters used with primitives exchanged between ITC and DLS-user

Parameter name	Description
S_addr	The S_addr parameter specifies the DL-address of the publisher
D_addr	The D_addr parameter specifies the DL-address of the subscriber
DLSDU	This parameter specifies the information that is transferred by buffer transfer from the local DLE as a publisher to the remote multi-peer DLEs as subscribers

7.3.4.2 Primitive exchanged between ITC and CSM

The primitive and their associated parameters between the ITC and the CSM are described in 7.2.2.1.

7.3.5 ITC state table

Figure 15 depicts the state transition diagram of the ITC, and the state table of the ITC is shown in Table 39.

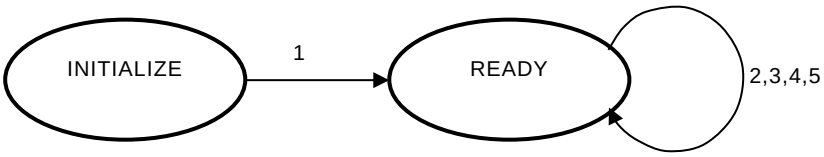


Figure 15 – State transition diagram of ITC

Table 39 – Transitions of ITC

#	Current state	Event /condition ⇒actions	Next state
1	INITIALIZE	POWER-ON or RESET / =>	READY
2	READY	CSM-RPDO.ind { PDODU } / => DL-PDO.ind { PDODU }	READY
3	READY	DL-PDO.req {addr := D_addr, PDODU } / ((CN = “TRUE” && addr = C_ADDR_BROADCAST) || MN = “TRUE”) => BUFFER_PDODATA (addr , PDODU) -- DL-PDO.cnf { Status:="OK" }	READY
4	READY	DL-PDO.req {addr := D_addr, PDODU } / (CN = “TRUE” && addr <> C_ADDR_BROADCAST) => -- DL-PDO.cnf { Status:="KO" }	READY

5	READY	IF MN = “TRUE” THEN
		addr := SLOT_CTR ( )
		ELSE
		addr := C_ADDR_BROADCAST
		ENDIF
		/
		=>
		PDODU := DEBUFFER_PDODATA (addr)
		CSM-TPDO.ind { D_addr := addr, PDODU }
			READY

7.3.6 Functions of ITC

All the functions used by the ITC are summarized in Table 40.

Table 40 – ITC function table

Function name	Input	Output	Operation
BUFFER_PDOPDATA	Addr, PDODU	(none)	Buffer the input data into the PDODATA buffer(addr)
DEBUFFER_PDODATA	Addr	PDODU	Get the output data from the PDODATA buffer(addr)
SLOT_CTR	(none)	(addr)	Determines the next Node ID, which shall be served at the next isochronous slot

7.4 Asynchronous transmission TX/RX control (ATC)

7.4.1 Overview

The asynchronous transmission TX/RX control (ATC) is responsible for buffering and dispatching in time DLSDU received from the ASND frame between DLS-user and the CSM.

7.4.2 Primitive definitions

7.4.2.1 Primitive exchanged between ATC and DLS-user

Table 41 summarizes all primitives exchanged between the ATC and the DLS-user.

Table 41 – Primitives exchanged between ATC and DLS-user

Primitive name	Source	Associated parameters	Description
DL-SDO.req	DLS-user	SDO_C_addr SDO_S_addr SDO_Prio SDO_DLSDU	Transmit Client request to Server.
DL-SDO.ind	ATC	SDO_C_addr SDO_S_addr SDO_DLSDU	Receive Client request from Server
DL-SDO.res	DLS-user	SDO_C_addr SDO_S_addr SDO_Prio SDO_DLSDU	Transmit Server response to Client
DL-SDO.cnf	ATC	SDO_C_addr SDO_S_addr SDO_DLSDU	Receiver Server response
DL-UDT.req	DLS-user	UDT_S_addr UDT_D_addr UDT_Prio UDT_DLSDU	Transmit unspecified data
DL-UDT.ind	ATC	UDT_S_addr UDT_D_addr UDT_DLSDU	Receive unspecified data
DL-STA.req	DLS-user	STA_S_addr	Request status response from slave
DL-STA.ind	ATC	STA_S_addr	Receive master request and prepare status response
DL-STA.res	DLS-user	STA_DLSDU	Transmit status response to master
DL-STA.cnf	ATC	STA_S_addr STA_DLSDU	Receive status response from slave

Primitive name	Source	Associated parameters	Description
DL-IDE.req	DLS-user	IDE_S_addr	Request ident response from slave
DL-IDE.ind	ATC	IDE_S_addr	Receive master request and prepare ident response
DL-IDE.res	DLS-user	IDE_DLSDU	Transmit ident response to master
DL-IDE.cnf	ATC	IDE_S_addr IDE_DLSDU	Receive ident response from slave
DL-CMD.req	DLS-user	CMD_D_addr CMD_DLSDU	Transmit NMT command
DL-CMD.ind	ATC	CMD_DLSDU	Receive NMT command
DL-SYN.req	DLS-user	SYN_S_addr SYN_DLSDU	Request sync response from slave
DL-SYN.ind	ATC	SYN_S_addr SYN_DLSDU	Receive master request and prepare sync response
DL-SYN.res	DLS-user	SYN_DLSDU	Transmit sync response to master
DL-SYN.cnf	ATC	SYN_S_addr SYN_DLSDU	Receive sync response from slave

The parameters used with the primitives exchange between the ATC and the DLS-user are described in Table 42.

Table 42 – Parameters used with primitives exchanged between ATC and DLS-user

Parameter name	Description
SDO_C_addr	The client-address parameter specifies the DL-address of the client DLE
SDO_S_addr	The server-address parameter specifies the DL-address of the server DLE
SDO_Prio	This parameter specifies the priority of the frame in the asynchronous send queue
SDO_DSLDU	This parameter specifies the DLS-user data that shall be transferred by the DLE
UDT_S_addr	The source-address parameter specifies the DL address of the source DLE
UDT_D_addr	The destination-address parameter specifies the DL-address of the server DLE
UDT_Prio	This parameter specifies the priority of the frame in the asynchronous send queue
UDT_DLSDU	This parameter specifies the DLS-user data that shall be transferred by the DLE
STA_S_addr	The slave-address parameter specifies the DL-address of the slave DLE, which should respond with its status message
STA_DLSDU	This parameter specifies the DLS-user data (status message) that shall be transferred by the DLE NOTE Some of the status parameters are inserted / removed by the DLL because of duplicated appearance of this information in normal frames and in the status message.
IDE_S_addr	The slave-address parameter specifies the DL-address of the slave DLE, which should respond with its ident message
IDE_DLSDU	This parameter specifies the DLS-user data (ident message) that shall be transferred by the DLE NOTE Some of the ident parameters are inserted / removed by the DLL because of duplicated appearance of this information in normal frames and in the ident message.
CMD_D_addr	The destination-address parameter specifies the DL-address of the subscribed DLE. The value 255, used for broadcast message, indicates this message for all connected nodes
CMD_DLSDU	This parameter specifies the information that is transferred by buffer transfer from the local DLE as a publisher to the remote multi-peer DLEs as subscribers. The further differentiation between the different possible NMT commands and the interpretation of the included NMT command data is responsible for the DLS-user

Parameter name	Description
SYN_S_addr	The slave-address parameter specifies the DL-address of the slave DLE, which should respond with its sync message
SYN_DLSDU	This parameter specifies the DLS-user data (sync message) that shall be transferred by the DLE NOTE Some of the sync parameters are inserted / removed by the DLL because of duplicated appearance of this information in normal frames and in the sync message.

7.4.2.2 Primitive exchanged between ATC and ES

Table 43 summarizes all primitives exchanged between the ATC and the ES.

Table 43 – Primitives exchanged between ATC and ES

Primitive name	Source	Associated parameters	Description
ES-STA.req	ES	Destination	Requests a Status Data transfer from the specified node. This primitive is only available at the MN

The parameters used with the primitives exchange between the ATC and the ES are described in Table 44.

Table 44 – Parameters used with primitives exchanged between ATC and ES

Parameter name	Description
Destination	This parameter indicates the Node ID of the transmitting node

7.4.2.3 Primitive exchanged between ATC and CSM

The primitive and their associated parameters between the ATC and the CSM are described in 7.2.2.2.

7.4.3 ATC state table

Figure 16 depicts the state transition diagram of the ATC, and the state table of the ATC is shown in Table 45.

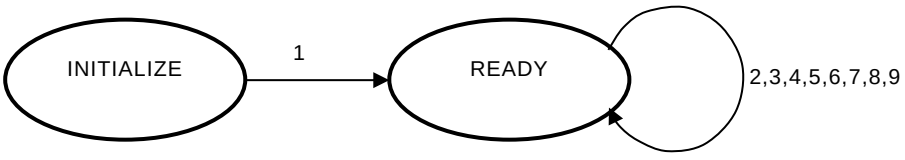


Figure 16 – State transition diagram of ATC

Table 45 – Transitions of ATC

#	Current state	Event /condition ⇒actions	Next state
1	INITIALIZE	POWER-ON or RESET / =>	READY

2	READY	CSM-RASND.ind { ASNDDU }
		/
		=>
		IF ASNDDU.SERVICEID = IDENT_RESPONSE THEN
		DL-IDE.cnf { ASNDDU.payload }
		ELSE IF ASNDDU.SERVICEID = STATUS_RESPONSE
		THEN DL-STA.cnf { ANSDDU.payload }
		ELSE IF ASNDDU.SERVICEID = NMT_REQUEST THEN
		DL-IDE.req { ASNDDU.payload }
		ELSE IF ASNDDU.SERVICEID = NMT_COMMAND THEN
		DL-CMD.ind { ASNDDU.payload }
		ELSE IF ASNDDU.SERVICEID = SDO THEN
		DL-SDO.ind { ASNDDU.payload }
		DL-SDO.cnf { ASNDDU.payload }
		ELSE IF ASNDDU.SERVICEID = SYNC_RESPONSE
		THEN
		DL-SYN.cnf { ASNDDU.payload }
		ELSE
		DL-UDT.ind { ASNDDU.payload }
		ENDIF
			READY

3	READY	DL-SDO.req { prio,SDODU }
		DL-SDO.res { prio,SDODU }
		/
		=>
		IF CHECK_ASNDDATAQ (prio) <> “Full” THEN
		QUEUE_ASNDDATA { prio,SDODU }
		-- DL-SDO.cnf { Status:=”OK” }
		ELSE
		-- DL-SDO.cnf { Status:="KO" }
		ENDIF
			READY

4	READY	DL-UDT.req { prio, IPDU }
		/
		=>
		IF CHECK_ASNDDATAQ (prio) <> “Full” THEN
		QUEUE_ASNDDATA { prio,IPDU }
		-- DL-UDT.cnf { Status:=”OK” }
		ELSE
		--DL-UDT.cnf { Status:="KO" }
		ENDIF
			READY

5	READY	DL-STA.req { STADU }
		ES-STA.res { STADU }
		/
		=>
		IF MN = “TRUE” THEN
		IF CHECK_ASNDDATAQ (8) <> “Full” THEN
		QUEUE_ASNDDATA { 8,STADU }
		-- DL-STA.cnf { Status:=”OK” }
		ELSE
		--DL-STA.cnf { Status:="KO" }
		ENDIF
		ELSE
		IF CHECK_ASNDDATAQ (11) <> “Full” THEN
		NMTDU := BUILD_NMTREQUEST
		(STATUS_REQUEST,src)
		QUEUE_ASNDDATA { 11,NMTDU }
		-- DL-IDE.cnf { Status:=”OK” }
		ELSE
		-- DL-IDE.cnf { Status:=”KO” }
		ENDIF
		ENDIF
			READY

6	READY	DL-IDE.req { IDEDU } / =>	READY

#	Current state	Event /condition ⇒actions	Next state

		IF MN = “TRUE” THEN
		IF CHECK_ASNDDATAQ (9) <> “Full” THEN
		QUEUE_ASNDDATA { 9,STADU }
		-- DL-IDE.cnf { Status:=”OK” }
		ELSE
		--DL-IDE.cnf { Status:="KO" }
		ENDIF
		ELSE
		IF CHECK_ASNDDATAQ (11) <> “FULL” THEN
		NMTDU := BUILD_NMTREQUEST
		(IDENT_REQUEST,src)
		QUEUE_ASNDDATA { 11,NMTDU }
		-- DL-IDE.cnf { Status:=”OK” }
		ELSE
		-- DL-IDE.cnf { Status:=”KO” }
		ENDIF
		ENDIF	

7	READY	DL-CMD.req { NMTDU }
		/
		=>
		IF CHECK_ASNDDATAQ (11) <> “Full” THEN
		QUEUE_ASNDDATA { 11,NMTDU }
		-- DL-CMD.cnf { Status:=”OK” }
		ELSE
		-- DL-CMD.cnf { Status:=”KO” }
		ENDIF
			READY

8	READY	CSM-TASND.ind { dest,ServiceID,ASNDDU }
		/
		=>
		IF ServiceID = NO_SERVICE THEN
		ASNDDU := BUILD_ASND (NO_SERVICE)
		ELSE IF ServiceID = IDENT_REQUEST THEN
		dummy := DEQUEUE_ASNDDATA(9)
		DL-IDE.ind { }
		DL-IDE.res { ASNDDU }
		ELSE IF ServiceID = STATUS_REQUEST THEN
		dummy := DEQUEUE_ASNDDATA(8)
		DL-STA.ind { }
		DL-STA.res { ASNDDU }
		ELSE IF ServiceID = SYNC_REQUEST THEN
		dummy := DEQUEUE_ASNDDATA(10)
		DL-SYN.ind { }
		DL-SYN.res { ASNDDU }
		ELSE IF ServiceID = NMT_REQUEST_INVITE THEN
		ASNDDU := DEQUEUE_ASNDDATA(11)
		ELSE
		prio := FIND_HIGHESTPRIO ( )
		ASNDDU := DEQUEUE_ASNDDATA(prio)
		ENDIF
		CSM_TASND.ind { ASNDDU }
			READY

9	READY	ATC-Prio.req { } / => Priority := FIND_HIGHESTPRIO ( ) RequestToSend := FIND_HIGHESTPRIO_RS ( ) ATC-Prio.ind {Priority , RequestToSend }	READY

10	READY	DL-SYN.req { SYNDU }
		/
		=>
		IF MN = “TRUE” THEN
		IF CHECK_ASNDDATAQ (10) <> “Full” THEN
		QUEUE_ASNDDATA {10,SYNDU }
		-- DL-SYN.cnf { Status:=”OK” }
		ELSE
		--DL-SYN.cnf { Status:="KO" }
		ENDIF
		ENDIF
			READY

7.4.4 Functions of ATC

All the functions used by the ITC are summarized in Table 46.

Table 46 – ATC function table

Function name	Input	Output	Operation
BUILD_ASND	ServiceID	ASNDDU	ASNDDU is assembled with the requested ServiceID
BUILD_NMTREQUEST	ServiceID ServiceTarget	NMTDU	NMTDU is assembled with the requested ServiceID and the requested ServiceTarget
CHECK_ASNDDATAQ	Addr	Status	Check the status of the queue(addr) for PDOPDATA. The returned status is any one of "Full", "Empty" and "Queued"
DEQUEUE_ASNDDATA	Addr	ASNDDU	Dequeue from the PDODATA queue(addr) on a FIFO basis
FIND_HIGHESTPRIO	(none)	PR	Returns the highest priority of the queue, that is not empty
FIND_HIGHESTPRIO_RS	(none)	RS	Returns the number of entries of the queue, that is not empty
QUEUE_ASNDDATA	Addr, ASNDDU	(none)	Queues the input data into the PDODATA Queue(addr) on a FIFO basis

7.5 Asynchronous slot scheduler (ASS)

7.5.1 Overview

The ASS schedules the asynchronous slot. It collects the request from all CNs and the MN, queues them inside an internal priority queue and decides in a fair way, which node gets the right to transmit data inside the next asynchronous slot.

This procedure is only available on the MN.

7.5.2 Primitive definitions

7.5.2.1 Primitive exchanged between ASS and CSM

The primitive and their associated parameters between the ASS and the CSM are described in 7.2.2.3.

7.5.3 ASS state table

Figure 17 depicts the state transition diagram of the ASS, and the state table of the ASS is shown in Table 47.

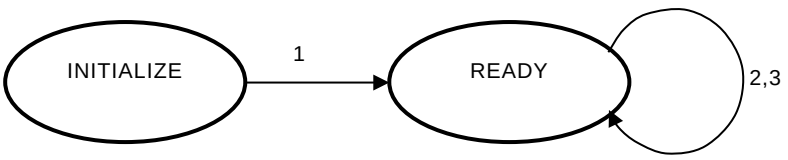


Figure 17 – State transition diagram of ASS

Table 47 – Transitions of ASS

#	Current state	Event /condition ⇒actions	Next state
1	INITIALIZE	POWER-ON or RESET / =>	READY
2	READY	CSM-Prio.ind { Source, Priority, RequestToSend } /CHECK_SOADATAQ(Source,Priority) <> “Full” && (MN = “TRUE” && Priority <= 9) || (CN = “TRUE” && Priority <= 7) => QUEUE_SOADATA(Source,Priority,RequestToSend)	READY

3	READY	CSM-SoA.req { }
		/
		=>
		Source := SCHEDULE_SOURCE ( )
		Priority := SCHEDULE_PRIORITY ( )
		dummy := DEQUEUE_SOADATA(Source,Priority)
		IF Priority < 7 THEN
		RequestedServiceID := UNSPECIFIED_INVITE
		ELSE IF Priority = 7 THEN
		RequestedServiceID := NMT_REQUEST_INVITE
		ELSE IF Priority = 8 THEN
		RequestedServiceID := STATUS_REQUEST
		ELSE IF Priority = 9 THEN
		RequestedServiceID := IDENT_REQUEST
		ELSE IF Priority = 10 THEN
		RequestedServiceID := SYNC_REQUEST
		ELSE
		RequestedServiceID := NO_SERVICE
		ENDIF
		RequestedServiceTarget := Source
		CSM-SoA.ind { RequestedServiceID,
		RequestedServiceTarget }
			READY

7.5.4 Functions of ASS

All the functions used by the ASS are summarized in Table 48.

Table 48 – ASS function table

Function name	Input	Output	Operation
QUEUE_SOADATA	Addr1 Addr2 RequestToSend	(none)	Queues the input data into the SOADATA Queue(addr1,addr2) on a FIFO basis
DEQUEUE_SOADATA	Addr1 Addr2	Request ToSend	Dequeue from the SOADATA queue(addr1,addr2) on a FIFO basis
SCHEDULE_SOURCE	(none)	Source	Returns the address of the node, which gets the right to transmit on the next asynchronous slot
SCHEDULE_PRIORITY	(none)	Priority	Returns the priority of the node, which gets the right to transmit on the next asynchronous slot

7.6 Exception signaling (ES)

7.6.1 Overview

The exception signaling handles the initialization of the exception signaling system from the MN to the CNs and the signaling of new exceptions from the CNs to the MN.

7.6.2 Primitive definitions

7.6.2.1 Primitive exchanged between ES and DLS-user

Table 49 summarizes all primitives exchanged between the ES and the DLS-user.

Table 49 – Primitives exchanged between ES and DLS-user

Primitive name	Source	Associated parameters	Description
DL-IERR.req	To DLE	D_addr	Request exception signaling Initialization. Only allowed for MN
DL-IERR.cnf	From DLE	D_addr	Confirmation of exception signaling Initialization
DL-ERR.req	To DLE		Request exception signaling. Only allowed for CN
DL-ERR.ind	From DLE	S_addr	Exception signaling indication
DL-ERR.cnf	From DLE		Confirmation of exception signaling

The parameters used with the primitives exchanged between the ES and the DLS-user are described in Table 50.

Table 50 – Parameters used with primitives exchanged between ES and DLS-user

Parameter name	Description
D_addr	The destination-address parameter specifies the DL-address of the subscribed DLE. The global address (255) for broadcast and the MN address (240) is not permitted
S_addr	The source-address parameter specifies the DL-address of the requested DLE. The global address (255) for broadcast and the MN address (240) is not permitted

7.6.2.2 Primitive exchanged between ES and ATC

The primitive and their associated parameters between the ES and the CSM are described in 7.4.2.2.

7.6.2.3 Primitive exchanged between ES and CSM

The primitive and their associated parameters between the ES and the CSM are described in 7.2.2.4.

7.6.3 ES state table

Figure 18 depicts the state transition diagram of the ES, and the state table of the ES is shown in Table 51.

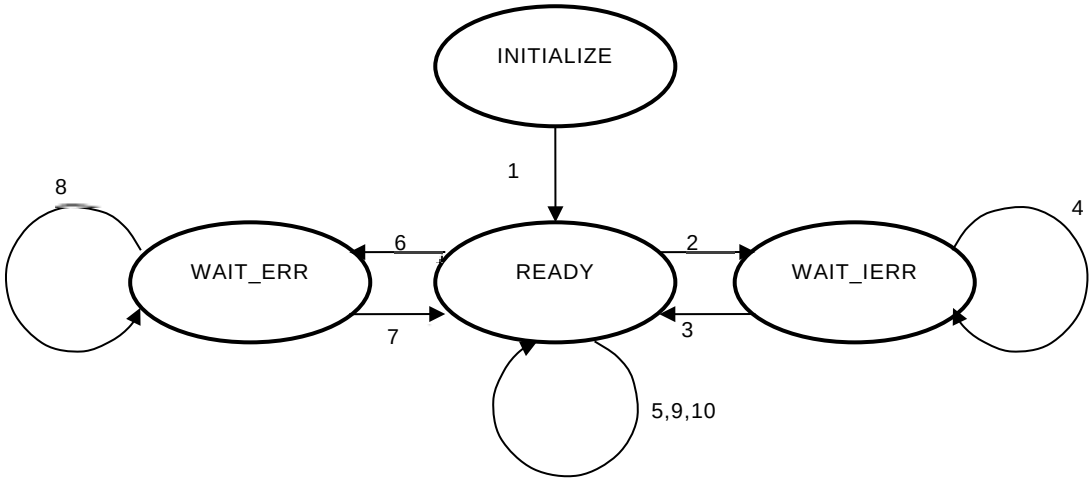


Figure 18 – State transition diagram of ES

Table 51 – Transitions of ES

#	Current state	Event /condition ⇒actions	Next state
1	INITIALIZE	POWER-ON or RESET / =>	READY
2	READY	DL-IERR.req { source } / MN = „TRUE“ => ER := “TRUE” CSM-TERR.ind { source , ER } ES-STA.req { source }	WAIT_IERR
3	WAIT_IERR	CSM-RERR.ind {source , EC } / EC = “TRUE” && MN = “TRUE” => ER := “FALSE” CSM-TERR.ind {source , EC } DL-IERR.cnf { source } ExceptionFlag(source) = “FALSE”	READY
4	WAIT_IERR	CSM-RERR.ind {source, EC } / EC = “FALSE” && MN = “TRUE” =>	WAIT_IERR
5	READY	CSM-RERR.ind { source , ER } / CN = “TRUE” => ExceptionFlag = “FALSE” EC := ER CSM-TERR.req { MN , EC }	READY
6	READY	DL-ERR.req { } / CN = “TRUE” => ExceptionFlag := !ExceptionFlag EN := ExceptionFlag CSM-TERR.ind { source , EN }	WAIT_ERR
7	WAIT_ERR	CSM-RERR.ind {source , EA } / CN = “TRUE” && EA = ExceptionFlag => DL-ERR.cnf { }	READY
8	WAIT_ERR	CSM-RERR.ind {source, EA } / CN = “TRUE” && EA <> ExceptionFlag	WAIT_ERR

#	Current state	Event /condition ⇒actions	Next state
		=>	
9	READY	CSM-RERR.ind {source, EN} / MN = “TRUE” && EN <> ExceptionFlag(source) => ExceptionFlag := EN CSM-TERR.ind { source , EN } ES-STA.req { source } DL-ERR.ind { source }	READY
10	READY	CSM-RERR.ind {source, EN} / MN = “TRUE” && EN = ExceptionFlag(source) =>	READY

7.6.4 Functions of ES

No functions are used.

7.7 NMT signaling (NS)

7.7.1 Overview

NS is used to transmit the NMT status from the own DLS-user to the CSM and to all nodes and to receive the NMT status from all nodes.

7.7.2 Primitive definitions

7.7.2.1 Primitive exchanged between NS and DLS-user

Table 52 summarizes all primitives exchanged between the NS and the DLS-user.

Table 52 – Primitives exchanged between NS and DLS-user

Primitive name	Source	Associated parameters	Description
DL-NMT.req	To DLE	S_addr	Transmit NMT status
DL-NMT.ind	From DLE	S_addr NMTStatus	Receive NMT status

The parameters used with the primitives exchanged between the NS and the DLS-user are described in Table 53.

Table 53 – Parameters used with primitives exchanged between NS and DLS-user

Parameter name	Description
S_addr	The source-address parameter specifies the DL-address of the published DLE
NMTStatus	This parameter indicates the current NMTStatus of the corresponding node

7.7.2.2 Primitive exchanged between NS and CSM

The primitive and their associated parameters between the NS and the CSM are described in 7.2.2.5.

7.7.3 NS state table

Figure 19 depicts the state transition diagram of the NS, and the state table of the NS is shown in Table 54.

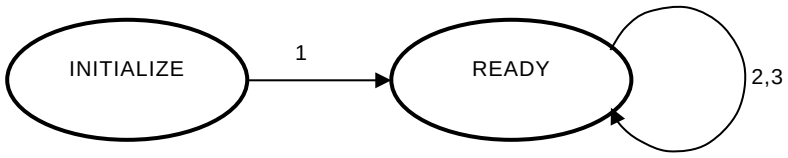


Figure 19 – State transition diagram of NS

Table 54 – Transitions of NS

#	Current state	Event /condition ⇒actions	Next state
1	INITIALIZE	POWER-ON or RESET / =>	READY
2	READY	DL-NMT.req { NMTStatus } / => CSM-TNMT.ind { NMTStatus }	READY
3	READY	CSM-RNMT.ind { source, NMTStatus } / => DL-NMT.ind { source, NMTStatus }	READY

7.7.4 Functions of NS

No functions are used.

7.8 DLL management protocol

7.8.1 Overview

The interface protocol between the DLM and the DLMS-user is described in this subclause.

7.8.2 Primitive definitions

7.8.2.1 Primitive exchanged between DLM and DLS-user

Table 55 summarizes all primitives exchanged between the DLM and the DLS-user.

Table 55 – Primitives exchanged between DLM and DLS-user

Primitive name	Source	Associated parameters	Description
DLM-Reset.req	DLS-user	(none)	This request primitive causes DLMS to reset the DLE
DLM-Reset.cnf	DLM	(out Status)	This indicates the status of the reset
DLM-Set-value.req	DLS-user	(in Variable-name, Desired-value)	This service is used to assign new values to the variables of the DLE
DLM-Set-value confirm	DLM	(out Status)	The DLMS-user receives confirmation that the specified variable has been set to the new value
DLM-Get-value.req	DLS-user	(in Variable-name)	This service is used to read the value of a

Primitive name	Source	Associated parameters	Description
			DLE variable
DLM-Get-value.cnf	DLM	(out Status, Current-value)	This service returns the actual value of the specified variable
DLM-Event.ind	DLM	(out Event-identifier, Entry Type, Time Stamp, Additional-information)	This service is used to inform the DLMS-user about certain events or errors in the DLL
DLM-Frame.ind	DLM	(out DLM-frame-identifier MC PS time reltime)	This service is used to inform the DLMS-user about the current frame type in process

The parameters used with the primitives exchanged between the DLM and the DLS-user are described in Table 56.

Table 56 – Parameters used with primitives exchanged between DLM and DLS-user

Parameter name	Description
DLM_Status	This parameter allows the DLMS-user to determine whether the requested DLMS was provided successfully, or failed for the reason specified. The value conveyed in this parameter is as follows: “OK – successfully completed”; “Failure – terminated before completion”
Variable-name	This parameter specifies the variable within the DLE whose value shall be set or read
Desired-value	This parameter specifies the desired value for the selected variable
Status	This parameter allows the DLMS-user to determine whether the requested DLMS was provided successfully, or failed for the reason specified. The value conveyed in this parameter is as follows: “OK – success – the variable could be updated"; “Failure – the variable does not exist or could not assume the new value”; "Failure – invalid parameters in the request"
Additional-information	This optional parameter provides event-specific additional information
DLM-frame-identifier	This parameter specifies the primitive within the DLE whose occurrence is being announced. The possible values are defined in the corresponding part of IEC 61158-4-13
MC	This parameter is toggled by the MN when the final multiplexed cycle has ended
PS	This parameter is toggled by the MN every n-th cycle. This prescaled signal is useful for “slow” nodes, which cannot react every cycle
time	This parameter is distributed by the MN and indicates the starting time of the Type 13 fieldbus cycle
reltime	This parameter is distributed by the MN and indicates the relative time, which is incremented by the Type 13 fieldbus cycle time when a SoC DLPDU is generated. The unit of relative time is μs
DLM-event-identifier	This parameter specifies the primitive or composite event within the DLE whose occurrence is being announced. The possible values are defined in the corresponding part of IEC 61158-4-13.
EntryType	Mode and profile information about the occurred error
TimeStamp	Nettime from the Type 13 fieldbus cycle when the error/event was detected
AdditionalInformation	This field contains device profile or vendor specific additional error information

7.8.2.2 Primitive exchanged between DLM and CSM

The primitive and their associated parameters between the DLM and the CSM are described in clause7.2.2.6.

7.8.3 DLM state table

Figure 20 depicts the state transition diagram of the DLM, and the state table of the DLM is shown in Table 57.

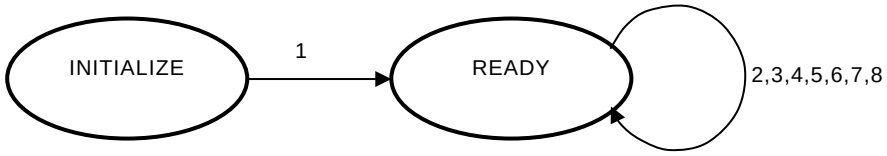


Figure 20 – State transition diagram of DLM

Table 57 – Transitions of DLM

#	Current state	Event /condition ⇒actions	Next state
1	INITIALIZE	POWER-ON or RESET / =>	READY
2	READY	DLM-Reset.req { } => RESET = “TRUE” Status := “Success” DLM-Reset.Cnf { Status }	READY
3	READY	DLM-Set-value.req {Variable-name, Desired-value } / CHECK_VALUE (Variable-name, Desired-value) = "Valid" => SET_VALUE (Variable-name, Desired-value) Status := "Success" DLM-Set-value.cnf {Status}	READY
4	RE ADY	DLM-Set-value.req {Variable-name, Desired-value } / CHECK_VALUE (Variable-name, Desired-value) <> "Valid" => Status := "Failure" DLM-Set-value.cnf {Status}	READY
5	READY	DLM-Get-value.req {Variable-name } / CHECK_VAR (Variable-name) = "Valid" => Current-value := GET_CURRENT_VAL (Variable-name) Status := "Success" DLM-Get-value.cnf { Current-value, Status }	READY
6	READY	DLM-Get-value.req {Variable-name } / CHECK_VAR (Variable-name) <> "Valid" => Current-value := NIL Status := "Failure" DLM-Get-value.cnf { Current-value, Status }	READY
7	READY	CSM-Frame.ind { DLM-frame-identifier,MS,PS,time,reltime} / => DLM-Frame.ind { DLM-frame-identifier,MS,PS,time,reltime}	READY

#	Current state	Event /condition ⇒actions	Next state
8	READY	CSM-Event.ind { DLM-Event-identifier, EntryType, TimeStamp, Additional-Information} / => DLM-Event.ind { DLM-Event-identifier, EntryType, TimeStamp, Additional-Information}	READY

7.8.4 Functions of DLM

All the functions used by the DLM are summarized in Table 58.

Table 58 – DLM function table

Function name	Description and operation
CHECK_VALUE (Variable-name, Desired-value)	Check if the requested variable with desired value is valid. Possible variables with the value are defined in 4.5.
CHECK_VAR (Variable-name)	Check if the requested variable is valid. Possible variables are defined in 4.5.
GET_CURRENT_VAL (Variable-name)	Get the value of requested variable
SET_VALUE	Set the value of requested variable

Bibliography

NOTE All parts of the IEC 61158 series, as well as IEC 61784-1 and IEC 61784-2 are maintained simultaneously. Cross-references to these documents within the text therefore refer to the editions as dated in this list of normative references.

IEC 61158-1:2014, *Industrial communication networks – Fieldbus specifications – Part 1: Overview and guidance for the IEC 61158 and IEC 61784 series*

IEC 61158-3-13:2014, *Industrial communication networks – Fieldbus specifications – Part 3-13: Data-link layer service definition – Type 13 elements*

IEC 61158-5-13:2014, *Industrial communication networks – Fieldbus specifications – Part 5-13: Application layer service definition – Type 13 elements*

IEC 61158-6-13:2014, *Industrial communication networks – Fieldbus specifications – Part 6-13: Application layer protocol specification – Type 13 elements*

IEC 61784-1, *Industrial communication networks – Profiles – Part 1: Fieldbus profiles*

IEC 61784-2, *Industrial communication networks – Profiles – Part 2: Additional fieldbus profiles for real-time networks based on ISO/IEC 8802-3*

EPSG DS 301 V1.2.0, *Ethernet POWERLINK Communication Profile Specification, Draft Standard Version 1.2.0, EPSG 2013*, available at http://www.ethernet-powerlink.org/

_____

SOMMAIRE

COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

_____

RÉSEAUX DE COMMUNICATION INDUSTRIELS – SPÉCIFICATIONS DES BUS DE TERRAIN –

Partie 4-13: Spécification du protocole de la couche liaison de données – Éléments de type 13

AVANT-PROPOS

1) La Commission Electrotechnique Internationale (CEI) est une organisation mondiale de normalisation composée de l'ensemble des comités électrotechniques nationaux (Comités nationaux de la CEI). La CEI a pour objet de favoriser la coopération internationale pour toutes les questions de normalisation dans les domaines de l'électricité et de l'électronique. A cet effet, la CEI – entre autres activités – publie des Normes internationales, des Spécifications techniques, des Rapports techniques, des Spécifications accessibles au public (PAS) et des Guides (ci-après dénommés "Publication(s) de la CEI"). Leur élaboration est confiée à des comités d'études, aux travaux desquels tout Comité national intéressé par le sujet traité peut participer. Les organisations internationales, gouvernementales et non gouvernementales, en liaison avec la CEI, participent également aux travaux. La CEI collabore étroitement avec l'Organisation Internationale de Normalisation (ISO), selon des conditions fixées par accord entre les deux organisations.
2) Les décisions ou accords officiels de la CEI concernant les questions techniques représentent, dans la mesure du possible, un accord international sur les sujets étudiés, étant donné que les Comités nationaux de la CEI intéressés sont représentés dans chaque comité d'études.
3) Les Publications de la CEI se présentent sous la forme de recommandations internationales et sont agréées comme telles par les Comités nationaux de la CEI. Tous les efforts raisonnables sont entrepris afin que la CEI s'assure de l'exactitude du contenu technique de ses publications; la CEI ne peut pas être tenue responsable de l'éventuelle mauvaise utilisation ou interprétation qui en est faite par un quelconque utilisateur final.
4) Dans le but d'encourager l'uniformité internationale, les Comités nationaux de la CEI s'engagent, dans toute la mesure possible, à appliquer de façon transparente les Publications de la CEI dans leurs publications nationales et régionales. Toutes divergences entre toutes Publications de la CEI et toutes publications nationales ou régionales correspondantes doivent être indiquées en termes clairs dans ces dernières.
5) La CEI elle-même ne fournit aucune attestation de conformité. Des organismes de certification indépendants fournissent des services d'évaluation de conformité et, dans certains secteurs, accèdent aux marques de conformité de la CEI. La CEI n'est responsable d'aucun des services effectués par les organismes de certification indépendants.
6) Tous les utilisateurs doivent s'assurer qu'ils sont en possession de la dernière édition de cette publication.
7) Aucune responsabilité ne doit être imputée à la CEI, à ses administrateurs, employés, auxiliaires ou mandataires, y compris ses experts particuliers et les membres de ses comités d'études et des Comités nationaux de la CEI, pour tout préjudice causé en cas de dommages corporels et matériels, ou de tout autre dommage de quelque nature que ce soit, directe ou indirecte, ou pour supporter les coûts (y compris les frais de justice) et les dépenses découlant de la publication ou de l'utilisation de cette Publication de la CEI ou de toute autre Publication de la CEI, ou au crédit qui lui est accordé.
8) L'attention est attirée sur les références normatives citées dans cette publication. L'utilisation de publications référencées est obligatoire pour une application correcte de la présente publication.
9) L'attention est attirée sur le fait que certains des éléments de la présente Publication de la CEI peuvent faire l'objet de droits de brevet. La CEI ne saurait être tenue pour responsable de ne pas avoir identifié de tels droits de brevets et de ne pas avoir signalé leur existence.

L'attention est attirée sur le fait que l'utilisation du type de protocole associé est restreinte par les détenteurs des droits de propriété intellectuelle. En tout état de cause, l'engagement de renonciation partielle aux droits de propriété intellectuelle pris par les détenteurs de ces droits autorise l'utilisation d'un type de protocole de couche avec les autres protocoles de couche du même type, ou dans des combinaisons avec d'autres types autorisées explicitement par les détenteurs des droits de propriété intellectuelle pour ce type.

NOTE Les combinaisons de types de protocoles sont spécifiées dans la CEI 61784-1 et la CEI 61784-2.

La Norme internationale CEI 61158-4-13 a été établie par le sous-comité 65C: Réseaux industriels, du comité d'études 65 de la CEI: Mesure, commande et automation dans les processus industriels.

Cette deuxième édition annule et remplace la première édition parue en 2007. Cette édition constitue une révision technique.

Les modifications majeures par rapport à l'édition précédente sont énumérées ci-dessous:

- ajout d'une nouvelle classe de communication,
- corrections et
- améliorations rédactionnelles.

Le texte de cette norme est issu des documents suivants:

FDIS	Rapport de vote
65C/762/FDIS	65C/772/RVD

Le rapport de vote indiqué dans le tableau ci-dessus donne toute information sur le vote ayant abouti à l'approbation de cette norme.

Cette publication a été rédigée selon les Directives ISO/CEI, Partie 2.

Une liste de toutes les parties de la série CEI 61158, publiées sous le titre général *Réseaux de communication industriels – Spécifications des bus de terrain*, peut être consultée sur le site web de la CEI.

Le comité a décidé que le contenu de cette publication ne sera pas modifié avant la date de stabilité indiquée sur le site web de la CEI sous "http://webstore.iec.ch" dans les données relatives à la publication recherchée. À cette date, la publication sera:

- reconduite;
- supprimée;
- remplacée par une édition révisée, ou
- amendée.

INTRODUCTION

La présente partie de la norme CEI 61158 s'inscrit dans une série créée pour faciliter l'interconnexion des composants de systèmes d'automatisation. Elle renvoie aux autres normes de l'ensemble défini par le modèle de référence de bus de terrain "à trois couches" décrit dans la CEI 61158-1.

Le protocole de liaison de données assure un service de liaison de données en s'appuyant sur les services offerts par la couche physique. La présente norme a pour principal objet de préciser un ensemble de règles de communication, exprimées sous la forme de procédures à réaliser par des entités de liaison de données homologues (DLE) au moment de la communication. Ces règles de communication visent à fournir une base saine pour le développement, dans divers buts:

a) en tant que guide pour les développeurs et les concepteurs;

b) dans une optique d'utilisation lors de l'essai et de l'achat de matériel;

c) dans le cadre d'un accord pour l'admission de systèmes dans l'environnement de systèmes ouverts;

d) en tant que précision apportée à la compréhension des communications en temps critique dans le modèle OSI.

La présente norme traite, en particulier, de la communication et de l'interfonctionnement des capteurs, effecteurs et autres appareils d'automatisation. L'utilisation conjointe de la présente norme avec d'autres normes entrant dans les modèles de référence OSI ou de bus de terrain permet à des systèmes autrement incompatibles de fonctionner ensemble dans n'importe quelle combinaison.

RÉSEAUX DE COMMUNICATION INDUSTRIELS – SPÉCIFICATIONS DES BUS DE TERRAIN –

Partie 4-13: Spécification du protocole de la couche liaison de données – Éléments de type 13

1 Domaine d'application

1.1 Généralités

La couche liaison de données assure les communications de messagerie de base prioritaire entre les appareils d'un environnement d'automatisation.

Ce protocole offre des opportunités de communication à toutes les entités de liaison de données participantes

a) de manière cyclique avec un démarrage synchrone, selon une programmation préétablie, et

b) de manière cyclique ou acyclique et asynchrone, comme demandé par chaque cycle de chacune de ces entités de liaison de données.

Par conséquent, ce protocole peut se caractériser comme assurant un accès cyclique et acyclique asynchrone, mais avec un redémarrage synchrone de chaque cycle.

1.2 Spécifications

La présente norme spécifie

a) des procédures pour le transfert dans les délais impartis de données et d'informations de commande d'une entité utilisateur de liaison de données vers une entité utilisateur homologue, et parmi les entités de liaison de données formant le fournisseur de services de liaison de données distribué;

b) des procédures pour donner des occasions de communications à toutes les entités DL participantes, de manière séquentielle et cyclique pour le transfert déterministe et synchronisé à des intervalles cycliques ne dépassant pas une milliseconde;

c) des procédures pour donner des occasions de communications disponibles pour la transmission de données prioritaires, conjointement avec une transmission de données non prioritaires n'ayant aucune influence sur la transmission de données prioritaires;

d) des procédures pour donner des occasions de communications cycliques et acycliques pour la transmission des données prioritaires avec accès prioritisé;

e) des procédures pour donner des occasions de communications basées sur la commande de l'accès au support de l'ISO/CEI 8802-3, avec indication des nœuds à ajouter ou à enlever lors d'un fonctionnement normal;

f) la structure des DLPDU de bus de terrain utilisées par le protocole de la présente norme pour le transfert des données et des informations de commande, et leur représentation sous forme d'unités de données d'interface physique.

1.3 Procédures

Les procédures sont définies en termes

a) d'interactions entre les entités DL (DLE) par l'échange de DLPDU de bus de terrain;

b) d'interactions entre un fournisseur de service DL (DLS) et un utilisateur de DLS au sein du même système par l'échange de primitives DLS;

c) d'interactions entre un fournisseur de DLS et un fournisseur de services Ph dans le même système par l'échange de primitives de services Ph.

1.4 Applicabilité

Ces procédures s'appliquent aux instances de communication entre des systèmes qui prennent en charge des services de communications prioritaires dans la couche liaison de données des modèles de référence OSI ou de bus de terrain, et qui peuvent être connectés dans un environnement d'interconnexion de systèmes ouverts.

Les profils sont un moyen simple à plusieurs attributs de récapituler les capacités d'une mise en œuvre, et donc son applicabilité en fonction des différents besoins de communications prioritaires.

1.5 Conformité

La présente norme spécifie également les exigences de conformité relatives aux systèmes mettant en œuvre ces procédures. La présente norme ne comporte aucun essai visant à démontrer la conformité à ces exigences.

2 Références normatives

Les documents suivants sont cités en référence de manière normative, en intégralité ou en partie, dans le présent document et sont indispensables pour son application. Pour les références datées, seule l'édition citée s'applique. Pour les références non datées, la dernière édition du document de référence s'applique (y compris les éventuels amendements).

IEC 61588, *Precision clock synchronization protocol for networked measurement and control systems* (disponible en anglais seulement)

ISO/CEI 7498-1, *Technologies de l'information – Interconnexion de systèmes ouverts (OSI) – Modèle de référence de base: Le modèle de base*

ISO/CEI 7498-3, *Technologies de l'information – Interconnexion de systèmes ouverts (OSI) – Modèle de référence de base: Dénomination et adressage*

ISO/IEC 8802-3:2000, *Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications* (disponible en anglais seulement)

ISO/CEI 10731, *Technologies de l'information – Interconnexion de systèmes ouverts – Modèle de référence de base – Conventions pour la définition des services OSI*

3 Termes, définitions, symboles, abréviations et conventions

Pour les besoins du présent document, les termes, définitions, symboles, abréviations et conventions suivants s'appliquent.

3.1 Termes et définitions relatifs au modèle de référence

La présente norme est en partie issue des concepts élaborés dans les normes ISO/CEI 7498-1 et ISO/CEI 7498-3 et utilise les termes suivants:

3.1.1	**adresse de DL**	[7498-3]
3.1.2	**mapping d'adresse de DL**	[7498-1]
3.1.3	**adresse de DL appelée**	[7498-3]
3.1.4	**adresse de DL appelante**	[7498-3]
3.1.5	**connexion centralisée à plusieurs extrémités**	[7498-1]
3.1.6	**connexion de DL**	[7498-1]
3.1.7	**extrémité de connexion de DL**	[7498-1]
3.1.8	**identifiant d'extrémité de connexion de DL**	[7498-1]
3.1.9	**transmission en mode connexion de DL**	[7498-1]
3.1.10	**transmission en mode sans connexion de DL**	[7498-1]
3.1.11	**entités (N) correspondantes** **entités de DL correspondantes (N=2)** **entités de Ph correspondantes (N=1)**	[7498-1]
3.1.12	**transmission duplex de DL**	[7498-1]
3.1.13	**entité (N)** **entité de DL (N=2)** **entité de Ph (N=1)**	[7498-1]
3.1.14	**fonctionnalité de DL**	[7498-1]
3.1.15	**contrôle de flux**	[7498-1]
3.1.16	**couche (N)** **couche DL (N=2)** **couche Ph (N=1)**	[7498-1]
3.1.17	**gestion de couche**	[7498-1]
3.1.18	**vue locale de DL**	[7498-3]
3.1.19	**nom de DL**	[7498-3]
3.1.20	**domaine (d'adressage) de dénomination**	[7498-3]
3.1.21	**entités homologues**	[7498-1]
3.1.22	**nom primitif**	[7498-3]
3.1.23	**protocole de DL**	[7498-1]
3.1.24	**identifiant de connexion de protocole de DL**	[7498-1]
3.1.25	**unité de données de protocole de DL**	[7498-1]
3.1.26	**relais de DL**	[7498-1]
3.1.27	**réinitialisation**	[7498-1]
3.1.28	**adresse de DL en réponse**	[7498-3]
3.1.29	**acheminement**	[7498-1]

3.1.30 segmentation [7498-1]

3.1.31 service (N) [7498-1]
service de DL (N=2)
service de Ph (N=1)

3.1.32 point d'accès au service (N) [7498-1]
point d'accès au service de DL (N=2)
point d'accès au service de Ph (N=1)

3.1.33 adresse de point d'accès au service de DL [7498-3]

3.1.34 identifiant de connexion de service de DL [7498-1]

3.1.35 unité de données de service de DL [7498-1]

3.1.36 transmission simplex de DL [7498-1]

3.1.37 sous-système de DL [7498-1]

3.1.38 gestion des systèmes [7498-1]

3.1.39 données de l'utilisateur de DLS [7498-1]

3.2 Termes et définitions relatifs à la convention de service

Pour les besoins du présent document, les termes suivants, définis dans l'ISO/CEI 10731 et relatifs à la couche Liaison de données, s'appliquent:

3.2.1 acceptant

3.2.2 service asymétrique

3.2.3 (primitive) "confirm";
(primitive) requestor.deliver

3.2.4 (primitive) "deliver"

3.2.5 fonctionnalité confirmée de DL

3.2.6 fonctionnalité de DL

3.2.7 vue locale de DL

3.2.8 fonctionnalité obligatoire de DL

3.2.9 fonctionnalité non confirmée de DL

3.2.10 fonctionnalité de DL lancée par le fournisseur

3.2.11 fonctionnalité facultative de fournisseur de DL

3.2.12 primitive de service de DL;
primitive

3.2.13 fournisseur de service de DL

3.2.14 utilisateur de service de DL

3.2.15 fonctionnalité facultative d'utilisateur de DLS

3.2.16 (primitive) "indication";
(primitive) acceptor.deliver

3.2.17 multihomologue

3.2.18 (primitive) “request”;
(primitive) requestor.submit

3.2.19 demandeur

3.2.20 (primitive) “response”;
(primitive) acceptor.submit

3.2.21 (primitive) “submit”

3.2.22 service symétrique

3.3 Termes et définitions pour les services de liaison de données

Pour les besoins du présent document, les termes et définitions suivants s'appliquent.

3.3.1
async-only CN (nœud commandé "async-only")
nœud commandé qui n'est accessible que par échange sur interrogation (polling)

3.3.2
période asynchrone
deuxième partie du cycle de Type 13, commençant par une trame "start of asynchronous" (début de l'asynchrone) (SoA)

3.3.3
mode Ethernet de base
mode qui fournit une communication Ethernet traditionnel

3.3.4
“continuous-time-triggered” (cadencé en continu)
classe de communication où une communication isochrone a lieu à chaque cycle

Note 1 à l'article: Les données envoyées d'un MN vers divers CN sont condensées en une PollResponse. Aucune demande PollRequest à ces CN n'est nécessaire. Les CN envoient leur PollResponse à déclenchement temporel. (Une alternative à continu).

Note 2 à l'article: Il existe trois classes de nœuds: *“continuous” (continu), “multiplexed” (multiplexé)* et *“continuous-time-triggered” (cadencé en continu).* Chaque nœud est un membre d'une seule de ces classes.

3.3.5
“continuous” (continu)
classe de communication où une communication isochrone a lieu à chaque cycle (opposée à la classe "multiplexed" (multiplexé))

Note 1 à l'article: Il existe trois classes de nœuds: *continuous (continu), multiplexed (multiplexé)* et *“continuous-time-triggered” (cadencé en continu).* Chaque nœud est un membre d'une seule de ces classes.

3.3.6
nœud commandé (CN)
nœud du réseau, sans la capacité de gérer le mécanisme SCNM

3.3.7
diagramme d’états de cycle
diagramme d’états qui contrôle le cycle de couche liaison de données et qui est lui-même commandé par le diagramme d’états de NMT qui détermine le mode de fonctionnement actuel

3.3.8
durée de cycle
temps entre deux trames "start of cyclic" (début du cyclique) (SoC) consécutives

Note 1 à l'article: La durée de cycle comprend le temps de transmission des données et un certain temps d'inactivité avant le début du cycle prochain.

3.3.9
adresse de DLCEP
adresse de DL qui désigne soit

a) une extrémité de connexion de DL d'homologue, soit

b) une extrémité de connexion de DL d'éditeur à homologues multiples, et implicitement l'ensemble correspondant d'extrémités de connexion de DL d'abonnés, où chaque extrémité de connexion de DL existe au sein d'un DLSAP distinct et est associée à une adresse de DLSAP distincte correspondante

3.3.10
segment de DL,
liaison,
liaison locale
sous-réseau de DL unique dans lequel toutes les éventuelles DLE connectées peuvent communiquer directement, sans intervention de relayage de DL, chaque fois que toutes celles des DLE qui participent à une instance de communication sont simultanément attentives au sous-réseau de DL pendant la/les période(s) de communication tentée(s)

3.3.11
DLSAP
point distinctif où des services de DL sont fournis par une entité de DL unique à une unique entité de couche supérieure

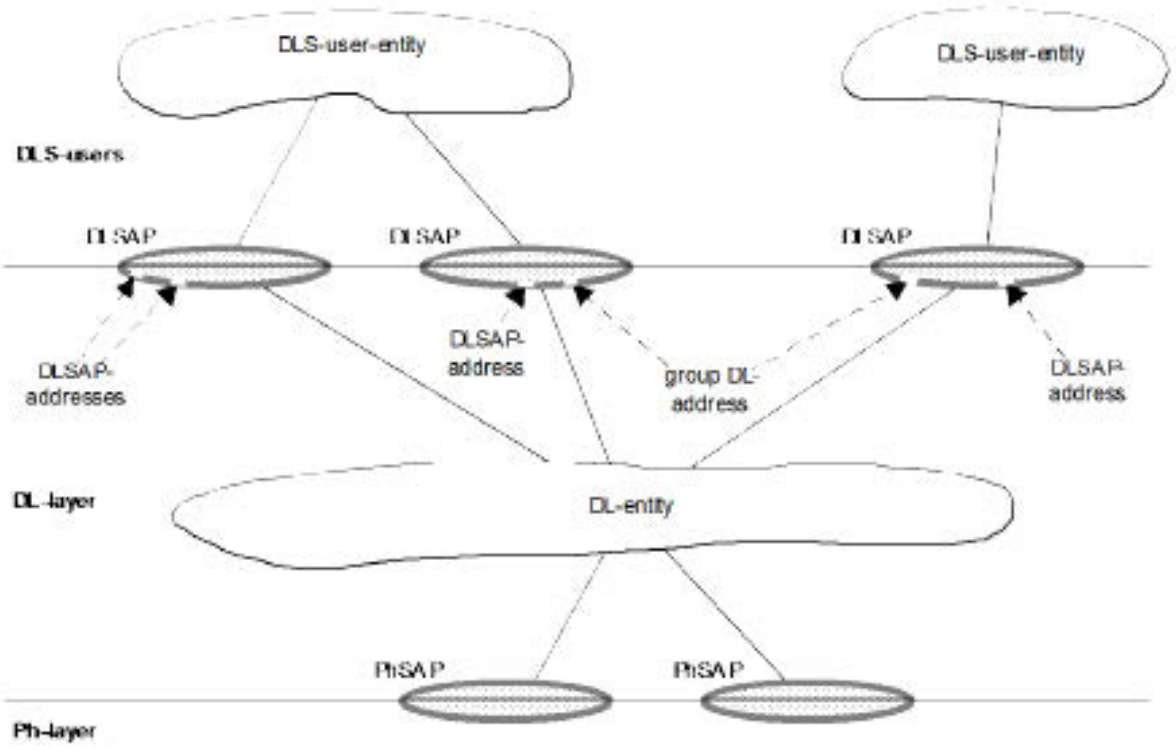
Note 1 à l'article: Cette définition, dérivée de l'ISO/CEI 7498-1, est reprise ici pour faciliter la compréhension de la distinction critique entre les DLSAP et leurs adresses de DL.

3.3.12
adresse de DL(SAP)
soit une adresse de DLSAP individuelle, désignant un unique DLSAP d'un unique utilisateur de DLS, soit une adresse de DL de groupe désignant potentiellement plusieurs DLSAP, chacun d'un unique utilisateur de DLS

Note 1 à l'article: Cette terminologie est choisie parce que l'ISO/CEI 7498-3 ne permet pas l'utilisation du terme "adresse de DLSAP" pour désigner plus d'un seul DLSAP au niveau d'un utilisateur de DLS unique.

3.3.13
adresse de DLSAP (individuelle)
adresse de DL qui désigne un seul DLSAP au sein de la liaison étendue

Note 1 à l'article: Une entité de DL unique peut avoir plusieurs adresses de DLSAP associées à un seul DLSAP (voir Figure 1).



NOTE 1 Les DLSAP et les PhSAP sont illustrés sous la forme d'ovales traversant la frontière entre deux couches adjacentes.

NOTE 2 Les adresses de DL sont illustrées comme désignant de petits espaces (points d'accès) dans la partie DLL d'un DLSAP.

NOTE 3 Une entité de DL unique peut avoir plusieurs adresses de DLSAP et adresses de DL de groupe associées à un même DLSAP.

Légende

Anglais	Français
DLS-use-entity	Entité d'utilisateur de DLS
DLS-users	Utilisateurs de DLS
DLSAP-addresses	Adresses de DLSAP
DLSAP-address	Adresse de DLSAP
Group DL address	Adresse de DL de groupe
DL-entity	Entité de DL
DL-layer	Couche de DL
Ph-layer	Couche Physique
PhSAP	PhSAP

Figure 1 – Relation entre les DLSAP, les adresses des DLSAP et les adresses de DL de groupe

3.3.14
cycle de bus de terrain de Type 13
intervalle de temps fixe répété de façon consécutive organisé par le MN

3.3.15
ID des nœuds de bus de terrain de Type 13
numéro unique au sein d'un réseau de Type 13 utilisé pour adresser un nœud de bus de terrain de Type 13

3.3.16
trame
synonyme critiqué de DLPDU

3.3.17
données isochrones
données qui sont émises à chaque cycle (ou chaque énième cycle en cas de données isochrones multiplexées)

3.3.18
période isochrone
période au cours de chaque cycle qui offre un fonctionnement déterministe grâce à sa réservation pour l'échange de données isochrones (continues ou multiplexées)

3.3.19
Ethernet traditionnel
Ethernet tel que normalisé en ISO/CEI 8802-3 (fonctionnement non déterministe en environnements à temps non prioritaire)

3.3.20
nœud gérant
nœud qui peut gérer le mécanisme SCNM

3.3.21
multiplexed (multiplexé)
classe de communication où une communication cyclique a lieu de manière à ce que “m” nœuds soient servis en “s” cycles (une alternative à la classe "continuous" (continu))

3.3.22
intervalle de temps multiplexé
intervalle de temps affecté aux données isochrones multiplexées et qui est partagé entre plusieurs nœuds

3.3.23
connexion multipoint
connexion d'un nœud à plusieurs autres nœuds

Note 1 à l’article: Une connexion multipoint permet le transfert de données à partir d'un même éditeur vers plusieurs nœuds abonnés.

3.3.24
DLC à homologues multiples
connexion de DL centralisée à points d'extrémité multiples proposant une transmission duplex de DL entre un seul utilisateur de DLS distingué, appelé l'éditeur ou l'utilisateur de DLS d'édition, et un ensemble d'utilisateurs de DLS homologues mais non distingués, appelés collectivement les abonnés ou utilisateurs de DLS d'abonnement, dans laquelle l'utilisateur de DLS d'édition peut effectuer des envois vers les utilisateurs de DLS d'abonnement en tant que groupe (mais pas individuellement) et les utilisateurs de DLS d'abonnement peuvent effectuer des envois vers l'utilisateur de DLS d'édition (mais pas les uns vers les autres)

3.3.25
NetTime
temps d'horloge du MN tel que distribué par la trame SoC à tous les CN

3.3.26
gestion de réseau
fonctions et services de gestion qui accomplissent l'initialisation, la configuration et le traitement d'erreurs du réseau

3.3.27
nœud
entité de DL unique, telle qu'elle apparaît dans une liaison locale

3.3.28
PollRequest
trame qui est utilisée dans la partie isochrone d'un cycle de communications

3.3.29
PollResponse
trame qui est utilisée dans la partie isochrone d'un cycle de communications pour répondre à une trame "PollRequest"

3.3.30
objet de données de processus
objet permettant l'échange de données isochrones entre les nœuds

3.3.31
protocole
convention relative aux formats de données, aux séquences temporelles et à la correction d'erreurs lors de l'échange de données des systèmes de communications

3.3.32
utilisateur de DLS destinataire
utilisateur du service de DL auquel sont destinées les données de l'utilisateur de DLS

Note 1 à l'article: Un utilisateur de service de DL peut être simultanément un utilisateur de DLS expéditeur et destinataire.

3.3.33
utilisateur de DLS expéditeur
utilisateur du service de DL à la source des données de l'utilisateur de DLS

3.3.34
objet de données de service
objet permettant l'échange de données asynchrones entre les nœuds

3.3.35
gestion du réseau de communication par intervalle de temps
mécanisme qui assure l'absence de collisions durant l'accès au réseau physique de tous les nœuds du réseau, permettant ainsi une communication déterministe à travers l'Ethernet traditionnel

3.4 Symboles et abréviations

ASnd	"Asynchronous send" (Envoi de l'asynchrone) (type d'une trame de Type 13)
ASS	Asynchronous slot scheduler (Programmateur d'intervalle de temps asynchrone)
ATC	Asynchronous transmissionTX/RX control (Commande TX/RX en transmission asynchrone)
CN	Controlled Node (Nœud commandé)
CSM	Cycle State Machine (Diagramme d'états de cycle)
DL-	Préfixe relatif à la couche liaison de données
DLC	DL-connection (Connexion de DL)
DLCEP	DL-connection-end-point (Extrémité de connexion de DL)
DLE	DL-entity (Entité de DL (instance active locale de la couche liaison de données))
DLL	DL-Layer (Couche DL)
DLPDU	DL-protocol-data-unit (Unité de données de protocole de DL)

DLM	DL-management (Gestion de DL)
DLMS	DL-management service (Service de gestion de DL)
DLS	DL-service (Service de DL)
DLSAP	DL-service-access-point (Point d'accès au service de DL)
DLSDU	DL-service-data-unit (Unité de données de service de DL)
ES	Exception signaling (Signalisation d'exception)
FIFO	First-In First-Out (Premier entré, premier sorti), (méthode de mise en file d'attente))
ITC	Isochronous transmission RX/TX control (Commande RX/TX en transmission isochrone()
MAC	Media Access Control (Commande de l'accès au support)
MN	Managing Node (Nœud gérant)
NMT	Network management (Gestion de réseau)
NS	NMT Signaling (Signalisation de NMT)
OSI	Open Systems Interconnection (Interconnexion de systèmes ouverts)
PDO	Process Data Object (Objet de données de processus)
PDU	Protocol data unit (Unité de données de protocole)
Ph-	Préfixe relatif à la couche physique
PhL	Physical Layer (Couche physique)
PhPDU	Ph-protocol-data-unit (Unité de données de protocole de couche physique)
PhS	Ph-service (Service de couche physique)
PReq	PollRequest (type d'une trame de Type 13)
PRes	PollResponse (type d'une trame de Type 13)
SCNM	Slot Communication Network Management (Gestion du réseau de communication par intervalle de temps()
SDO	Service Data Object (Objet de données de service)
SoA	Start of asynchronous (Début de l'asynchrone) (type d'une trame de Type 13)
SoC	Start of cyclic (Début du cyclique) (type d'une trame de Type 13)
UDT	Unspecified-data transfer (Transfert de données non spécifiées)

3.5 Conventions communes

La présente norme emploie les conventions de description énoncées dans l'ISO/CEI 10731.

Le modèle de service, les primitives de service et les diagrammes de séquence temporelle utilisés sont des descriptions totalement abstraites; ils ne constituent pas une spécification pour une mise en œuvre.

Les primitives de service, utilisées pour représenter les interactions utilisateur de service/fournisseur de service (voir ISO/CEI 10731), acheminent des paramètres qui indiquent les informations disponibles dans l'interaction entre utilisateur et fournisseur.

La présente norme utilise un format de tableau pour décrire les paramètres de composants des primitives de DLS. Les paramètres qui s'appliquent à chaque groupe de primitives de DLS sont consignés en tableaux dans toute la suite de la présente norme. Chaque tableau comporte jusqu'à six colonnes, contenant le nom du paramètre de service, et une colonne chacune pour ces primitives et les sens de transfert de paramètres utilisés par le DLS:

- Les paramètres d'entrée de la primitive "request";
- Les paramètres de sortie de la primitive "request";
- Les paramètres de sortie de la primitive "indication";
- Les paramètres d'entrée de la primitive "response"; et

- Les paramètres de sortie de la primitive "confirm".

NOTE Les primitives "request", "indication", "response" et "confirm" sont aussi respectivement appelées primitives "requestor.submit", "acceptor.deliver", "acceptor.submit" et "requestor.deliver" (voir ISO/CEI 10731).

Un paramètre (ou une partie de celui-ci) est énuméré dans chaque rangée de chaque tableau. Dans les colonnes appropriées de la primitive de service, un code est utilisé pour spécifier le type d'usage du paramètre sur la primitive et le sens de paramètres spécifiés dans la colonne:

M Paramètre: obligatoire pour la primitive.

U Paramètre il s'agit d'une option de l'utilisateur qui peut ou peut ne pas être fournie, en fonction de l'usage dynamique de l'utilisateur de DLS. Lorsqu'il n'est pas fourni, une valeur par défaut est supposée pour le paramètre.

C le paramètre est conditionné à d'autres paramètres ou à l'environnement de l'utilisateur de DLS.

(Blanc/vide) Le paramètre n'est jamais présent.

Certaines entrées sont, en plus, qualifiées par des éléments entre parenthèses. Ceux-ci peuvent être

a) une contrainte spécifique à un paramètre:

(=) indique que le paramètre équivaut du point de vue de la sémantique au paramètre dans la primitive de service située immédiatement à sa gauche dans le tableau.

b) une indication qu'une certaine note s'applique à l'entrée

"(n)"indique que la note "n" suivante contient des informations complémentaires relatives au paramètre et à son utilisation.

Dans n'importe quelle interface particulière, il n'est pas indispensable d'énoncer tous les paramètres de façon explicite. Certains peuvent être associés de façon implicite au DLSAP où la primitive est émise.

Dans les diagrammes qui illustrent ces interfaces, des lignes tiretées indiquent des relations de cause à effet ou de temps-séquence tandis que les traits ondulés indiquent que des événements sont grosso modo contemporains.

3.6 Conventions supplémentaires

Dans les diagrammes qui illustrent les interfaces de DLS et de DLM, les lignes tiretées indiquent des relations de cause à effet ou de temps-séquence entre des actions dans des stations différentes, tandis que les lignes en traits pleins avec flèches indiquent des relations de cause à effet et de temps-séquence qui se produisent au sein du fournisseur de DLE dans une seule station.

La notation suivante, forme raccourcie des classes de primitives définies en 3.5, est utilisée dans les figures et dans les tableaux.

req	primitive "request"
ind	primitive "indication"
cnf	primitive "confirm" (confirmation)
res	primitive "response"

4 Vue d'ensemble du protocole de DL

4.1 Vue d'ensemble

Un bus de terrain de Type 13 étend Ethernet conformément à l'ISO/CEI 8802-3 avec des méthodes pour transférer des données en un temps prévisible et une synchronisation précise pour satisfaire aux demandes temporelles typiques pour une automatisation haute performance et des applications de mouvement. Cela ne modifie pas les principes de base de la Norme Fast Ethernet ISO/CEI 8802-3, mais l'étend vers Ethernet en temps réel. Ainsi, il est possible de tirer profit de et de continuer d'utiliser tout silicium d'Ethernet standard, composant d'infrastructure ou équipement d'essai et de mesure, tel qu'un analyseur de réseau.

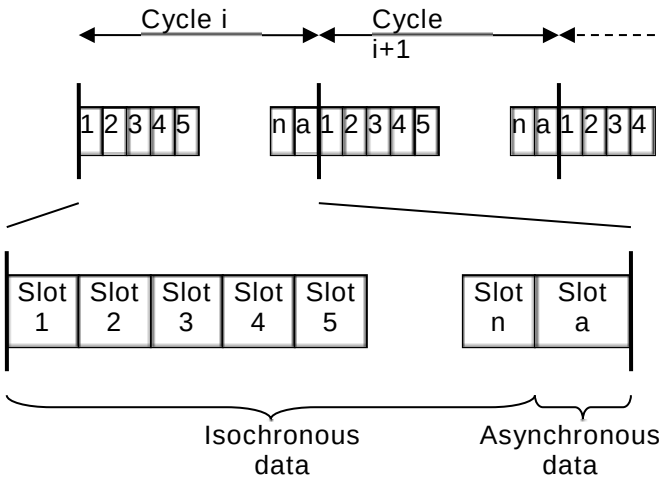
4.2 Description générale

4.2.1 Généralités

Le bus de terrain de Type 13 fournit des méthodes pour obtenir:

- La transmission des données prioritaires dans des cycles isochrones précis. L'échange des données est basé sur une relation édition/abonnement.
- La synchronisation des nœuds dans un réseau avec une haute précision.
- La transmission de moins de données prioritaires sur demande de manière asynchrone. La communication de données asynchrones peut être utilisée pour transférer des protocoles basés sur l'IP tels que TCP ou UDP et des protocoles de couches supérieures tels que HTTP, FTP,...

Le bus de terrain de Type 13 gère le trafic du réseau de façon à ce qu'il y ait des intervalles de temps dédiés aux données isochrones et asynchrones (voir Figure 2). Il veille à ce qu'un seul et unique appareil en réseau ait accès aux supports du réseau. Par conséquent, la transmission de données isochrones et celle des données asynchrones n'interfèreront jamais et un temps de communication précis est garanti. Le mécanisme s'appelle gestion du réseau de communication par intervalle de temps (SCNM). SCNM est géré par un appareil en réseau particulier – Le Nœud Gérant (MN) – qui inclut la fonctionnalité de MN. Tous les autres nœuds sont des nœuds commandés (CN).



Légende

Anglais	Français
Cycle i	Cycle i
Cycle i+1	Cycle i+1
Slot	Intervalle de temps

Anglais	Français
Isochronous data	Données isochrones
Asynchronous data	Données asynchrones

Figure 2 – Gestion du réseau de communication par intervalle de temps

L'accès au réseau est géré par un maître, le nœud gérant (MN) de bus de terrain de Type 13. Un nœud ne peut être autorisé à envoyer des données sur le réseau que par le MN. Les règles d'accès central excluent les collisions; par conséquent un réseau de Type 13 est donc déterministe.

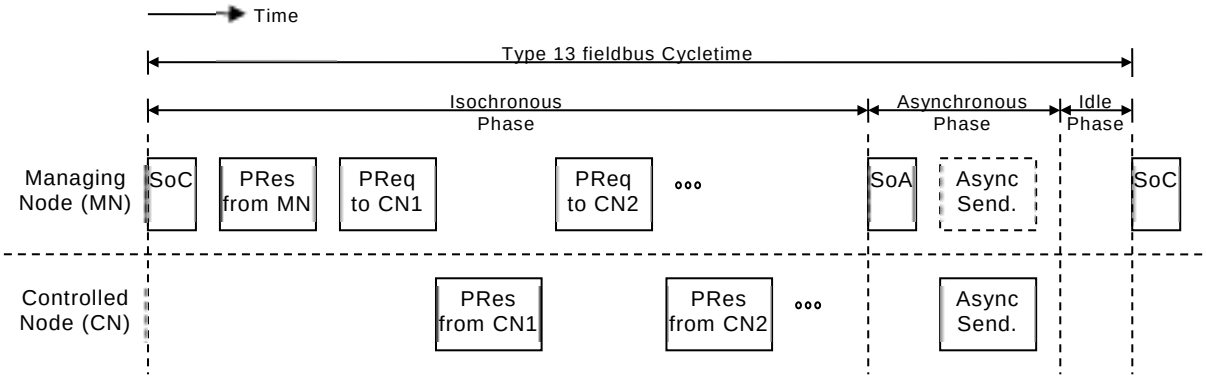
Les CN sont des nœuds de bus passifs; ils n'envoient que lorsque cela est requis par le MN.

4.2.2 Vue d'ensemble de la commande de l'accès aux supports et du protocole de transmission

La vue d'ensemble du flux des trames de données sur le support est présentée à la Figure 3.

L'échange de données entre les nœuds se produit de manière cyclique. Il est requis que la répétition survienne sur un intervalle fixe, appelé cycle de bus de terrain de Type 13. Les phases temporelles suivantes existent au sein d'un cycle de bus de terrain de Type 13.

- Phase isochrone
- Phase asynchrone
- Phase d'inactivité



Légende

Anglais	Français
Time	Temps
Type 13 fieldbus Cycletime	Durée de cycle de bus de terrain de Type 13
Isochronous Phase	Phase Isochrone
Asynchronous Phase	Phase Asynchrone
Idle Phase	Phase d'inactivité
Managing Node (MN)	Nœud Gérant (MN)
Controlled Node (CN)	Nœud Commandé (CN)
PReq to CN1	PReq au CN1
PReq to CN2	PReq au CN2
PRes from CN1	PRes de CN1
PRes from CN2	PRes de CN2

Figure 3 – Vue d'ensemble du flux de trames de données lors d'un cycle

4.2.2.1 Phase isochrone

La DLL doit donner l'occasion de transférer des données à chaque nœud dans un ordre séquentiel et dans une période prédéfinie. Au début d'un cycle de bus de terrain de Type 13, le MN doit envoyer une trame SoC à tous les nœuds par le biais de multidiffusion Ethernet. L'heure d'envoi et de réception de cette trame est la base pour l'heure commune de tous les nœuds. Seule la trame SoC est générée sur une base périodique. La génération de toutes les autres trames doit être contrôlée par les événements (avec une surveillance temporelle supplémentaire par nœud).

Le MN commence l'échange de données isochrones après que la trame SoC a été envoyée. Une trame PReq est envoyée à chaque nœud configuré et actif. Le nœud faisant l'objet d'un accès répond par une trame PRes.

Les trames PReq et PRes doivent transférer les données d'application. Le MN ne doit envoyer que les données PReq à un CN par trame. Le transfert de PReq est destiné uniquement aux données pertinentes pour le CN adressé. En revanche, la trame PRes est reçue par tous les nœuds. Cela rend les relations de communication possibles conformément au modèle producteur/consommateur.

La procédure PReq/PRes est répétée pour chaque CN isochrone configuré et actif. Le MN peut envoyer une trame PRes multidiffusion à tous les nœuds. Cette trame est destinée au transfert des données pertinentes pour les groupes de CN. L'ordre dans lequel les CN sont échangées sur interrogation (polled) et dans lequel la trame PRes du MN est envoyée, relève du domaine d'application de la présente norme.

La phase isochrone est calculée entre le début de la trame SoC et le début de la trame SoA.

La taille et la longueur du cycle de bus de terrain de Type 13 sont essentiellement affectées par la taille de la phase isochrone. Lors de la configuration du cycle de bus de terrain de Type 13, la somme des durées requises pour les accès PReq/PRes à chaque CN configuré doit être prise en compte. L'utilisation de la technologie d'accès multiplexé ou cadencé en continu diminue la durée.

L'accès continu, cadencé en continu et multiplexé fonctionne en parallèle lors d'un cycle de bus de terrain de Type 13. Le partage de la phase isochrone en intervalles de temps continus, cadencés en continu et multiplexés est spécifique à la configuration et ne relève pas du domaine d'application de la présente norme.

Bien que les nœuds multiplexés ne soient pas traités à chaque cycle, ils peuvent surveiller le transfert de données des nœuds continus dans son intégralité.

Dans le cas de perte de cycle MN, la séquence d'accès multiplexé est poursuivie sur une base temporelle, après que la phase d'erreur de perte de cycle soit terminée.

4.2.2.2 Phase asynchrone

Dans la phase asynchrone du cycle de bus de terrain de Type 13, l'accès au réseau est autorisé à un CN ou au MN pour le transfert de seulement un message asynchrone.

Il existe deux types de trames asynchrones disponibles:

- la trame ASnd (Async Send);
- un message d'Ethernet Traditionnel.

Dans la trame SoA, le MN accorde l'autorisation d'envoyer une trame asynchrone à un CN ou à lui-même. Si aucune demande de transmission de message asynchrone n'est en attente aux files d'attente de programmation du MN, le MN doit produire une SoA sans attribuer de

droit d'envoi à un nœud. Aucune trame ASnd et aucun message Ethernet traditionnel ne suivront la trame SoA dans ce cas.

La trame SoA est la première trame dans la phase asynchrone et constitue un signal informant tous les CN que toutes les données isochrones ont été échangées lors de la phase isochrone.

Dans le cas d'une trame ASnd de bus de terrain de Type 13, les données utilisateur peuvent contenir les données suivantes, selon la demande de MN et l'application.

- Ident Data (Données d'identification): Les données d'identification d'un CN.
- Status Data (Données de Statut): Le statut actuel et les informations détaillées relatives aux erreurs d'un nœud.
- Sync Data (Données de synchronisation): Les données de synchronisation d'un CN cadencé en continu.
- NMT Data (Données NMT): Une NMTCommand du MN à un ou plusieurs CN ou une NMTRequest d'un CN au MN.
- Unspecified Data (Données Non Spécifiées): Une communication avec des données non spécifiées peut être utilisée pour transférer une communication entre pairs avec accès au dictionnaire d'objets d'un appareil, aux protocoles basés sur l'IP tels que TCP ou UDP et aux protocoles de couches supérieures tels que HTTP, FTP, ...

Dans le cas d'un message Ethernet traditionnel, les données seraient susceptibles de contenir une trame Ethernet légale. Le message est transféré à l'application sans interprétation complémentaire.

La phase asynchrone est calculée entre le début de SoA et la fin de la réponse asynchrone.

4.2.2.3 Phase d'inactivité

La phase d'inactivité constitue l'intervalle de temps restant entre la fin de la phase asynchrone et le début du cycle suivant. Durant la phase d'inactivité, tous les composants du réseau "attendent" le début du cycle suivant. La durée de la phase d'inactivité peut être 0.

4.3 Service supposé de la PhL

Le Paragraphe 4.3 décrit le service physique supposé (PhS) et les contraintes utilisées par la DLE. Le Service Physique est supposé fournir les primitives de service suivantes spécifiées par l'Article 2 de l'ISO/CEI 8802-3:2000.

Les primitives supposées de PhS sont

- MA_DATA.request
- MA_DATA.indication

La relation temporelle des primitives est présentée à la Figure 4.

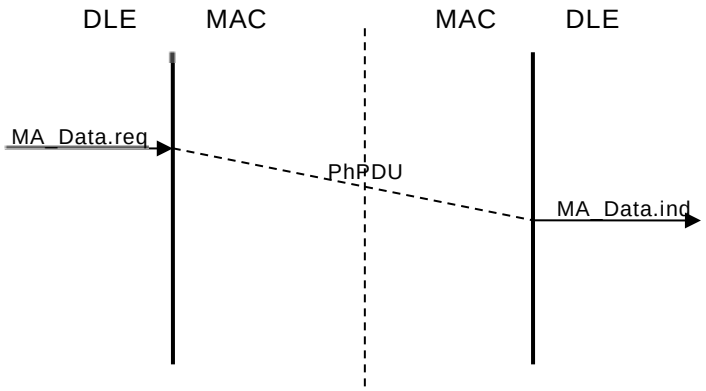


Figure 4 – Interaction des primitives de PhS avec la DLE

La primitive “request” MA_DATA définit le transfert de données d’une entité de client MAC à une seule ou plusieurs entités homologues dans le cas d'adresses groupées.

La primitive “indication” MA_DATA définit le transfert de données d’une entité de sous-couche MAC (à travers la sous-couche de commande MAC facultative, si mise en œuvre) à l’entité ou aux entités de client MAC dans le cas d'adresses groupées.

4.4 Architecture de la DLL

La DLL de bus de terrain de Type 13 est modélisée comme une association de composants de commande de diagramme d’états de cycle (CSM), de commande TX/RX en transmission isochrone (ITC), de commande TX/RX en transmission asynchrone (ITC), de programmateur d'intervalle de temps asynchrone (ASS), de signalisation d'exception (ES), de signalisation de NMT (NS) et d’interface de gestion de la DLL.

Le diagramme d’états de cycle en tant que composant de commande principal fournit la fonction pour une commande de l’accès au support déterministe contribuant à un support fiable et efficace des services de transfert de données de haut niveau. Selon l’attribution des responsabilités au MN et au CN, la principale responsabilité de CN est

- de prendre en charge la communication au sein d'un cycle de bus de terrain de Type 13;
- de suivre l’ordre des trames reçues au sein d’un cycle et de réagir selon le flux de données décrit;
- de générer un événement d’erreur en cas de détection d'une erreur de communication;
- de maintenir la communication quelles que soient les erreurs.

Le diagramme d’états de cycle du MN est chargé de:

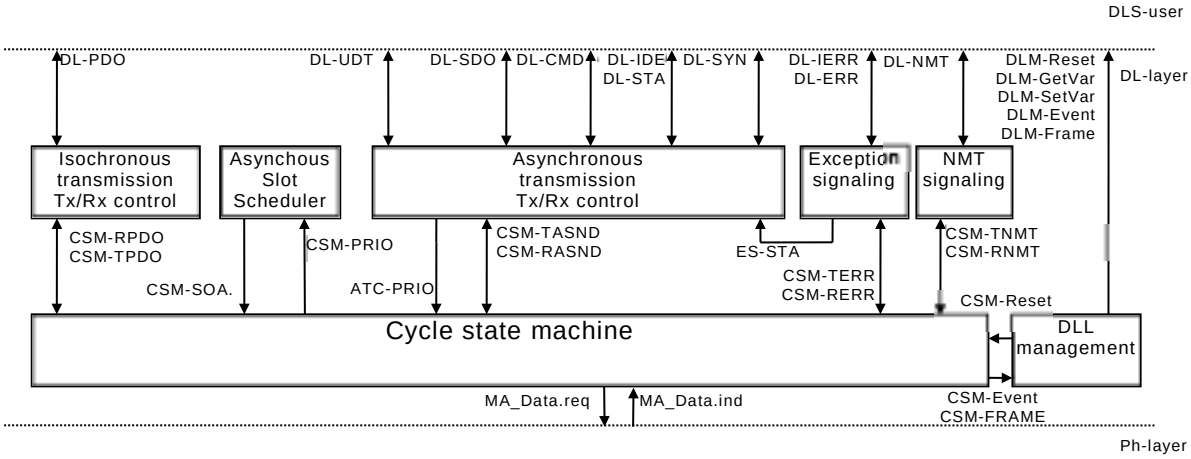
- La gestion de la communication au sein d'un cycle de bus de terrain de Type 13.
- La génération du flux de trames lors d’un cycle de bus de terrain de Type 13.
- La surveillance de la réaction des CN.
- La génération d’un événement d’erreur en cas de détection d'une erreur de communication;

La couche liaison de données se compose des composants énumérés dans le Tableau 1.

Tableau 1 – Composants de la couche liaison de données

Composants	Description
Diagramme d'états de cycle (CSM)	Le diagramme d'états de cycle commande le cycle de bus de terrain de Type 13 sur la couche liaison de données, assemble et transmet les DLPDU, reçoit et désassemble les DLPDU avec les informations de commande, et détermine l'heure et la durée des transmissions
Commande TX/RX en transmission isochrone (ITC)	L'ITC met en mémoire tampon et répartit en temps la DLSDU reçue pour le transfert de données cyclique prioritaires entre l'utilisateur de DLS et la SCM
Commande TX/RX en transmission asynchrone (ATC)	L'ATC met en file d'attente et répartit dans le temps les DLSDU reçues pour le transfert de données asynchrones prioritaires entre l'utilisateur de DLS et la SCM
Programmateur d'intervalle de temps asynchrone (ASS)	Le programmateur d'intervalle de temps asynchrone des MN décide du moment de survenue d'un transfert de données asynchrones
Signalisation d'exception (ES)	La signalisation d'exception signale une exception de CN détectée au NMT du MN
Signalisation de NMT (NS)	Le NS rapporte le statut actuel du NMT
Interface de Gestion de la DLL	L'interface de gestion de la DLL maintient les variables de gestion de la station qui appartiennent à la DLL, et gère des modifications synchronisées des paramètres de liaison

L'agencement interne de ces composants, et de leurs interfaces, est présenté à la Figure 5. Les pointes de flèches représentent le sens principal du flux de données et de commande.



Légende

Anglais	Français
DLS-user	Utilisateur de DLS
DL-layer	Couche DL
Isochronous transmission Tx/Rx control	Commande Tx/Rx en transmission isochrone
Asynchronous Slot Scheduler	Programmateur d'intervalle de temps asynchrone
Asynchronous transmission Tx/Rx control	Commande Tx/Rx en transmission asynchrone
Exception signaling	Signalisation d'exception
Cycle state machine	Diagramme d'états de cycle
DLL management	Gestion de la DLL
Ph-layer	Couche Ph

Figure 5 – Architecture interne de la couche liaison de données

4.5 Paramètres et variables locaux

Cette spécification utilise des paramètres de demande de l'utilisateur de DLS P(...) et des variables locales V(...) pour clarifier l'effet de certains actions ainsi que les conditions dans lesquelles ces actions sont valides, des temporisateurs locaux T(...) pour surveiller les actions du fournisseur de DLS distribué et pour garantir une réponse de DLE locale en cas d'absence de ces actions, et des compteurs locaux C(...) pour exécuter des fonctions de mesure de taux.

Sauf spécification contraire, au moment de leur création ou de l'activation de la DLE:

a) toutes les variables doivent être initialisées à leur valeur par défaut, ou à leur valeur minimale autorisée si aucune valeur par défaut n'est précisée;

b) tous les compteurs doivent être initialisés à zéro;

c) tous les temporisateurs doivent être initialisés à zéro;

La gestion de DL peut modifier les valeurs des variables de configuration.

4.5.1 Variables, paramètre, compteur et temporisateur pour prendre en charge la fonction de la DLE

4.5.1.1 T(ASND_TIME)

T(ASND_TIME) est utilisé par la DLE pour mesurer le temps écoulé depuis le dernier envoi d'une trame SoA. La valeur est diminuée dans la plage de V(AsyncSlotTimeout_U32) à 0.

4.5.1.2 V(AsyncSlotTimeout_U32)

Cette variable porte et désigne la valeur du temps d'attente pour la trame ASnd en ns. Un événement est généré lorsque la trame ASnd n'a pas été (ou pas complètement) reçue dans un temps prédéfini. La valeur initiale est 100 000 lors de la mise sous tension ou de la réinitialisation de ce nœud. La plage de cette valeur commence à 250.

4.5.1.3 P(C_DLL_ISOCHR_MAX_PAYL)

Ce paramètre indique la taille maximale des données de charge utile de PReq et PRes en octets. La valeur est mise à 1 490.

4.5.1.4 P(C_DLL_MAX_PAYL_OFFSET)

Ce paramètre indique le décalage maximal de la charge utile de la trame Ethernet en octets. La valeur est mise à 1 499.

4.5.1.5 P(D_DLL_CNFeatureMultiplex_BOOL)

Ce paramètre indique la capacité de CN à effectuer la commande de communication isochrone multiplexée.

4.5.1.6 P(D_DLL_FeaturePResTimeTriggered_BOOL)

Ce paramètre indique la capacité d'un nœud à effectuer la commande de communication isochrone cadencée en continu.

4.5.1.7 P(D_DLL_MNFeatureMultiplex_Bool)

Ce paramètre indique la capacité des MN à effectuer la commande de communication isochrone multiplexée.

4.5.1.8 P(D_NMT_CycleTimeMax_U32)

Ce paramètre représente la durée de cycle de bus de terrain de Type 13 maximal en µs.

4.5.1.9 P(D_NMT_CycleTimeMin_U32)

Ce paramètre représente la durée de cycle de bus de terrain de Type 13 minimal en µs.

4.5.1.10 P(D_NMT_NetTime_BOOL)

Ce paramètre indique la capacité des MN à prendre en charge la transmission de “NetTime” par le biais de la DLPDU SoC.

4.5.1.11 P(D_NMT_NetTimeIsRealTime_BOOL)

Ce paramètre indique la capacité des MN à prendre en charge le temps réel par le biais du paramètre “NetTime” dans la DLPDU SoC.

4.5.1.12 P(D_NMT_RelativeTime_BOOL)

Ce paramètre indique la capacité des MN à prendre en charge la transmission de “RelativeTime” par le biais de la DLPDU SoC.

4.5.1.13 T(FRAME_TIME)

T(FRAME_TIME) est utilisé par la DLE pour mesurer le temps écoulé depuis la dernière réception d’une trame SoC, SoA et PReq. La valeur est diminuée dans la plage de V(FRAME_TIMEOUT) à 0.

4.5.1.14 V(FRAME_TIMEOUT)

Cette variable comporte et désigne la valeur du temps d’attente de la trame SoC, SoA et PReq. Un événement est généré lorsqu’une des trames n’a pas été (ou pas complètement) reçue dans un temps prédéfini. Cet événement signifie qu’une trame obligatoire du MN a été perdue.

4.5.1.15 V(MultiplCycleCnt_U8)

Cette variable décrit la longueur du cycle multiplexé en multiples de cycle de bus de terrain de Type 13. Il convient que la variable soit fixée par la configuration du système et soit égale dans tous les nœuds du segment. Si cette variable est zéro, il n’y a pas de prise en charge du cycle multiplexé sur le réseau.

4.5.1.16 V(NMT_CycleLen_U32)

Cette variable définit l’intervalle de temps du cycle de communication (intervalle de synchronisation) en µs. Il convient que la variable soit fixée par la configuration du système dans la plage de P(D_NMT_CycleTimeMin_U32) à P(D_NMT_CycleTimeMax_U32).

4.5.1.17 V(NMT_Version_U8)

La variable comporte la version du profil de communication de bus de terrain de Type 13 qui est mise en œuvre par l’appareil. La valeur doit être fixée par l’appareil lors de l’initialisation du système.

4.5.1.18 V(NMT_IsochrSlotAssign_AU8)

Cette variable attribue des CN à un intervalle de temps isochrone particulier. Il convient que la variable soit fixée par la configuration du système dans la plage de 0 à 254.

4.5.1.19 V(NMT_MultiplCycleAssign_AU8)

Cette variable attribue des CN aux cycles de bus de terrain de Type 13 particuliers de la période de cycles multiplexé définie par V(MultiplCycleCnt_U8). Elle doit être égale dans tous les nœuds du segment. Il convient que la variable soit fixée par la configuration du système dans la plage de 0 à V(MultiplCycleCnt_U8).

4.5.1.20 V(NodeID_U8)

La variable comporte l'ID de Nœud réel de l'appareil. V(NodeID_U8) peut être fourni par des paramètres d'équipements (commutateur à positions multiples etc.) ou configuré par le logiciel dans la plage de 1 à 240, 253 et 254.

4.5.1.21 T(PRES_TIME)

T(PRES_TIME) est utilisé par la DLE pour mesurer le temps écoulé depuis le dernier envoi d'une trame PReq. La valeur est diminuée dans la plage de V(PRES_TIMEOUT) à 0.

4.5.1.22 T(PRES_TT_TIME)

T(PRES_TT_TIME) est utilisé par la DLE d'un CN pour mesurer le temps écoulé depuis la dernière réception d'une trame PRes du MN. La valeur est diminuée dans la plage de V(PRES_TT_TIMEOUT) à 0.

4.5.1.23 V(PRES_TIMEOUT)

Cette variable comporte et désigne la valeur du temps d'attente de la trame PRes en µs. Un événement est généré lorsque la trame PRes n'a pas été (ou pas complètement) reçue dans un temps prédéfini.

4.5.1.24 V(PRES_TT_TIMEOUT)

Cette variable porte et désigne la valeur du temps d'attente en µs d'un CN cadencé en continu pour envoyer sa trame PRes.

4.5.1.25 V(Prescaler_U16)

Cette variable décrit le taux de basculement de l'indicateur PS au sein de la DLPDU SoC. La valeur fournit le nombre de cycles de bus de terrain de Type 13 devant être effectués pour basculer l'indicateur par le MN. La valeur initiale est 2 lors de la mise sous tension ou de la réinitialisation de ce nœud. La plage de cette valeur est 0 à 1 000.

Si cette variable est 0, il ne doit pas y avoir de basculement de l'indicateur de PS. V(Prescaler_U16) doit être égale dans tous les nœuds du segment.

5 Structure générale et codage des PhPDU, des DLPDU et des éléments de procédure associés

5.1 Vue d'ensemble

La DLL et ses procédures sont nécessaires pour fournir les services offerts à l'utilisateur de DLS en utilisant les services disponibles à la PhL. L'Article 5 décrit la structure et la sémantique de MA_PDU, DLPDU et la procédure, généralement utilisée dans cette spécification. Néanmoins, cette partie est identique et complètement conforme à la norme ISO/CEI 8802-3.

NOTE Au sein de l'Article 5, toute référence au bit *k* d'un octet est une référence au bit dont le poids dans entier non signé à un octet est 2^k, et cela est parfois référencé comme la numérotation binaire "petit boutiste".

5.2 Structure et codage de MA_PDU

La sous-couche MAC locale utilise les primitives de service fournies par la sous-couche PhS spécifiée par l'ISO/CEI 8802.3 Article 2. Toutes les primitives de service fournies par la sous-couche PhS sont comme suit et sont considérées comme obligatoires:

a) MA_DATA request;

b) MA_DATA indication;

5.3 Structure, codage et éléments de procédure de la trame MAC commune

5.3.1 Structure de la trame MAC

5.3.1.1 Format de la trame MAC pour la DLPDU de bus de terrain de Type 13

La DLPDU pour le bus de terrain de Type 13 est encapsulée dans le champ de données d'une trame MAC comme spécifié par l'ISO/CEI 8802.3, Article 3. La valeur du champ "Length/Type" est désignée à $88AB_{Hn}$ qui est autorisé et enregistré comme le numéro d'identification du protocole par l'Autorité d'Enregistrement de l'IEEE, pour être identifié comme la trame de bus de terrain de Type 13. La Figure 6 présente la DLPDU de bus de terrain de Type 13.

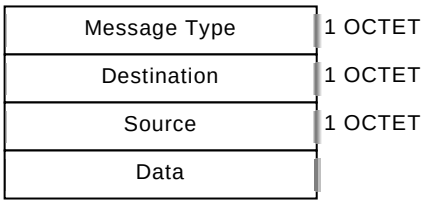


Figure 6 – DLPDU de bus de terrain de Type 13

La longueur de la trame doit être réduite à la taille configurée. Néanmoins, la durée de cycle ne peut pas être garantie. Les trames Ethernet ne doivent pas être inférieures au minimum de 64 octets spécifié.

5.3.1.2 DLPDU de bus de terrain de Type 13 du format de la trame MAC

Toute trame MAC comme spécifiée par l'ISO/CEI 8802-3, Article 3 s'applique, à l'exception des trames MAC avec un champ "Length/Type" désigné sur $88AB_{H.}$

5.3.2 Éléments de la trame MAC

5.3.2.1 Généralités

Ce paragraphe spécifie l'utilisation par le bus de terrain de type 13 des éléments de trame MAC conformément à l'ISO/CEI 8802-3.

5.3.2.2 Champ "Destination Address" (Adresse de Destination) (DA)

Le champ "Destination Address" (DA) est spécifié dans l'ISO/CEI 8802-3:2000, Article 3. Le champ "Destination Address" spécifie la (les) station(s) à laquelle (auxquelles) la trame est destinée. Il peut s'agir d'une adresse individuelle ou de multidiffusion (y compris de diffusion).

Dans le cas de DLPDU de bus de terrain de Type 13, le Tableau 2 présente les adresses de multidiffusion de MAC, lorsque la destination est mise à "Diffusion de bus de terrain de Type 13" (voir 5.3.3.2). Sinon, la DA est mise à l'ID de nœud correspondant, résolu par la couche Application.

Tableau 2 – Adresses de multidiffusion de MAC

Type de trame	ID/Abr.	Adresse de multidiffusion de MAC
Start of cycle (SoC)	SoC	01-11-1E-00-00-01 (C_DLL_MULTICAST_SOC)
PollResponse (PRes)	PRes	01-11-1E-00-00-02 (C_DLL_MULTICAST_PRES)
Start of Asynchronous (SoA)	SoA	01-11-1E-00-00-03 (C_DLL_MULTICAST_SOA)
AsynchronousSend (ASnd)	ASnd	01-11-1E-00-00-04 (C_DLL_MULTICAST_ASND)

5.3.2.3 Champ “Source Address” (Adresse Source) (SA)

Le champ “Source Address” (DA) est spécifié dans l’ISO/CEI 8802-3:2000, Article 3. Le champ “Source Address” spécifie la station envoyant la trame.

5.3.2.4 Champ “Length/type” (Longueur/type)

Le champ “Length/type” est spécifié dans l’ISO/CEI 8802-3:2000, Article 3. Afin d’être identifiée comme trame de bus de terrain de Type 13, la valeur du champ “length/type” est mise à $88AB_H$, qui est agréée et enregistrée comme numéro d’identification du protocole pour le bus de terrain de Type 13 par l’autorité d’enregistrement de l’IEEE. Chaque trame n'ayant pas une valeur égale à $88AB_H$ est traitée comme une trame Ethernet traditionnelle au sein d'un intervalle de temps asynchrone.

5.3.3 Éléments de la DLPDU de bus de terrain de Type 13

Ce paragraphe spécifie les éléments d'une DLPDU de bus de terrain de Type 13, comme illustré à la Figure 6.

5.3.3.1 Message type (Type de message)

Le champ “message type” (type de message) est utilisé par la DLE pour identifier et désigner le type de trame du bus de terrain de Type 13 pour la commande de l’accès au support. Le Tableau 3 énumère la liste des types de message pour le bus de terrain de Type 13.

Tableau 3 – Types de message

Type de Message	Valeur	ID/Abr.
Start of Cycle	01h	SoC
PollRequest	03h	PReq
PollResponse	04h	PRes
Start of Asynchronous	05h	SoA
Asynchronous Send	06h	ASnd

5.3.3.2 Destination

ID de nœud du (des) nœud(s) adressé(s). Le champ “destination address” spécifie la (les) station(s) à laquelle (auxquelles) la trame est destinée. Il peut s’agir d’une adresse individuelle ou de diffusion.

Le Tableau 4 présente la liste des ID de nœud pour la (les) adresse(s) de destination.

Tableau 4 – Attribution d'ID de nœud

ID de nœud	Description
0	Non valide
1..239	CN ordinaires de bus de terrain de Type 13
240	MN de bus de terrain de Type 13
241..250	Réservé
251	Pseudo ID de nœud à utiliser par un nœud pour s'adresser lui même
252	ID de nœud de simulation
253	CN de Type 13 (Appareil de diagnostic)
254	CN de Type 13 (Appareil d'acheminement)
255	Diffusion de bus de terrain de Type 13

5.3.3.3 Source

ID de nœud du (des) nœud(s) émettant. Le champ "source address" (adresse source) spécifie la station envoyant la trame. Dans le cas d'un message de diffusion/multidiffusion transmis, l'adresse source est interprétée par la DLE. La valeur de l'adresse source est toujours mise à V(NodeID_U8).

Le Tableau 4 présente la liste des ID de nœud pour l'adresse source. Les ID de nœud 251, 252, et 255 (diffusion) ne doivent pas être utilisés comme adresse source.

5.3.3.4 Data (Données)

Le champ "data" (données) contient une séquence de *N* octets qui fournit une transparence complète des données dans le sens où toute séquence de valeurs d'octets arbitraire peut apparaître dans le champ "data" jusqu'à un nombre maximal spécifié par l'ISO/CEI 8802-3. Une taille de trame minimale est requise, c'est-à-dire minFrameSize par l'ISO/CEI 8802-3, et si la taille d'une trame est inférieure à minFrameSize, alors le champ "data" est étendu en rattachant des bits supplémentaires dans les unités d'octets.

La structure de ce champ pour la DLPDU de bus de terrain de Type 13 est décrite à l'Article 6.

5.4 DLPDU non valide

Une DLPDU non valide doit être définie comme celle qui satisfait au moins à une des conditions suivantes.

a) La longueur de trame est incohérente avec une valeur de la longueur spécifiée dans le champ "Length/type". Si le champ "Length/type" contient une valeur type comme définie par l'ISO/CEI 8802-3:2000, 3.2.6, alors la longueur de la trame est supposée conforme à ce champ et il convient qu'elle ne soit pas considérée comme une DLPDU non valide sur cette base.

b) Il ne s'agit pas d'un nombre entier d'octets en longueur.

c) Les bits de la DLPDU entrante (exclusive du champ FCS lui-même) ne génèrent pas de valeur CRC identique à celle reçue.

d) Elle est incompatible avec une valeur de du champ "Message Type" de la DLPDU de bus de terrain de Type 13.

Les contenus de la DLPDU non valide ne doivent pas être transmis à l'utilisateur de DLS ou à la DLE. L'occurrence de la DLPDU non valide peut être communiquée à la gestion de réseau.

NOTE Une DLPDU non valide peut être ignorée, rejetée, ou utilisée de manière privée par les utilisateurs de DLS. L'utilisation de ces DLPDU relève du domaine d'application de la présente spécification.

6 Structure, codage et éléments de procédure spécifiques à une DLPDU

6.1 Généralités

L'Article 6 définit la structure, le contenu et le codage de chaque type et format de la DLPDU, et spécifie les éléments de procédure pour la DLPDU.

Au sein de chaque paragraphe, la structure, le contenu, les paramètres et le codage de la DLPDU sont décrits, et la partie spécifique du bus de terrain de Type 13 de la structure de la DLPDU, présentée à la Figure 6, est spécifiée. Les aspects liés à l'envoi et à la réception des utilisateurs de DLS et de leurs DLE sont décrits ultérieurement. Tous les formats et codages de données sont décrits dans un format petit boutiste («little endian») tout au long de l'Article 6.

NOTE Au sein de l'Article 6, toute référence au bit K d'un octet est une référence au bit dont le poids dans un entier non signé à un octet est 2^K, et cela est parfois référencé comme la numérotation binaire "petit boutiste".

6.2 Vue d'ensemble

Le bus de terrain de Type 13 recueille plus d'une fonction dans une trame. Il n'est donc généralement pas possible d'appliquer une relation de communication unique à la trame complète, mais seulement à des services particuliers à l'intérieur de la trame.

La DLPDU PRes par exemple transmise par le CN inclut plusieurs services.

- La transmission du statut actuel de NMT du CN est la partie réponse d'une relation maître/esclave non confirmée déclenchée par le MN.
- La demande de l'intervalle de temps asynchrone est la partie demande d'une relation client/serveur.
- La transmission des données PDO est effectuée conformément à une relation producteur/consommateur du modèle de transmission.

6.3 Début de la Synchronisation (SoC)

6.3.1 Généralités

La DLPDU SoC est utilisée pour la synchronisation de tous les nœuds. Seul le MN doit lancer la DLPDU SoC.

Au début d'un cycle de bus de terrain de Type 13, le MN doit envoyer une DLPDU SoC à tous les nœuds. L'heure d'envoi et de réception de cette trame est la base pour l'heure commune de tous les nœuds. Seule la DLPDU SoC est générée sur une base périodique. La génération de toutes les autres trames doit être contrôlée par les événements (avec une surveillance temporelle supplémentaire par nœud).

Le MN commence l'échange de données isochrones après que la trame SoC a été envoyée.

6.3.2 Structure de la DLPDU SoC

La structure de la DLPDU SoC est présentée dans le Tableau 5.

Tableau 5 – Structure de la DLPDU SoC

Champ de la DLPDU	Type de données	Valeur/description
Message type	OCTET[1]	01h/Identification de la DLPDU SoC
Destination	OCTET[1]	ID de nœud pour le nœud adressé
Source	OCTET[1]	ID de nœud du nœud émettant
reserved	OCTET[1]	
SOC-flag	OCTET[1]	
reserved	OCTET[1]	
NetTime	OCTET[8]	Heure de début du cycle de bus de terrain de Type 13
RelativeTime	OCTET[8]	Temps relatif
reserved	OCTET[24]	

6.3.2.1 Paramètres de la DLPDU SoC

6.3.2.1.1 Destination (dest)

Ce paramètre indique l'ID de nœud du nœud adressé (voir 5.3.3.2). Pour la DLPDU SoC, le paramètre "destination" est mis à 255 (C_ADDR_BROADCAST).

6.3.2.1.2 Source (src)

Ce paramètre indique l'ID de nœud du nœud émettant (voir 5.3.3.3). Pour la DLPDU SoC, le paramètre est mis à 240 (C_ADDR_MN_DEF_NODE_ID).

6.3.2.1.3 SOC-Flag

6.3.2.1.3.1 Généralités

La structure de SoC-Flag est présentée dans le Tableau 6.

Tableau 6 – Structure de SoC-Flag

Numéro de bit	Valeur	Description
7		Cycle multiplexé complet (Multiplexed cycle complete (MC))
6		Intervalle préétalonné (Prescaled slot (PS))
5-0		Réservé

6.3.2.1.3.2 Multiplexed cycle complete (MC)

Ce paramètre est basculé par le MN dès que le cycle multiplexé final a été terminé. Le paramètre est généré au sein de l'ITC et signalé à l'application au sein du statut de trame de DLMS.

6.3.2.1.3.3 Prescaled slot (PS)

Ce paramètre est basculé par le MN tous les *n* cycles (n est configurable par V(Prescaler_U16))

Le paramètre est signalé à l'application au sein du statut de trame de DLMS.

NOTE Ce signal préétalonné est utile pour les nœuds "lents" qui ne peuvent pas réagir à chaque cycle.

6.3.2.1.4 NetTime (time)

Ce paramètre est distribué par le MN et indique l'heure de début du cycle de bus de terrain de Type 13.

La transmission de "NetTime" est facultative. La prise en charge est indiquée par P(D_NMT_NetTime_BOOL).

La distribution conforme à la CEI 61588 par le biais du paramètre "NetTime" est indiquée par P(D_NMT_NetTimeIsRealTime_BOOL).

6.3.2.1.5 RelativeTime (reltime)

Ce paramètre est distribué par le MN et indique le temps relatif, qui est incrémenté du durée de cycle de Type 13 lorsqu'une DLPDU SoC est générée. L'unité de "RelativeTime" est μs.

La transmission de "RelativeTime" est facultative. La prise en charge est indiquée par P(D_NMT_RelativeTime_BOOL).

6.3.2.2 Données utilisateur

Aucune donnée utilisateur n'est transportée par la DLPDU SoC.

6.3.3 DLPDU SoC expéditrice

Le CSM du MN envoie la DLPDU SoC avec un ensemble de paramètres, comme stipulé ci-dessus. L'intervalle de temps de la base périodique est déterminé par V(NMT_CycleLen_U32).

6.3.4 DLPDU SoC destinataire

Chaque nœud reçoit cette DLPDU pour synchroniser son comportement sur cette DLPDU. Le CSM de la CN utilise cette DLPDU pour:

- Indiquer le début d'un cycle de bus de terrain de Type 13.
- Reconnaître les exceptions.
- Dépasser la durée de cycle.

La réception et le paramètre de cette DLPDU sont signalés par le statut de la trame DLMS à des fins de synchronisation.

Le MN doit observer le trafic du réseau afin de garantir qu'il n'y a pas d'autre MN actif sur le réseau. La réception d'une DLPDU SoC ou SoA indique qu'un autre MN est actif.

6.4 PollRequest (PReq)

6.4.1 Généralités

Le transfert de données en temps réel est effectué à l'aide d'objets de données de processus (PDO). La communication des PDO dans le bus de terrain de Type 13 est toujours effectuée de façon isochrone par les DLPDU PReq et PRes.

Le MN 'envoie seulement la DLPDU PReq à un seul CN par trame. Le transfert de PReq est destiné uniquement aux données pertinentes pour le CN adressé. Le CN adressé répond par son objet de données de processus à tous les nœuds. Cela rend les relations de communication possibles conformément au modèle producteur/consommateur. La procédure PReq/PRes doit être répétée pour chaque CN isochrone configuré et actif durant chaque cycle. Le MN peut envoyer une DLPDU PRes à tous les CN.

Une attribution ultérieure de la DLPDU PReq est la signalisation d'exception. Un indicateur à l'intérieur de la DLPDU est utilisé pour accuser réception d'un signal d'exception par un CN particulier au CN particulier.

6.4.2 Structure de la DLPDU PReq

6.4.2.1 Généralités

La structure de la DLPDU PReq est présentée dans le Tableau 7.

Tableau 7 – Structure de la DLPDU PReq

Champ de la DLPDU	Type de données	Valeur/description
Message type	OCTET[1]	03h/identification de la DLPDU PReq
Destination	OCTET[1]	ID de nœud du nœud adressé
Source	OCTET[1]	ID de nœud du nœud émettant
reserved	OCTET[1]	
PReq-flag	OCTET[1]	
reserved	OCTET[1]	
Payload	OCTET[4- P(C_DLL_MAX_PAYL_OFFSET)-10]	DLSDU

6.4.2.2 Paramètres de la DLPDU PReq

6.4.2.2.1 Destination (dest)

Ce paramètre indique l'ID de nœud du nœud adressé (voir 5.3.3.2). Pour la DLPDU PReq, seuls les ID de nœud des CN individuels sont autorisés.

Ce paramètre est inséré/évalué dans/à partir de l'Utilisateur de DLS comme le paramètre "D_addr" au sein de DL-PDO de DLS.

6.4.2.2.2 Source (src)

Ce paramètre indique l'ID de nœud du nœud émettant (voir 5.3.3.3). Pour la DLPDU PReq, le paramètre est mis à 240 (C_ADDR_MN_DEF_NODE_ID).

Ce paramètre est transmis à l'Utilisateur de DLS comme le paramètre "S_addr" au sein de DL-PDO de DLS.

6.4.2.2.3 PRreq-Flag

6.4.2.2.3.1 Généralités

La structure de PReq-Flag est présentée dans le Tableau 8.

Tableau 8 – Structure de PReq-Flag

Numéro de bit	Valeur	Description
7-6		Réservés
5		Multiplexed slot (MS)
	0	Intervalle de temps isochrone pris en charge dans un intervalle non multiplexé
	1	Intervalle de temps isochrone pris en charge dans un intervalle multiplexé
4-3		Réservés
2		Exception acknowledge (EA)

Numéro de bit	Valeur	Description
1		Réservé
0		Ready (RD)
	0	DLSDU n'est pas valide
	1	DLSDU est valide

6.4.2.2.3.2 Multiplexed slot (MS)

Ce paramètre est mis par le MN lorsque le CN adressé est pris en charge dans un intervalle de temps multiplexé. Le paramètre est généré au sein de l'ITC et signalé à l'application au sein du statut de DL-PDO de DLS.

6.4.2.2.3.3 Exception acknowledge (EA)

Ce paramètre est basculé par le MN pour accuser réception d'un signal d'exception requis à partir du CN adressé. Le paramètre est généré/évalué au sein de l'ES.

Pour les CN uniquement asynchrones, le paramètre est aussi signalé au sein de la DLSDU de DL-STA. Le CSM est chargé d'insérer ce paramètre dans la DL-STA et de le retirer.

6.4.2.2.3.4 Ready (RD)

Ce paramètre indique que les données de charge utile transférées sont valides. Le paramètre est transmis à/reçu de l'Utilisateur de DLS au sein de la primitive DL-PDO de DLS.

6.4.2.2.4 Payload (pl)

La DLPDU PReq peut transmettre des données utilisateur de longueur pouvant atteindre P(C_DLL_ISOCHR_MAX_PAYL). Le paramètre est transmis à/reçu de l'Utilisateur de DLS au sein de la primitive DL-PDO de DLS.

6.4.3 Envoi de la DLPDU PReq

Une DLPDU PReq doit être envoyée à chaque nœud configuré et actif lors de chaque cycle de bus de terrain de Type 13 en prenant en compte l'intervalle multiplexé. Une DLPDU PReq ne doit pas être envoyée à un CN configuré comme un nœud cadencé en continu.

Le CSM du MN gère l'accès à tous les CN par le biais de la DLPDU PReq lors d'un cycle multiplexé. Pour chaque intervalle, le CSM des MN demande les DLSDU expéditrices par le biais de l'ITC et les assemble à la DLPDU PReq comprenant les informations de signalisation d'exception.

Le MN utilise un temps d'attente après l'envoi d'une DLPDU PReq pour recevoir la réponse DLPDU PRes afin de détecter les erreurs de transmission et les défaillances de nœuds.

6.4.4 Réception de la DLPDU PReq

Chaque CN continu ou multiplexé isochrone doit recevoir une DLPDU PReq du MN dans le cycle de bus de terrain de Type 13 et doit envoyer une DLPDU PRes au MN. Les CN peuvent être accessibles à chaque cycle ou tous les n cycles (nœuds multiplexés, n > 1).

Les données reçues sont décompressées et transmises directement à l'ITC par le biais de CSM-RPDO. Le paramètre d'accusé de réception de l'exception est transmis à l'EX par le biais de CSM-ERR pour un traitement ultérieur.

6.5 Poll response (PRes)

6.5.1 Généralités

Chaque CN continu ou multiplexé isochrone doit recevoir des DLPDU PReq de diffusion unique du MN dans le cycle de bus de terrain de Type 13 et doit répondre avec une DLPDU PRes au MN. Chaque CN isochrone cadencé en continu doit recevoir la DLPDU PRes du MN dans le cycle de bus de terrain de Type 13 et doit envoyer une DLPDU PRes au MN de manière cadencée. Les DLPDU PReq et PRes peuvent transporter des données isochrones.

PReq ne peut être reçue que par le CN spécifiquement adressé. Cependant, la DLPDU PRes doit être envoyée par le CN en tant que message de multidiffusion, permettant ainsi à tous les autres CN de surveiller les données envoyées.

Les données supplémentaires du MN peuvent être reçues par une DLPDU PRes multidiffusion transmise par le MN.

Les CN participant dans l'intervalle de temps isochrone doivent demander le droit de transmettre des données asynchrones à partir du MN par le biais du paramètre PR/PS de la DLPDU PRes.

De nouvelles conditions d'exception et le statut actuel de NMT du CN sont également transmis au sein de la DLPDU PRes.

6.5.2 Structure de la DLPDU PRes

6.5.2.1 Généralités

La structure de la DLPDU PRes est présentée dans le Tableau 9.

Tableau 9 – Structure de la DLPDU PRes

Champ de la DLPDU	Type de données	Valeur/description
Message type	OCTET[1]	04h/identification de la DLPDU PRes
Destination	OCTET[1]	ID de nœud du nœud adressé.
Source	OCTET[1]	ID de nœud du nœud émettant
NMTStatus	OCTET[1]	Statut du diagramme d'états NMT des CN
PRes-flag	OCTET[2]	
Payload	OCTET[4- P(C_DLL_MAX_PAYL_OFFSET)-10]	DLSDU

6.5.2.2 Paramètres de la DLPDU PRes

6.5.2.2.1 Destination (dest)

Ce paramètre indique l'ID de nœud du nœud adressé (voir 5.3.3.2). Pour la DLPDU PRes, le paramètre "destination" est mis à 255 (C_ADDR_BROADCAST).

Ce paramètre est inséré/évalué dans/à partir de l'Utilisateur de DLS comme le paramètre D_addr au sein de DL-PDO de DLS.

6.5.2.2.2 Source (src)

Ce paramètre indique l'ID de nœud du nœud émettant (voir 5.3.3.3).

Ce paramètre est transmis à l'Utilisateur de DLS comme le paramètre "S_addr" au sein de DL-PDO de DLS.

6.5.2.2.3 NMTStatus (stat)

Ce paramètre rapporte le statut actuel du diagramme d'état NMT des CN et est inséré/évalué dans/à partir de l'Utilisateur de DLS avec DL-NMT de DLS.

6.5.2.2.4 PRes-Flag

6.5.2.2.4.1 Généralités

La structure de PRes-Flag est présentée dans le Tableau 10.

Tableau 10 – Structure de PRes-Flag

Numéro de bit	Valeur	Description
15-14		Réservé
13		Multiplexed slot (MS)
	0	Intervalle de temps isochrone pris en charge dans un intervalle non multiplexé
	1	Intervalle de temps isochrone pris en charge dans un intervalle multiplexé
12		EN
11-9		Réservé
8		Ready (RD)
	0	DLSDU n'est pas valide
	1	DLSDU est valide
7-6		Réservé
5-3		Priority (PR)
	000	PRIO_0
	001	PRIO_1
	010	PRIO_2
	011	PRIO_GENERIC_REQUEST
	100	PRIO_4
	101	PRIO_5
	110	PRIO_6
	111	PRIO_NMT_REQUEST
2-0		RequestToSend (RS)
	000	Aucune demande en cours
	001	1 demande en cours
	010	2 demandes en cours
	011	3 demandes en cours
	100	4 demandes en cours
	101	5 demandes en cours
	110	5 demandes en cours
	111	7 demandes en cours et plus

6.5.2.2.4.2 Multiplexed slot (MS)

Ce paramètre est mis en place par la DLPDU PReq correspondante lorsque le CN adressé est pris en charge dans un intervalle de temps multiplexé. Le paramètre est signalé à l'application au sein de la primitive DL-PDO de DLS.

En se basant sur ces informations, d'autres CN peuvent identifier que le CN expéditeur est servi par un intervalle multiplexé.

6.5.2.2.4.3 Exception new (EN)

Ce paramètre est basculé par le CN pour indiquer un signal d'exception au MN. Le paramètre est généré/évalué au sein de l'ES.

Pour les CN uniquement asynchrones, le paramètre est aussi signalé au sein de la DLSDU de DL-STA. Le CSM est chargé d'insérer ce paramètre dans la DL-STA et de le retirer.

6.5.2.2.4.4 Ready (RD)

Ce paramètre indique que les données de charge utile transférées sont valides. Le paramètre est transmis à/reçu de l'Utilisateur de DLS au sein de la primitive DL-PDO de DLS.

6.5.2.2.4.5 Priority (PR)

Ce paramètre indique la priorité de la (des) trame(s) dans la file d'attente d'envoi asynchrone avec la priorité la plus haute. La plage de cette valeur est 0 (priorité la plus basse) à 7 (priorité la plus haute). Deux de ces niveaux sont dédiés aux bus de terrain de Type 13:

- PRIO_NMT_REQUEST: Il s'agit de la priorité la plus haute qui doit être exclusivement appliquée si un CN demande la génération d'une commande NMT par le MN.
- PRIO_GENERIC_REQUEST: Priorité moyenne qui est le niveau de priorité normal pour les demandes de commande non NMT.

Les niveaux de priorité restants au-dessus et en-dessous de PRIO_GENERIC_REQUEST sont disponibles à des fins d'application.

Avant de transmettre la DLPDU PRes, ATC est requis pour le statut actuel des files d'attente d'envoi asynchrone.

6.5.2.2.4.6 RequestToSend (RS)

Ce paramètre indique le nombre de trames en attente dans la file d'attente d'envoi asynchrone avec la priorité la plus haute. La plage de cette valeur est 0 (aucune demande en attente) à 7 (7 demandes en attente et plus).

Avant de transmettre la DLPDU PRes, ATC est requis pour le statut actuel des files d'attente d'envoi asynchrone.

6.5.2.2.5 Payload (pl)

La DLPDU PReq peut transmettre des données utilisateur de longueur pouvant atteindre P(C_DLL_ISOCHR_MAX_PAYL). Le paramètre est transmis à/reçu de l'Utilisateur de DLS au sein de la primitive DL-PDO de DLS.

6.5.3 Envoi de la DLPDU PRes

Une DLPDU PRes doit être envoyée en tant que réponse à une DLPDU PReq reçue dans le cas d'un CN continu ou multiplexé, en tant que réponse à une DLPDU PRes reçue de la part du MN dans le cas d'un CN cadencé en continu sur une base cadencée ou par le MN.

La DLPDU est assemblée en demandant l'ITC pour la DLSDU expéditrice à tous les nœuds, en demandant l'ATC pour le statut actuel des files d'attente d'envoi asynchrone, en demandant l'ES pour le nouveau paramètre d'exception réel et en demandant le NS pour le "NMTStatus" réel des CN.

6.5.4 Réception de la DLPDU PRes

La DLPDU PRes est reçue par tous les CN isochrones ainsi que par le MN.

Les données reçues sont décompressées et transmises directement à l'ITC par le biais du service CSM-RPDO. Le paramètre EA est transmis à l'ES par le biais de CSM-ERR et le "NMTStatus" est transmis au NS pour un traitement ultérieur.

Le MN utilise un temps d'attente après l'envoi d'une réponse DLPDU PReq après avoir envoyé sa DLPDU PRes dans le cas de nœuds cadencés en continu pour la réception de la réponse DLPDU PRes correspondante pour détecter des erreurs de transmission et des défaillance de nœuds.

6.6 Début de l'asynchrone (SoA)

6.6.1 Généralités

La DLPDU SoA doit être utilisée pour

- identifier les CN,
- demander des informations relatives au statut d'un CN,
- envoyer des commandes NMT,
- échanger sur interrogation (poll) des CN uniquement asynchrones et
- accorder le droit de transmission asynchrone à un CN,
- synchroniser des CN cadencés en continu.

La DLPDU SoA est la première trame dans la phase asynchrone et constitue un signal informant tous les CN que toutes les données isochrones ont été échangées lors de la phase isochrone. Si aucune demande de transmission de message asynchrone n'est en attente, la DLPDU SoA ne possédant aucun droit d'envoyer un message est transmise.

La DLPDU SoA est prise en charge par le MN. Le MN connaît sa propre file d'attente d'envoi asynchrone et les files d'attente de tous les CN actifs, rapportés par le biais de la DLPDU PRes. L'ASS du MN doit déterminer le cycle dans lequel le droit d'envoyer la trame asynchrone sera accordé. Il doit garantir qu'aucune demande d'envoi ne sera effacée pendant une durée indéfinie, même si la charge du réseau est importante.

L'attribution de l'intervalle de temps asynchrone au MN lui-même doit être indiquée de la même façon que les attributions aux CN.

6.6.2 Structure de la DLPDU SoA

6.6.2.1 Généralités

La structure de la DLPDU SoA est présentée dans le Tableau 11.

Tableau 11 – Structure de la DLPDU SoA

Champ de la DLPDU	Type de données	Valeur/description
Message Type	OCTET[1]	05h/identification de la DLPDU SoA
Destination	OCTET[1]	ID de nœud du nœud adressé.
Source	OCTET[1]	ID de nœud du nœud émettant
NMTStatus	OCTET[1]	Statut du diagramme d'états NMT des MN
SoA-Flag	OCTET[1]	
reserved	OCTET[1]	
svid	OCTET[1]	ServiceID (svid) requis
svtg	OCTET[1]	Cible de service requis (svtg)
FieldbusVersion	OCTET[1]	Version de bus de terrain de Type 13 du MN
reserved	OCTET[1]	
Payload	OCTET[36]	DLSDU

6.6.2.2 Paramètres de la DLPDU SoA

6.6.2.2.1 Destination (dest)

Ce paramètre indique l'ID de nœud du nœud adressé (voir 5.3.3.2). Pour la DLPDU SoA, le paramètre "destination" est mis à 255 (C_ADDR_BROADCAST).

NOTE La cible de cette DLPDU est indiquée au sein du paramètre "RequestedServiceTarget" (voir 6.6.2.2.6). Le paramètre "destination" mis à 255 (C_ADDR_BROADCAST) est nécessaire à des fins de synchronisation du CSM.

6.6.2.2.2 Source (src)

Ce paramètre indique l'ID de nœud du nœud émettant (voir 15.4.8). Pour la DLPDU SoA, le paramètre est mis à 240 (C_ADDR_MN_DEF_NODE_ID).

6.6.2.2.3 NMTStatus (stat)

Ce paramètre rapporte le statut actuel du diagramme d'état NMT des MN et est inséré/évalué dans/à partir de l'Utilisateur de DLS avec DL-NMT de DLS.

6.6.2.2.4 SoA-Flag

6.6.2.2.4.1 Généralités

La structure de SoA-Flag est présentée dans le Tableau 12.

Tableau 12 – Structure de SoA-Flag

Numéro de bit	Valeur	Description
7-3		Réservé
2		Exception acknowledge (EA)
1		Exception reset (ER)
	0	Aucune demande
	1	Initialisation de la demande du système d'exception
0		Réservé

6.6.2.2.4.2 Exception acknowledge (EA)

Ce paramètre est basculé par le MN pour accuser réception d'un signal d'exception requis provenant du CN adressé. Le paramètre est généré/évalué au sein de l'ES. Le paramètre n'est valide que si "RequestedServiceID" est égal à "StatusRequest".

Pour des CN isochrones, le paramètre est aussi signalé au sein de la DLPDU PReq. Le CSM est chargé d'insérer ce paramètre dans la DL-STA et de le retirer.

6.6.2.2.4.3 Exception reset (ER)

Ce paramètre est basculé par le MN pour réinitialiser l'indicateur d'exception interne du CN adressé. Le paramètre est généré/évalué au sein de l'ES. Le paramètre n'est valide que si "RequestedServiceID" est égal à "StatusRequest".

6.6.2.2.5 RequestedServiceID (svid)

Ce paramètre indique l'ID de service asynchrone dédié à l'intervalle de temps asynchrone suivant.

Le Tableau 13 présente les définitions des entrées de "RequestServiceID".

Tableau 13 – Définition de "RequestedServiceID" dans la DLPDU SoA

Description	Valeur (ID)	Description
NoService	00_h NO_SERVICE	Doit être utilisé si l'intervalle de temps asynchrone n'est pas attribué à un nœud. "RequestedServiceTarget" doit être C_ADDR_INVALID
IdentRequest	01_h IDENT_REQUEST	Doit être utilisé pour identifier les CN inactifs et/ou pour demander les données d'identification d'un CN. Le CN adressé doit répondre immédiatement après avoir reçu le SoA avec la DLPDU ASnd IdentResponse spécifique au nœud
StatusRequest	02_h STATUS_REQUEST	Doit être utilisé pour demander le statut actuel et les informations détaillées relatives à un nœud. Les CN uniquement asynchrones doivent être demandés de manière cyclique par "StatusRequest" pour surveiller leur statut et effectuer leurs demandes d'intervalle de temps asynchrone. Le nœud adressé doit répondre immédiatement après avoir reçu le SoA avec la trame DLPDU ASnd StatusResponse spécifique au nœud
NMTRequestInvite	03_h NMT_REQUEST_INVITE	Doit être utilisé pour attribuer l'intervalle de temps asynchrone à un nœud qui a indiqué une NMTCommand/NMTRequest en attente. Le nœud adressé doit répondre immédiatement après avoir reçu le SoA avec la DLPDU ASnd NMTCommand/NMTRequest
SyncRequest	06_h SYNC_REQUEST	Doit être utilisé pour configurer/synchroniser les CN cadencés en continu et/ou pour demander les données de synchronisation d'un CN. Le CN adressé doit répondre immédiatement après avoir reçu le SoA avec la DLPDU ASnd SyncResponse spécifique au nœud
UnspecifiedInvite	FF_h UNSPECIFIED_INVITE	Doit être utilisé pour attribuer l'intervalle de temps asynchrone à un nœud qui a indiqué une demande de transmission en attente.

Description	Valeur (ID)	Description
		Le nœud adressé doit répondre immédiatement après avoir reçu le SoA, avec n'importe quel type de DLPDU ASnd de bus de terrain de Type 13 ou une DLPDU Ethernet traditionnel.
NOTE Tous les "ServiceID" sans "NoService" sont pris en charge au sein de l'ATC.		

L'ASS des MN décide au sein du statut des files d'attente d'envoi connu, quel nœud avec quel statut obtient le prochain intervalle de temps asynchrone.

6.6.2.2.6 RequestedServiceTarget (svtg)

Ce paramètre indique l'adresse (adresse de DL) du nœud, qui doit envoyer au sein de l'intervalle de temps asynchrone.

C_ADDR_INVALID doit indiquer que l'intervalle de temps asynchrone n'est pas attribué.

6.6.2.2.7 Type 13 fieldbus Version (eplv)

Ce paramètre indique la version actuelle du bus de terrain de Type 13 du MN.

6.6.2.3 Charge utile

6.6.2.3.1 Généralités

Ce n'est que dans "SyncRequest" que les données utilisateur sont transportées par la DLPDU SoA. Dans tous les autres cas, aucune donnée utilisateur n'est transportée par la DLPDU SoA.

6.6.2.3.2 SyncRequest

Le paramètre est transmis à/reçu de l'Utilisateur de DLS comme le paramètre DLSDU de DL-SYN de DLS.

6.6.3 Envoi de la DLPDU SoA

Une DLPDU SoA est envoyée par le MN au début de la phase asynchrone et attribue la phase asynchrone à un nœud spécifique, si nécessaire. L'ASS des MN décide de manière juste quel "ServiceID" de quel nœud doit être servi après.

La DLPDU est assemblé en demandant l'ASS pour les prochains "ServiceID" et "ServiceTarget". Les paramètres EA et ER sont regroupés à partir de l'ES. Le "NMTStatus" des MN est requis du NS.

6.6.4 Réception de la DLPDU SoA

Après réception d'une DLPDU PReq, le CSM attend la réception d'une DLPDU SoA. La réception d'une DLPDU SoA confirme la fin de la phase isochrone.

Le paramètre "RequestedServiceTarget" de la DLPDU SoA reçue est analysé. Selon le ServiceID demandé, la transmission d'une DLPDU ASnd doit survenir immédiatement après la transmission/réception d'une DLPDU SoA. L'ATC est requise pour le "ServiceID" demandé.

Les paramètres EA et EN sont transmis à l'ES par le biais de CSM-ERR et le "NMTStatus" est transmis au NS pour un traitement ultérieur.

Le MN doit observer le trafic du réseau afin de garantir qu'il n'y a pas d'autre MN actif sur le réseau. La réception d'une DLPDU SoC ou SoA indique qu'un autre MN est actif.

En raison de l'adresse MAC distincte de chaque nœud, il est impossible que deux nœuds ou plus possèdent la même adresse MAC mais il est toujours possible que deux nœuds ou plus possèdent la même adresse de bus de terrain de Type 13. Seuls les CN à plusieurs communications asynchrones peuvent répondre sur une DLPDU SoA. Étant donné que le MN envoie une Diffusion MAC (DLPDU SoA) à une adresse de bus de terrain de Type 13 à diffusion unique, plusieurs CN avec la même adresse de bus de terrain de Type 13 sont capables de répondre. Le MN doit détecter plusieurs adresses de bus de terrain de Type 13 utilisées dans un réseau en comptant les réponses sur une DLPDU SoA IdentRequest.

6.7 Envoi Asynchrone (ASnd)

6.7.1 Généralités

Il existe deux types de DLPDU asynchrones disponibles:

- une DLPDU qui n'est pas de bus de terrain de Type 13 peut être envoyée,
- Une DLPDU de bus de terrain de Type 13 de la forme d'une DLPDU ASnd.

Bien qu'une seule DLPDU asynchrone par cycle de bus de terrain de Type 13 soit autorisée, le CSM du CN ne limite pas la quantité de DLPDU reçues au sein de la phase asynchrone du cycle.

Après attribution de l'intervalle de temps asynchrone par le biais de la DLPDU SoA, le nœud requis arrange les informations requises et les envoient par le biais d'une DLPDU qui n'est pas de bus de terrain de Type 13 ou par le biais d'une DLPDU ASnd. Différents ServiceID dans une DLPDU ASnd sont possibles.

- StatusResponse: est utilisé pour diffuser le statut actuel et les informations détaillées relatives aux erreurs d'un nœud.
- IdentResponse: est utilisé pour diffuser les données d'identification des CN inactifs en attente d'être inclus dans le réseau et/ou pour demander les données d'identification d'un CN.
- NMTCommand: est utilisé par le NMT du MN pour diffuser les informations communes aux NMT et pour forcer des commandes spécifiques à un, à des nœuds spécifiques ou à tous les nœuds sur le réseau de bus de terrain de Type 13.
- NMTRequest: est utilisé pour offrir la possibilité au CN également de demander les fonctions ci-dessus. La fonction requise est décodée dans la charge utile et diffusée par le MN dans l'un des prochains intervalles de temps asynchrones par le biais de la DLPDU ASnd.
- SDO: est utilisé pour la communication entre pairs avec accès au dictionnaire d'objets d'un appareil.
- SyncResponse: est utilisé pour diffuser les données de synchronisation actuelles d'un CN cadencé en continu.

Le nœud adressé doit répondre immédiatement après réception du SoA.

6.7.2 Structure de la DLPDU ASnd

6.7.2.1 Généralités

La structure de la DLPDU ASnd est présentée dans le Tableau 14.

Tableau 14 – Structure de la DLPDU ASnd

Champ de la DLPDU	Type de données	Valeur/description
Message Type	OCTET[1]	06h/identification de la DLPDU ASnd
Destination	OCTET[1]	ID de nœud du nœud adressé.
Source	OCTET[1]	ID de nœud du nœud émettant
ServiceID	OCTET[1]	ID de service
Payload	OCTET[1-P(C_DLL_MAX_PAYL_OFFSET)-4]	DLSDU

6.7.2.2 Paramètres de la DLPDU ASnd

6.7.2.2.1 Destination (dest)

Ce paramètre indique l'ID de nœud du nœud adressé (voir 5.3.3.2). Pour la DLPDU ASnd, la destination dépend du "ServiceID" utilisé (voir 6.7.2.2.3).

6.7.2.2.2 Source (src)

Ce paramètre indique l'ID de nœud du nœud émettant (voir 5.3.3.3).

6.7.2.2.3 ServiceID (svid)

Ce paramètre indique le "ServiceID" dédié à l'intervalle de temps asynchrone.

Le Tableau 15 présente les définitions des entrées de "ServiceID".

Tableau 15 – Définition de "ServiceID" dans la DLPDU ASnd

Description	Valeur (ID)	Destination	Description
IdentResponse	01_h (IDENT_RESPONSE)	Diffusion de bus de terrain de Type 13	Au sein de ServiceID IdentResponse, le CN diffuse les données d'identification. Le "ServiceID" doit être généré par un nœud ayant reçu une "IdentRequest" par le biais de la SoA
StatusResponse	02_h (STATUS_RESPONSE)	Diffusion de bus de terrain de Type 13	Au sein de ServiceID StatusResponse le CN diffuse le statut actuel et les informations détaillées relatives aux erreurs. Le "ServiceID" doit être généré par un nœud ayant reçu une "StatusRequest" par le biais de la SoA
NMTRequest	03_h (NMT_REQUEST)	MN de bus de terrain de Type 13	Au sein de ServiceID NMTRequest, le CN demande au MN de demander le "ServiceID" spécifié au sein de la prochaine phase asynchrone. Le "ServiceID" doit être généré par un CN ayant reçu une "NMTRequestInvite" par le biais de la SoA
NMTCommand	04_h (NMT_COMMAND)	(voir Note 2)	Au sein de ServiceID NMTCommand, le NMT du MN diffuse les informations communes aux NMT et force des commandes spécifiques à un, à des nœuds spécifiques ou à tous les nœuds sur le réseau de bus de terrain de Type 13. Ce "ServiceID" doit être généré par le MN sur demande interne ou externe par le biais de "NMTRequest"
SDO	05_h (SDO)	CN de bus de terrain de	Au sein du ServiceID SDO, une communication entre pairs avec accès au dictionnaire d'objets

Description	Valeur (ID)	Destination	Description
		Type 13 et MN de bus de terrain de Type 13	d'un appareil est générée. Ce "ServiceID" peut être généré par un nœud ayant reçu une "UnspecifiedInvite" par le biais de la SoA
SyncResponse	06_h (SYNC_RESPONSE)	Diffusion de bus de terrain de Type 13	Au sein de ServiceID SyncResponse, le CN diffuse les données de synchronisation actuelles. Ce "ServiceID" doit être généré par un nœud ayant reçu une "SyncRequest" par le biais de la SoA
Manufacturer specific	$A0_h$ à FE_h	CN de bus de terrain de Type 13 et MN de bus de terrain de Type 13	Au sein d'un ID de Service spécifique au fabricant, l'Utilisateur de DLS peut diffuser toute sorte de données. L'utilisation de ces ID de service relève du domaine d'application du présent document. Ce "ServiceID" peut être généré par un nœud ayant reçu une "UnspecifiedInvite" par le biais de la SoA.

NOTE 1 Les "ServiceID" non énumérés sont réservés.

NOTE 2 La destination dépend de la commande NMT sous-jacente. Les CN de bus de terrain de Type 13 ainsi que la diffusion de bus de terrain de Type 13 sont possibles.

6.7.2.2.4 Charge Utile (pl)

6.7.2.2.4.1 Généralité

La DLPDU ASnd peut transmettre des données utilisateur de longueur pouvant atteindre P(C_DLL_ISOCHR_MAX_PAYL).

Les données utilisateur suivantes sont définies pour le paramètre "payload" de la DLPDU ASnd:

6.7.2.2.4.2 IdentResponse

Le paramètre est transmis à/reçu de l'Utilisateur de DLS comme le paramètre DLSDU de DL-IDE de DLS.

6.7.2.2.4.3 StatusResponse

Le paramètre est transmis à/reçu de l'Utilisateur de DLS comme le paramètre DLSDU de DL-STA de DLS.

6.7.2.2.4.4 NMTCommand

Le paramètre est transmis à/reçu de l'Utilisateur de DLS comme le paramètre DLSDU de DL-CMD de DLS.

6.7.2.2.4.5 NMTRequest

Les CN demandant une "NMTCommand", une "IdentResponse" et une "StatusResponse" prennent en charge cette demande en invitant le MN avec une "NMTRequest" pour exécuter le service demandé.

Comme présenté dans le Tableau 16, les données utilisateur de "NMTRequest" doivent contenir "NMTRequestedCommandID", "NMTRequestedCommandTarget" et "NMTRequestedData".

Tableau 16 – Structure des données utilisateur de NMTRequest

Champ de la DPDLU	Type de données	Valeur/description
NMTRequestCommandID	OCTET[1]	
NMTRequestedCommandTarget	OCTET[1]	
NMTRequestedCommandData	OCTET[1 à P(C_DLL_MAX_PAYL_OFFSET)-4]	

où:

- NMTRequestedCommandID (rcid): Le service NMT doit être généré par le MN par sa valeur “NMTCommandID”. Les services “StatusResponse” et “IdentResponse” sont indiqués par les valeurs “STATUS_REQUEST” et “IDENT_REQUEST” de “RequestedServiceID” de la SoA.
- NMTRequestedCommandTarget (rct): Indique le nœud cible de la commande NMT demandée.
- NMTRequestedCommandData (rcd): données spécifiques à la commande NMT à générer par le MN.

6.7.2.2.4.6 SDO

Le paramètre est transmis à/reçu de l’Utilisateur de DLS comme le paramètre DLSDU de DL-SDO de DLS.

6.7.2.2.4.7 SyncResponse

Le paramètre est transmis à/reçu de l’Utilisateur de DLS comme le paramètre DLSDU de DL-SYN de DLS.

6.7.2.2.4.8 Manufacturer specific

Le paramètre est transmis à/reçu de l’Utilisateur de DLS comme le paramètre DLSDU de DL-UDT de DLS.

6.7.3 Envoi de la DLPDU ASnd

Après avoir été invité par une DLPDU SoA, le CN requis envoie une DLPDU ASnd, avec ServiceID mis à:

- IdentResponse: Au sein de l’ATC, les données utilisateur sont requises avec la DL-IDE de DLS. Les données utilisateur répondues sont assemblées avec le ServiceID=IdentResponse à la DLPDU ASnd.
- StatusResponse: Au sein de l’ATC, les données utilisateur sont demandées avec la DL-STA de DLS. Les données utilisateur répondues sont assemblées avec le ServiceID=StatusResponse à la DLPDU ASnd.
- NMTCommand: Les NMTCommands requises par l’Utilisateur de DLS sont mises en file d'attente dans l'ATC. Au sein de l’ATC, le contenu de la file d’attente la plus ancienne est répondu et ces données utilisateur sont assemblées avec le ServiceID=NMTCommand à la DLPDU ASnd.
- NMTRequest: Les NMTCommands requises, StatusRequest et IdentRequests des CN sont mises en file d’attente dans l’ATC. Au sein de l’ATC, le contenu de la file d'attente la plus ancienne est traduit dans des données utilisateur NMTRequest et ces dernières sont assemblées avec ServiceID=NMTRequest à la DLPDU à exécuter au MN.
- SDO: Les DLSDU SDO requises par l’Utilisateur de DLS sont mises en file d'attente dans l'ATC. Au sein de l’ATC, le contenu de la file d’attente la plus ancienne est répondu et ces données utilisateur sont assemblées avec le ServiceID=SDO à la DLPDU ASnd.

- SyncResponse: Au sein de l'ATC, les données utilisateur sont requises avec la DL-SYN de DLS. Les données utilisateur répondues sont assemblées avec le ServiceID=SyncResponse à la DLPDU ASnd.
- Spécifique au fabricant: Les DLSDU UDT requises par l'Utilisateur de DLS sont mises en file d'attente dans l'ATC. Au sein de l'ATC, ces données utilisateurs sont assemblées avec ServiceID="Manufacturer specific" à la DLPDU ASnd.

6.7.4 Réception de la DLPDU ASnd

Si une trame ASnd a été reçue, elle doit être traitée selon le "ServiceID" comme suit:

- IdentResponse: Les DLPDU ASnd IdentResponse reçues sont transmises à l'utilisateur de DLS par le biais de la primitive DL-IDE de l'ATC. Le MN doit recevoir l'"IdentResponse". Les CN peuvent recevoir "IdentResponse" s'ils sont configurés comme tels.
- StatusResponse: Les DLPDU ASnd StatusResponse reçues sont transmises à l'utilisateur de DLS par le biais de la primitive DL-STA de l'ATC. Le MN doit recevoir la "StatusResponse". Les CN peuvent recevoir "StatusResponse" s'ils sont configurés comme tels.
- NMTCommand: Les DLPDU ASnd NMTCommand reçues sont transmises à l'utilisateur de DLS par le biais de la primitive DL-CMD de l'ATC.
- NMTRequest: Les DLPDU ASnd NMTRequest reçues sont interprétées par l'ATC des MN et mises dans la file d'attente de "Status", "Ident" ou "NMTCommand", selon la "NMTRequestedCommandID" en demande. Les CN ne reçoivent pas cette DLPDU.
- SDO: Les DLPDU ASnd SDO reçues sont transmises à l'utilisateur de DLS par le biais de la primitive DL-SDO de l'ATC.
- SyncResponse: Les DLPDU ASnd SyncResponse reçues sont transmises à l'utilisateur de DLS par le biais de la primitive DL-SYN de l'ATC. Le MN doit recevoir la "SyncResponse". Les CN doivent recevoir "SyncResponse" s'ils prennent en charge la communication cadencée en continu.
- Manufacture specific: Les DLPDU ASnd UDT reçues sont transmises à l'utilisateur de DLS par le biais de la primitive DL-UDT de l'ATC.

7 Éléments de procédure de la DLE

7.1 Structure générale

La DLL se compose d'élément de commande pour la commande TX/RX en transmission isochrone (ITC), la commande TX/RX en transmission asynchrone (ATC), de diagramme d'états de cycle (CSM), de programmateur d'intervalle de temps asynchrone (ASS), de signalisation d'exception (ES), de signalisation de NMT (NS) et d'interface de gestion de la DLL.

Le CSM en tant qu'élément de commande principal fournit la fonction pour la commande de l'accès au support déterministe coopérant avec l'ITC, l'ATC, l'ES et le NS pour une prise en charge fiable et efficace des services de transfert de données asynchrones et isochrones.

L'interface de gestion de la DLL fournit des fonctions de gestion de la DLL.

7.2 Diagramme d'États de Cycle (Cycle State Machine) (CSM)

7.2.1 Vue d'ensemble

7.2.1.1 Diagramme d'états de cycle du MN (CSM-MN)

Le diagramme d'états de cycle du MN (CSM-MN) doit gérer la communication au sein d'un cycle de bus de terrain de Type 13. Le CSM-MN génère le flux de trames lors d'un cycle de

bus de terrain de Type 13 et surveille la réaction des CN. L'ordre des flux dépend de l'état du NMT (signalé par la primitive CSM-TNMT).

7.2.1.2 Diagramme d'états de cycle du CN (CSM-CN)

Le diagramme d'états de cycle du CN (CSM-CN) gère la communication au sein d'un Cycle de bus de terrain de Type 13. Le CSM-CN surveille l'ordre des trames reçues au sein d'un cycle et réagit comme suit. L'ordre de réception des trames prévu dépend de l'état du NMT (signalé par la primitive CSM-TNMT).

7.2.2 Définitions des primitives

7.2.2.1 Définitions des primitives entre CSM et ITC

Le Tableau 17 récapitule toutes les primitives échangées entre le CSM et l'ITC.

Tableau 17 – Primitives échangées entre CSM et ITC

Nom de primitive	Source	Paramètres associés	Description
CSM-PDO.ind	ITC	Isochr Isochr-time-triggered Isochr-Out	Indique l'attribution du prochain intervalle de temps isochrone.
CSM-TPDO.ind	ITC	D_addr DLSDU	Indication d'un élément PDO avec ses paramètres associés pour la transmission au prochain intervalle de temps isochrone. MN: L'élément PDO requis est déterminé par le compteur d'intervalle de temps interne de l'ITC. CN: Seul le PDO dépendant du CN est requis une fois par cycle de bus de terrain de Type 13.
CSM-RPDO.ind	CSM	S_addr D_addr DLSDU	Reporte un élément PDO reçu avec ses paramètres associés à l'ITC.

Les paramètres utilisés pour l'échange de primitives entre le CSM et l'ITC sont décrits dans le Tableau 18.

Tableau 18 – Paramètres utilisés avec les primitives échangées entre le CSM et l'ITC

Nom de paramètre	Description
S_addr	Le paramètre "S_addr" spécifie l'adresse de DL de l'éditeur.
D_addr	Le paramètre "D_addr" spécifie l'adresse de DL de l'abonné.
DLSDU	Ce paramètre spécifie l'information qui est transférée par le biais d'un tampon à partir de la DLE locale en tant qu'éditeur vers les DLE homologues multiples distantes en tant qu'abonnées.
Isochr	Ce paramètre spécifie avec une valeur différente de zéro qu'il existe des nœuds dans la liste isochrone du cycle actuel.
Isochr-time-triggered	Ce paramètre spécifie avec une valeur différente de zéro qu'il existe des nœuds dans la liste isochrone du cycle actuel qui appartiennent à la classe de communication "*continuous-time-triggered" (cadencé en continu)*.
Isochr-Out	Ce paramètre spécifie avec une valeur différente de zéro que le MN est configuré pour envoyer une PRes.

7.2.2.2 Définitions des primitives entre CSM et ATC

Le Tableau 19 récapitule toutes les primitives échangées entre le CSM et l'ATC.

Tableau 19 – Primitives échangées entre CSM et ATC

Nom de primitive	Source	Paramètres associés	Description
CSM-TASND.ind	ATC	Destination ServiceID Payload	Indique une DLPDU ASnd avec ses paramètres associés provenant de la file d'attente interne de l'ATC pour la transmission au prochain intervalle de temps asynchrone.
CSM-RASND.ind	CSM	Destination Source ServiceID Payload	Reporte une DLPDU ASnd reçue avec ses paramètres associés à l'ATC.
ATC-Prio.req	CSM	(aucun)	Demande la priorité et le nombre de trames en attente dans la file d'attente d'envoi asynchrone avec la priorité la plus haute de l'ATC. CN: Cette primitive est appelée juste avant l'envoi de la DLPDU PRes et la DLPDU ASnd StatusResponse. MN: Cette primitive est appelée juste avant l'envoi de la DLPDU SoA.
ATC-Prio.ind	ATC	Priority RequestToSend	Indique la priorité et le nombre de trames en attente dans la file d'attente d'envoi asynchrone avec la priorité la plus haute. CN: Les paramètres sont transmis par le biais de la PRes et de la DLPDU ASnd StatusResponse. Le CSM des MN reporte ce paramètre à l'ASS par le biais de CSM-Prio. MN: Le CSM interne des MN reporte ce paramètre à l'ASS par le biais de CSM-Prio.

Les paramètres utilisés pour l'échange de primitives entre le CSM et l'ATC sont décrits dans le Tableau 20.

Tableau 20 – Paramètres utilisés avec les primitives échangées entre CSM et ATC

Nom de paramètre	Description
Destination	Ce paramètre indique l'ID de nœud du nœud adressé.
Source	Ce paramètre indique l'ID de nœud du nœud émettant.
ServiceID	Ce paramètre indique le "ServiceID" dédié à l'intervalle de temps asynchrone.
Payload	Ce paramètre spécifie les données utilisateur.
Priority	Ce paramètre indique la priorité des données utilisateur dans la file d'attente d'envoi asynchrone avec la priorité la plus haute.
RequestToSend	Ce paramètre indique le nombre de données utilisateur en attente dans la file d'attente d'envoi asynchrone avec la priorité la plus haute.

7.2.2.3 Définitions des primitives entre CSM et ASS

Le Tableau 21 récapitule toutes les primitives échangées entre le CSM et l'ASS.

Tableau 21 – Primitives échangées entre CSM et ASS

Nom de primitive	Source	Paramètres associés	Description
CSM-CSM.ind	CSM	Resp-Expected Async-In Asycn-Out	Indique l'attribution du prochain intervalle de temps isochrone.
CSM-Prio.ind	CSM	Source Priority RequestToSend	Indique la priorité reçue et le paramètre "Request to Send" de la source spécifiée
CSM-SoA.req	CSM	(aucun)	Demande le service et le nœud, qui doivent transmettre la trame Asnd dans le prochain intervalle de temps asynchrone
CSM-SoA.ind	ASS	Requested-ServiceID Requested-ServiceTarget	Indique le type de données utilisateur avec le paramètre "ServiceID" et le nœud adressé avec le paramètre "ServiceTarged", qui doit envoyer sur le prochain intervalle de temps asynchrone. Les paramètres sont transmis par le biais de la DLPDU SoA en attente

Les paramètres utilisés pour l'échange de primitives entre le CSM et l'ASS sont décrits dans le Tableau 22.

Tableau 22 – Paramètres utilisés avec les primitives échangées entre CSM et ASS

Nom de paramètre	Description
RequestedServiceID	Ce paramètre indique l'ID de service asynchrone dédié à l'intervalle de temps asynchrone suivant.
RequestedServiceTarget	Ce paramètre indique l'ID de Nœud du nœud, qui doit envoyer au sein de l'intervalle de temps asynchrone suivant
Resp-Expected	Resp-expected = "TRUE" signifie qu'une DLPDU ASnd est attendue au prochain intervalle de temps asynchrone
Async-In	Cela indique qu'une DLPDU SoA doit être envoyée dans ce cycle et qu'une DLPDU ASnd ou une trame de bus de terrain autre que de Type 13 pourrait être reçue
Async-Out	Cela indique qu'une DLPDU SoA doit être envoyée dans ce cycle et qu'une DLPDU ASnd doit être envoyée dans ce cycle
Priority	Ce paramètre indique la priorité des données utilisateur dans la file d'attente d'envoi asynchrone avec la priorité la plus haute.
RequestToSend	Ce paramètre indique le nombre de données utilisateur en attente dans la file d'attente d'envoi asynchrone avec la priorité la plus haute

7.2.2.4 Définitions des primitives entre CSM et ES

Le Tableau 23 récapitule toutes les primitives échangées entre le CSM et l'ES.

Tableau 23 – Primitives échangées entre CSM et ES

Nom de primitive	Source	Paramètres associés	Description
CSM-TERR.ind	ES	Destination ER EC EN EA	Indication des indicateurs d'exception. MN: Seuls les indicateurs ER et EA sont transmis au CSM. CN: Seuls les indicateurs EC et EN sont transmis au CSM.
CSM-RERR.ind	CSM	Source ER (SoA) EC (Status) EN (PRes/Stat) EA (PReq/Stat)	Reporte les indicateurs d'exception reçus à l'ES à chaque fois qu'un nouvel indicateur est reçu. MN: Seuls les indicateurs EC et EN sont transmis à l'ES. CN: Seuls les indicateurs ER et EA sont transmis à l'ES

Les paramètres utilisés pour l'échange de primitives entre le CSM et l'ES sont décrits dans le Tableau 24.

Tableau 24 – Paramètres utilisés avec les primitives échangées entre CSM et ES

Nom de paramètre	Description
Source	Ce paramètre indique l'ID de nœud du nœud émettant.
Destination	Ce paramètre indique l'ID de nœud du nœud dédié.
ER	Réinitialisation d'exception: Initialisation de la signalisation d'exception Ce paramètre est mis en place par le MN pour réinitialiser l'indicateur d'exception interne du CN adressé
EC	Effacement d'exception: accusé de réception de l'initialisation de la signalisation d'exception. Ce paramètre représente le dernier indicateur ER reçu du MN pour indiquer au MN que l'initialisation de la signalisation d'exception a été effectuée
EN	Nouvelle exception: signale une nouvelle exception Ce paramètre est basculé par le CN pour indiquer une exception au MN
EA	Accusé de réception d'exception: Ce paramètre est basculé par le MN pour accuser réception d'un signal d'exception requis à partir du CN adressé

7.2.2.5 Définitions des primitives entre CSM et NS

Le Tableau 25 récapitule toutes les primitives échangées entre le CSM et le NS.

Tableau 25 – Primitives échangées entre CSM et NS

Nom de primitive	Source	Paramètres associés	Description
CSM-TNMT.ind	NS	NMTStatus	Indication du propre paramètre "NMTStatus".
CSM-RNMT.ind	CSM	Source NMTStatus	Reporte le paramètre "NMTStatus" reçu au NS à chaque fois qu'un nouveau "NMTstatus" est reçu. MN: Le paramètre "NMTStatus" des MN est transmis par le biais de la DLPDU SoA. CN: Le paramètre "NMTStatus" des CN est transmis par le biais de la DLPDU PRes, DLPDU ASnd StatusResponse et de la DLPDU ASnd IdentResponse

Les paramètres utilisés pour l'échange de primitives entre le CSM et le NS sont décrits dans le Tableau 26.

Tableau 26 – Paramètres utilisés avec les primitives échangés entre CSM et NS

Nom de paramètre	Description
Source	Ce paramètre indique l'ID de nœud du nœud émettant
NMTStatus	Statut actuel du diagramme d'états de NMT

7.2.2.6 Définitions des primitives entre CSM et DLM

Le Tableau 27 récapitule toutes les primitives échangées entre le CSM et le DLM.

Tableau 27 – Primitives échangées entre CSM et DLM

Nom de primitive	Source	Paramètres associés	Description
CSM-Reset.ind	DLM	(aucun)	Indique une réinitialisation du CSM de la part de l'utilisateur de DLS
CSM-Frame.ind	CSM	DLM-frame-identifier MC PS Time Reltime	Indique à l'utilisateur de DLS le type de DLPDU réel reçu
CSM-Event.ind	CSM	DLM-event-identifier EntryType TimeStamp Additional-Information	Indique le nouveau statut d'erreur à l'utilisateur de DLS

Les paramètres utilisés pour l'échange de primitives entre le CSM et le DLM sont décrits dans le Tableau 28.

Tableau 28 – Paramètres utilisés avec les primitives échangées entre CSM et DLM

Nom de paramètre	Description
DLM-frame-identifier	Ce paramètre spécifie la primitive au sein de la DLE dont l'occurrence est annoncée. Les valeurs possibles sont définies dans la partie correspondante de la présente partie de la CEI 61158.
MC	Ce paramètre est basculé par le MN dès que le cycle multiplexé final a été terminé.
PS	Ce paramètre est basculé par le MN chaque énième cycle.
Time	Ce paramètre est distribué par le MN et indique l'heure de début du cycle de bus de terrain de Type 13
Reltime	Ce paramètre est distribué par le MN et indique le temps relatif, qui est incrémenté du temps du cycle de Type 13 lorsqu'une DLPDU SoC est générée
DLM-event-identifier	Description détaillée de l'erreur survenue
EntryType	Informations de mode et de profil concernant l'erreur survenue
TimeStamp	"NetTime" à partir du cycle de Type 13 lorsque l'erreur/l'événement a été détecté(e).
AdditionalInformation	Ce champ contient des informations supplémentaires relatives aux erreurs et spécifiques à un vendeur d'appareils ou à un profil d'appareil

7.2.3 Table des états de CSM

7.2.3.1 Table des états de CSM au MN

7.2.3.1.1 Vue d'ensemble

Le diagramme d'états de cycle du MN doit gérer la communication au sein d'un cycle de bus de terrain de Type 13.

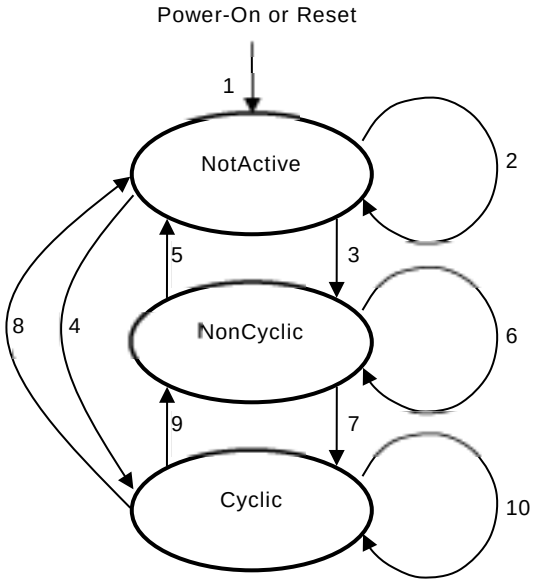
Le CSM des MN génère le flux de trames lors d'un cycle de bus de terrain de Type 13 et surveille la réaction des CN. L'ordre des flux dépend du paramètre "NMTStatus" des nœuds.

Généralement, les CN sont synchronisés par la réception de la DLPDU SoC. Cela signifie que le paramètre le plus important pour la synchronisation du réseau de bus de terrain de Type 13 est la précision temporelle de la fonction ME_SOC_TRIG.

Si une erreur dans la communication est détectée par le CSM, une primitive d'erreur sera générée.

7.2.3.1.2 Table d'états générale

Les transitions de CSM pourraient être représentées au sein d'un seul diagramme où le paramètre "NMTStatus", reçu par le biais de la primitive DL-NMT, constitue les conditions de transitions. Par souci de compréhension et de clarté, les transitions pertinentes des valeurs uniques de "NMTStatus" sont filtrées et affichées au sein d'un diagramme séparé en tant que "mode de fonctionnement" du CSM. Le diagramme états-transitions de la Figure 7 présente la réaction sur "NMTStatus" et l'activation des diagrammes d'états sous-jacents.



Légende

Anglais	Français
Power-On or Reset	Mise sous tension ou Réinitialisation
NotActive	NotActive (Inactif)
NonCyclic	NonCyclic (Non cyclique)
Cyclic	Cyclic (Cyclique)

Figure 7 – Diagramme états-transitions de CSM des MN

L'état "NotActive" signifie qu'aucun diagramme d'états sous-jacent n'est actif et qu'aucune réaction sur les trames reçues ne survient.

L'état "NonCyclic" signifie que le CSM des MN génère le Cycle de bus de terrain de Type 13 réduit et observe le comportement des CN. Le CSM des MN est en mode DLL_MS_NON_CYCLIC.

L'état "Cyclic" signifie que le CSM des MN génère le Cycle de bus de terrain de Type 13 réduit et observe le comportement des CN.

Les transitions sont décrites dans le Tableau 29.

Tableau 29 – Transitions du CSM des MN

#	État courant	Événement /condition ⇒actions	État suivant
1	Tout état	POWER-ON ou RESET =>	NotActive
2	NotActive	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_GS_INITIALISATION || NMTStatus = NMT_MS_NOT_ACTIVE || NMTStatus = NMT_MS_BASIC_ETHERNET => -- Désactiver le diagramme d'états CSM_MS_NON_CYCLIC -- Désactiver le diagramme d'états CSM_MS_CYCLIC	NotActive
3	Not Active	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_MS_PRE_OPERATIONAL_1 => -- Activer le diagramme d'états CSM_MS_NON_CYCLIC -- Désactiver le diagramme d'étatsCSM_MS_CYCLIC	NonCyclic
4	Not Active	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_MS_OPERATIONAL || NMTStatus =NMT_MS_READY_TO_OPERATE || NMTStatus =NMT_MS_PRE_OPERATIONAL_2 => -- Désactiver le diagramme d'états CSM_MS_NON_CYCLIC -- Activer le diagramme d'états CSM_MS_CYCLIC	Cyclic
5	NonCyclic	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_GS_INITIALISATION || NMTStatus = NMT_MS_NOT_ACTIVE || NMTStatus = NMT_MS_BASIC_ETHERNET => -- Désactiver le diagramme d'états CSM_MS_NON_CYCLIC -- Désactiver le diagramme d'états CSM_MS_CYCLIC	NotActive
6	NonCyclic	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_MS_PRE_OPERATIONAL_1 => -- Activer le diagramme d'états CSM_MS_NON_CYCLIC -- Désactiver le diagramme d'états CSM_MS_CYCLIC	NonCyclic
7	NonCyclic	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_MS_OPERATIONAL || NMTStatus =NMT_MS_READY_TO_OPERATE || NMTStatus =NMT_MS_PRE_OPERATIONAL_2 => -- Désactiver le diagramme d'états CSM_MS_NON_CYCLIC -- Activer le diagramme d'états CSM_MS_CYCLIC	Cyclic
8	Cyclic	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_GS_INITIALISATION ||	NotActive

#	État courant	Événement /condition ⇒actions	État suivant
		NMTStatus = NMT_MS_NOT_ACTIVE || NMTStatus = NMT_MS_BASIC_ETHERNET => -- Désactiver le diagramme d'états CSM_MS_NON_CYCLIC -- Désactiver le diagramme d'états CSM_MS_CYCLIC	
9	Cyclic	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_MS_PRE_OPERATIONAL_1 => -- Activer le diagramme d'états CSM_MS_NON_CYCLIC -- Désactiver le diagramme d'états CSM_MS_CYCLIC	NonCyclic
10	Cyclic	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_MS_OPERATIONAL || NMTStatus =NMT_MS_READY_TO_OPERATE || NMTStatus =NMT_MS_PRE_OPERATIONAL_2 => -- Désactiver le diagramme d'états CSM_MS_NON_CYCLIC -- Activer le diagramme d'états CSM_MS_CYCLIC	Cyclic

7.2.3.1.3 CSM_MS_NON_CYLIC

Le diagramme états-transitions est montré à la Figure 8.

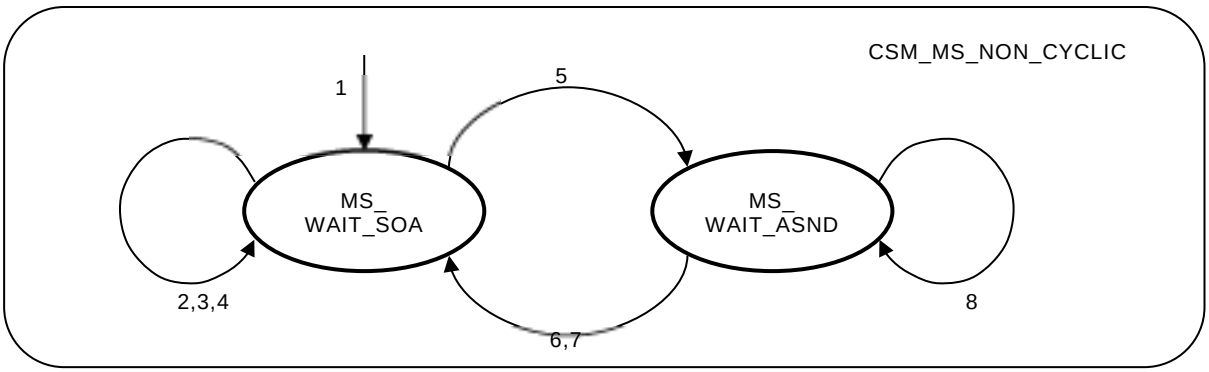


Figure 8 – Diagramme états-transitions du CSM des MN à CSM_MS_NON_CYCLIC

L'état "NON_CYCLIC" signifie que la communication cyclique n'a pas encore commencé ou a été arrêtée par le paramètre "NMTStatus" (NMTStatus = NMT_MS_PRE_OPERATIONAL_1). Le diagramme d'états attend de cette façon jusqu'à ce que le "NMTStatus" soit modifié en NMT_MS_PRE_OPERATIONAL_2. Il dépend du paramètre actuel "NMTStatus", dont les événements seront traités et ignorés.

Le CSM reste dans l'état "MS_WAIT_ASND" jusqu'à ce que la trame Ethernet soit reçue, que le temps d'attente de la phase asynchrone soit écoulé ou que le prochain cycle réduit commence avec le ME_SOA_TRIG() = "TRUE".

Les transitions sont décrites dans le Tableau 30.

Tableau 30 – Transitions du CSM des MN à CSM_MS_NON_CYCLIC

#	État courant	Événement /condition ⇒actions	État suivant
1	Tout état	Activation / =>	MS_WAIT_SOA
2	MS_WAIT_SOA	CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \ ME_SOA_TRIG ( ) = ”TRUE“ && Async-In <> 0 && Resp-Expected = “FALSE” => SOADU:= BUILD_SOA ( ) MA_Data.req { SOADU }	MS_WAIT_SOA
3	MS_WAIT_SOA	CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \ ME_SOA_TRIG ( ) = ”TRUE“ && Async-In = 0 && Async-Out <> 0 => SOADU:= BUILD_SOA ( ) MA_Data.req { SOADU } ASNDDU:= BUILD_ASND (C_ADDR_MN_DEF_NODE_ID) MA_Data.req { ASNDDU }	MS_WAIT_SOA
4	MS_WAIT_SOA	CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \ ME_SOA_TRIG ( ) = ”TRUE“ && Async-In = 0 && Asynch-Out = 0 => SOADU:= BUILD_SOA ( ) MA_Data.req { SOADU }	MS_WAIT_SOA
5	MS_WAIT_SOA	CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \ ME_SOA_TRIG ( ) = ”TRUE“ && Async-In <> 0 && Resp-Expected = “TRUE” => SOADU:= BUILD_SOA ( ) MA_Data.req { SOADU } START_TIMER(T(ASND_TIME), V(AsyncSlotTimeout_U32))	MS_WAIT_ASND
6	MS_WAIT_ASND	CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \ (ME_SOA_TRIG () = “TRUE” || EXPIRED_TIMER (T(ASND_TIME)) = "True") && Async-In = 0 => SOADU:= BUILD_SOA ( ) MA_Data.req { SOADU } ASNDDU:= BUILD_ASND (C_ADDR_MN_DEF_NODE_ID) MA_Data.req { ASNDDU }	MS_WAIT_SOA
7	MS_WAIT_ASND	CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \ (ME_SOA_TRIG () = “TRUE” || EXPIRED_TIMER (T(ASND_TIME)) = "True") && Async-In <> 0 && Resp-Expected = “FALSE” => SOADU:= BUILD_SOA ( ) MA_Data.req { SOADU }	MS_WAIT_SOA
8	MS_WAIT_ASND	CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \ ME_SOA_TRIG () = “TRUE” || EXPIRED_TIMER (T(ASND_TIME)) =”TRUE) Async-In <> 0 && Resp-Expected = “TRUE” => SOADU:= BUILD_SOA ( ) MA_Data.req { SOADU }	MS_WAIT_ASND

7.2.3.1.4 CSM_MS_CYCLIC

Le diagramme états-transitions est montré à la Figure 9.

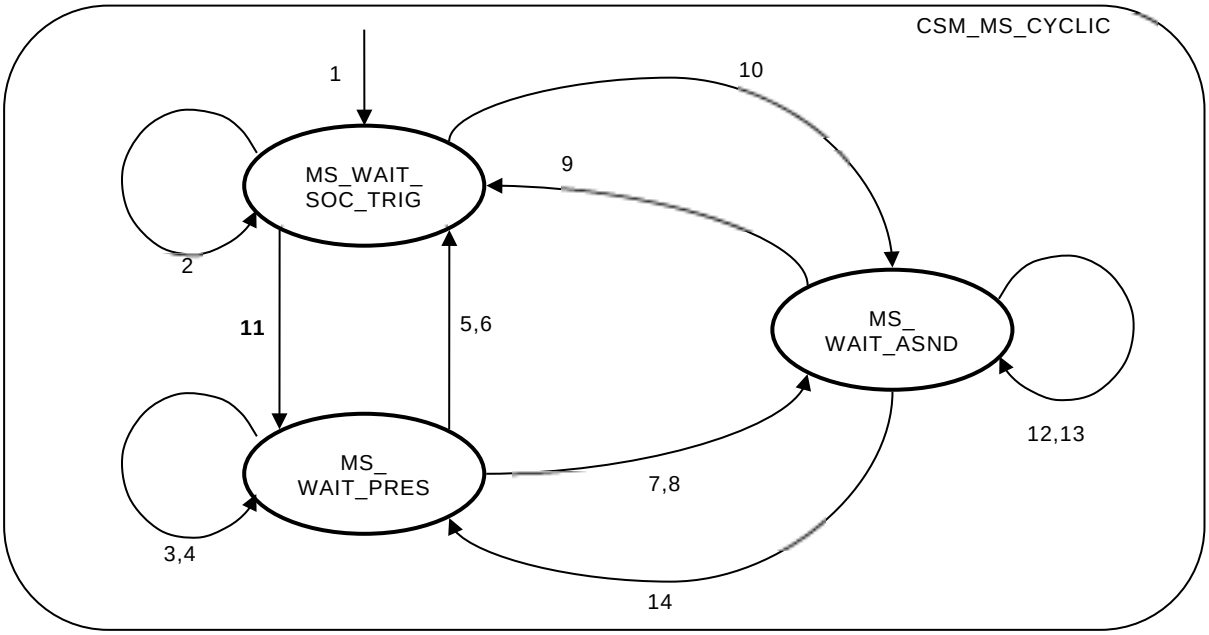


Figure 9 – Diagramme états-transitions du CSM des MN à CSM_MS_CYCLIC

L'état "CYCLIC" signifie que la communication cyclique est en cours. (NMTStatus = NMT_MS_PRE_OPERATIONAL_2, NMT_MS_READY_TO_OPERATE ou NMT_MS_OPERATIONAL).

Suite à l'envoi d'une trame PReq, le diagramme d'états attend une réponse dans l'état "MS_WAIT_PRES". Le temps d'attente est limité par un délai d'attente.

Si une trame SoA avec une invitation est envoyée, le diagramme d'états attend l'état "MS_WAIT_ASND" jusqu'à ce que la phase asynchrone se termine avec le ME_SOC_TRIG( ) = "TRUE".

Les transitions sont décrites dans le Tableau 31.

Tableau 31 – Transitions du CSM des MN à CSM_MS_CYCLIC

#	État courant	Événement /condition ⇒actions	État suivant
1	Tout état	Activation \ =>	MS_WAIT_SOC_TRIG

2	MS_WAIT_SOC_TRIG	CSM-PDO.ind { Isochr, Isochr-time-triggered, Isochr-Out} /
		Isochr = 0 &&
		Async-In = 0 &&
		ME_SOC_TRIG ( ) = “TRUE”
		=>
		SOCDU:= BUILD_SOC ( )
		MA_Data.req { SOCDU }
		IF Isochr-Out <> 0 THEN
		PRESDU:= BUILD_PRES ( )
		MA_DATA.req { PRESDU }
		END IF
		SOADU:= BUILD_SOA ( )
		MA_DATA.req { SOADU }
		IF Async-Out <> 0 THEN
		ASNDDU:= BUILD_ASND ( )
		MA_DATA.req { ASNDDU }
		ENDIF
			MS_WAIT_SOC_TRIG

3	MS_WAIT_PRES	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} /
		Isochr <> 0 &&
		ME_PRES ( ) = “TRUE”
		=>
		IF Isochr-time-triggered = 0 THEN
		PREQDU:= BUILD_PREQ ( )
		MA_DATAreq { PREQDU }
		ENDIF
			MS_WAIT_PRES

4	MS_WAIT_PRES	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} /
		Isochr <> 0 &&
		EXPIRED_TIMER (T(PRES_TIME)) = "TRUE")
		=>
		IF Isochr-time-triggered = 0 THEN
		PREQDU:= BUILD_PREQ ( )
		MA_DATA.req { PREQDU }
		ENDIF
		CSM-Event.ind
		{E_DLL_LOSS_PRES,3002h,time,AdditionalInfo}
			MS_WAIT_PRES

5	MS_WAIT_PRES	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} /
		CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \
		ME_PRES ( ) = “TRUE” &&
		Isochr = 0 &&
		Async-In = 0
		=>
		SOADU:= BUILD_SOA ( )
		MA_DATA.req { SOADU }
		IF Async-Out <> 0 THEN
		ASNDDU:= BUILD_ASND ( )
		MA_DATA.req { ASNDDU }
		ENDIF
			MS_WAIT_SOC_TRIG

6	MS_WAIT_PRES	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} / CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \ EXPIRED_TIMER (T(PRES_TIME)) = "True") && Isochr = 0 && Async-In = 0 => SOADU:= BUILD_SOA ( ) MA_DATA.req { SOADU } IF Async-Out <> 0 THEN ASNDDU:= BUILD_ASND ( ) MA_DATA.req { ASNDDU }	MS_WAIT_SOC_TRIG

#	État courant	Événement /condition ⇒actions	État suivant
		ENDIF CSM-Event.ind {E_DLL_LOSS_PRES,3002h,time,AdditionalInfo}	
7	MS_WAIT_PRES	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} / CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \ ME_PRES ( ) = “TRUE” && Isochr = 0 && Async-In <> 0 => SOADU:= BUILD_SOA ( ) MA_DATA.req { SOADU }	MS_WAIT_ASND

8	MS_WAIT_PRES	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} /
		CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \
		EXPIRED_TIMER (T(PRES_TIME)) = "True") &&
		Isochr = 0 &&
		Async-In <> 0
		=>
		IF Isochr-Out <> 0 THEN
		PRESDU:= BUILD_PRES ( )
		MA_DATA.req { PRESDU }
		ENDIF
		SOADU:= BUILD_SOA ( )
		MA_DATA.req { SOADU }
		CSM-Event.ind
		{E_DLL_LOSS_PRES,3002h,time,AdditionalInfo}
			MS_WAIT_ASND

9	MS_WAIT_ASND	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} /
		CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \
		ME_SOC_TRIG ( ) = “TRUE” &&
		Isochr = 0 &&
		Async-In = 0
		=>
		SOCDU:= BUILD_SOC ( )
		MA_DATA.req { SOCDU }
		IF Isochr-Out <> 0 THEN
		PRESDU:= BUILD_PRES ( )
		MA_DATA.req { PRESDU }
		ENDIF
		SOADU:= BUILD_SOA ( )
		MA_DATA.req { SOADU }
		IF Async-Out <> 0 THEN
		ASNDDU:= BUILD_ASND ( )
		MA_DATA.req { ASNDDU }
		ENDIF
			MS_WAIT_SOC_TRIG

10	MS_WAIT_SOC_TRIG	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} /
		CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \
		ME_SOC_TRIG ( ) = “TRUE” &&
		Isochr = 0 &&
		Async-In <> 0
		=>
		SOCDU:= BUILD_SOC ( )
		MA_DATA.req { SOCDU }
		IF Isochr-Out <> 0 THEN
		PRESDU:= BUILD_PRES ( )
		MA_DATA.req { PRESDU }
		ENDIF
		SOADU:= BUILD_SOA ( )
		MA_DATA.req { SOADU }
			MS_WAIT_ASND

11	MS_WAIT_SOC_TRIG	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} / ME_SOC_TRIG ( ) = “TRUE” && Isochr <> 0 && => SOCDU:= BUILD_SOC ( ) MA_DATA.req { SOCDU } IF Isochr-Out <> 0 THEN	MS_WAIT_PRES

#	État courant	Événement /condition ⇒actions	État suivant

		PRESDU:= BUILD_PRES ( )
		MA_DATA.req { PRESDU }
		END IF
		IF Isochr-time-triggered = 0 THEN
		PREQDU:= BUILD_PREQ ( )
		MA_DATAreq { PREQDU }
		ENDIF	

12	MS_WAIT_ASND	\ ME_ASND ( ) = “TRUE” => CSM-RASND.ind { }	MS_WAIT_ASND

13	MS_WAIT_ASND	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} /
		CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \
		ME_SOC_TRIG ( ) = “TRUE” &&
		Isochr = 0 &&
		Async-In <> 0
		=>
		SOCDU:= BUILD_SOC ( )
		MA_DATA.req { SOCDU }
		IF Isochr-Out <> 0 THEN
		PRESDU:= BUILD_PRES ( )
		MA_DATA.req { PRESDU }
		ENDIF
		SOADU:= BUILD_SOA ( )
		MA_DATA.req { SOADU }
			MS_WAIT_ASND

14	MS_WAIT_ASND	CSM-PDO.ind {Isochr, Isochr-time-triggered, Isochr-Out} /
		CSM-CSM.ind{Async-In,Async-Out,Resp-Expected } \
		ME_SOC_TRIG ( ) = “TRUE” &&
		Isochr <> 0
		=>
		SOCDU:= BUILD_SOC ( )
		MA_DATA.req { SOCDU }
		IF Isochr-Out <> 0 THEN
		PRESDU:= BUILD_PRES ( )
		MA_DATA.req { PRESDU }
		END IF
		IF Isochr-time-triggered = 0 THEN
		PREQDU:= BUILD_PREQ ( )
		MA_DATA.req { PREQDU }
		ENDIF
			MS_WAIT_PRES

7.2.3.2 Table des états de CSM à CN

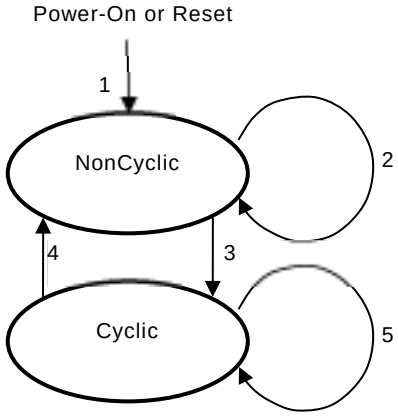
7.2.3.2.1 Vue d’ensemble

Le diagramme d’états de cycle du CN gère la communication au sein d’un cycle de bus de terrain de Type 13. Le CSM surveille l’ordre des trames reçues au sein d’un cycle et réagit comme décrit ci-dessous. L’ordre de réception des trames prévu dépend du paramètre “NMTStatus”.

Si une erreur dans la communication est détectée par le CSM, une primitive d'erreur sera générée. Le CSM essaiera de maintenir la communication quelles que soient les erreurs.

7.2.3.2.2 Table d'états générale

Les transitions de CSM pourraient être représentées au sein d'un seul diagramme où le paramètre “NMTStatus”, reçu par le biais de la primitive DL-NMT, constitue les conditions de transitions. Par souci de compréhension et de clarté, les transitions pertinentes des valeurs uniques de “NMTStatus” sont filtrées et affichées au sein d’un diagramme séparé en tant que “mode de fonctionnement” du CSM. Le diagramme états-transitions de la Figure 10 présente la réaction sur “NMTStatus” et l’activation des diagrammes d’états sous-jacents.



Légende

Anglais	Français
Power-On or Reset	Mise sous tension ou Réinitialisation
NonCyclic	NonCyclic (Non cyclique)
Cyclic	Cyclic (Cyclique)

Figure 10 – Diagramme états-transitions du CSM des CN

L'état "NonCyclic" signifie que la communication isochrone n'a pas encore commencé ou que la connexion a été perdue.

L'état "Cyclic" signifie que le CSM des CN est verrouillé dans la séquence de trames des MN.

Les transitions sont décrites dans le Tableau 32.

Tableau 32 – Transitions du CSM des CN

#	État courant	Événement /condition ⇒actions	État suivant
1	Tout état	POWER-ON ou RESET =>	NonCyclic
2	NonCyclic	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_GS_INITIALISATION || NMTStatus = NMT_CS_NOT_ACTIVE || NMTStatus = NMT_CS_BASIC_ETHERNET || NMTStatus = NMT_CS_PRE_OPERATIONAL_1 => -- Activer le diagramme d'états CSM_CS_NON_CYCLIC -- Désactiver le diagramme d'états CSM_CS_CYCLIC	NonCyclic
3	NonCyclic	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_CS_PRE_OPERATIONAL_2 || NMTStatus = NMT_CS_READY_TO_OPERATE || NMTStatus = NMT_CS_OPERATIONAL => -- Désactiver le diagramme d'états CSM_CS_NON_CYCLIC -- Activer le diagramme d'états CSM_CS_CYCLIC	Cyclic
4	Cyclic	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_GS_INITIALISATION || NMTStatus = NMT_CS_NOT_ACTIVE || NMTStatus = NMT_CS_BASIC_ETHERNET ||	NonCyclic

#	État courant	Événement /condition ⇒actions	État suivant
		NMTStatus = NMT_CS_PRE_OPERATIONAL_1 => -- Activer le diagramme d'états CSM_CS_NON_CYCLIC -- Désactiver le diagramme d'états CSM_CS_CYCLIC	
5	Cyclic	CSM-TNMT.ind { NMTStatus } / NMTStatus = NMT_CS_PRE_OPERATIONAL_2 || NMTStatus = NMT_CS_READY_TO_OPERATE || NMTStatus = NMT_CS_OPERATIONAL => -- Désactiver le diagramme d'états CSM_CS_NON_CYCLIC -- Activer le diagramme d'états CSM_CS_CYCLIC	Cyclic

7.2.3.2.3 CSM_CS_NON_CYCLIC

Le diagramme états-transitions est montré à la Figure 11.

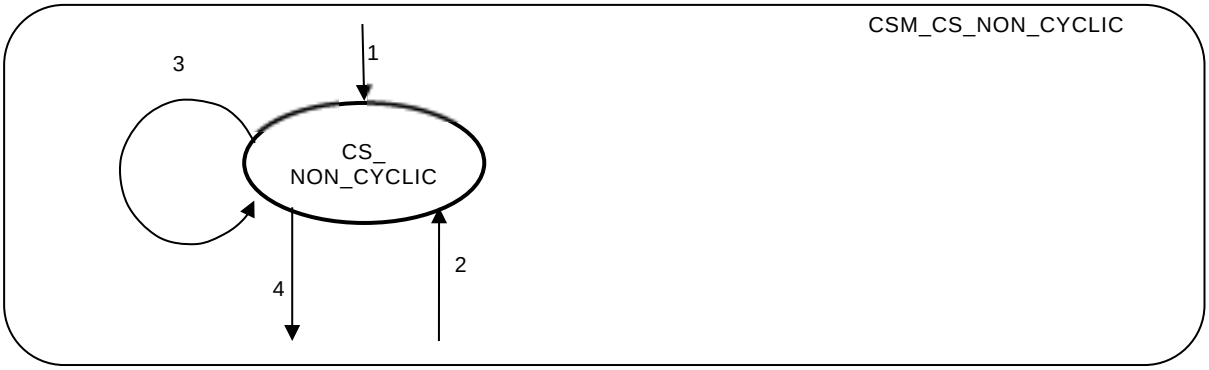


Figure 11 – Diagramme états-transitions du CSM des CN à CSM_CS_NON_CYCLIC

Les transitions sont décrites dans le Tableau 33.

Tableau 33 – Transitions du CSM des CN à CSM_CS_NON_CYCLIC

#	État courant	Événement /condition ⇒actions	État suivant
1	Tout état	Activation =>	CS_NON_CYCLIC
2	CS_WAIT_SOC, CS_WAIT_PREQ, CS_WAIT_SOA	/(CE_SOC ( ) = "TRUE" || CE_PREQ ( ) = "TRUE" || CE_SOA ( ) = "TRUE") && EXPIRED_TIMER (T(FRAME_TIME)) = "True") => -- Réaction spécifique à l'état de NMT --	CS_NON_CYCLIC
3	CS_NON_CYCLIC	/CE_SOA ( ) = "TRUE" || CE_ASND ( ) = "TRUE" => -- Réaction spécifique à l'état de NMT --	CS_NON_CYCLIC
4	CS_NON_CYCLIC	/CE_SOC ( ) = "TRUE => -- Réaction spécifique à l'état de NMT --	CS_WAIT_PREQ

7.2.3.2.4 CSM_CS_CYCLIC

Pour NMTStatus = NMT_CS_OPERATIONAL et NMT_CS_READY_TO_OPERATE, il existe trois trames obligatoires pour un nœud continu, qui doivent survenir à chaque cycle dans

l'ordre précis: SoC, PReq et SoA. Si un nœud est accessible multiplexé, seules les trames SoC et SoA sont obligatoires pour chaque cycle. La trame PReq n'est obligatoire que dans le cycle multiplexé pour lequel le nœud a été configuré.

Pour NMTStatus = NMT_CS_PRE_OPERATIONAL_2, il y a deux trames obligatoires, qui doivent survenir à chaque cycle dans l'ordre SoC et SoA. La trame PReq peut survenir entre SoC et SoA. Dans l'état NMT_CS_PRE_OPERATIONAL_2 la détection du temps d'attente de la SoC est désactivée car le nœud peut ne pas être encore configuré.

Le diagramme d'états de cycle garde une trace de toutes les trames reçues pour détecter les pertes de trames. La réception de trames doit être acceptée par le CN quel que soit l'état courant du CSM.

Le diagramme états-transitions est montré à la Figure 12.

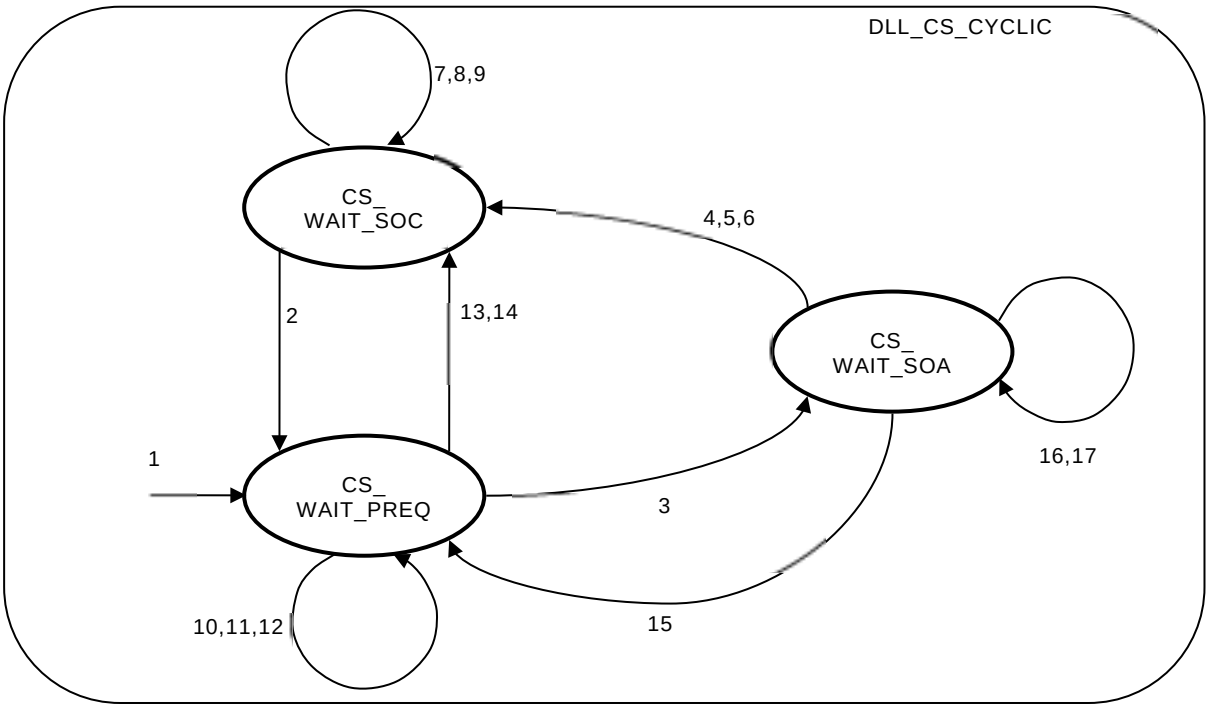


Figure 12 – Diagramme états-transitions du CSM des CN à CSM_CS_CYCLIC

Le diagramme d'états attend dans l'état "CS_WAIT_SOC" après avoir reçu la trame SoA jusqu'au début du prochain cycle (déclenché par une trame SoC du MN). Des trames Ethernet de tout type peuvent être reçues entre les trames SoA et SoC (phase asynchrone).

Après le début du cycle, le diagramme d'états attend dans l'état "CS_WAIT_PREQ" une trame PReq. Après réception de la PReq, le CN doit répondre avec une Trame PRes. Le CN peut recevoir et traiter des Trames PRes d'autres CN lorsqu'il est dans cet état.

Après réception d'une trame PReq, le diagramme d'états attend dans l'état "CS_WAIT_SOA" la réception d'une trame SoA. La réception d'une trame SoA confirme la fin de la phase isochrone. Le CN peut recevoir et traiter des Trames PRes d'autres CN lorsqu'il est dans cet état.

Les transitions sont décrites dans le Tableau 34.

Tableau 34 – Transitions du CSM des CN à CSM_CS_CYCLIC

#	État courant	Événement /condition ⇒actions	État suivant
1	Tout état	Activation =>	CS_WAIT_PREQ
2	CS_WAIT_SOC	/CE_SOC ( ) = “TRUE” => -- synchroniser le début de cycle-- DLM-Frame.ind { SOC }	CS_WAIT_PREQ
3	CS_WAIT_PREQ	/CE_PREQ ( ) = “TRUE” || EXPIRED_TIMER (T(PRES_TT_TIME)) = “TRUE” => IF CE_PREQ ( ) CSM-RPDO.ind { } ENDIF PRESDU:= BUILD_PRES ( ) MA_DATA.req { PRESDU }	CS_WAIT_SOA
4	CS_WAIT_SOA	/CE_SOA ( ) = “TRUE” => CSM-TASND.ind { } -- process SoA -- ASNDDU:= BUILD_ASND ( ) MA_DATA.req { ASNDDU}	CS_WAIT_SOC
5	CS_WAIT_SOA	/EXPIRED_TIMER (T(FRAME_TIME)) = "True") => -- Synchroniser avec la prochaine SoC CSM-Event.ind {E_DLL_LOSS_SOC,3002h,time,AdditionalInfo} CSM-Event.ind {E_DLL_LOSS_SOA,3002h,time,AdditionalInfo}	CS_WAIT_SOC
6	CS_WAIT_SOA	/CE_PREQ ( ) = “TRUE” || EXPIRED_TIMER (T(PRES_TT_TIME)) = “TRUE” => IF CE_PREQ ( ) CSM-RPDO.ind { } -- accept the PReq frame -- ENDIF PRESDU:= BUILD_PRES ( ) MA_DATA.req { PRESDU } CSM-Event.ind {E_DLL_LOSS_SOC,3002h,time,AdditionalInfo} CSM-Event.ind {E_DLL_LOSS_SOA,3002h,time,AdditionalInfo}	CS_WAIT_SOC
7	CS_WAIT_SOC	/CE_ASND ( ) = “TRUE” => CSM-RASND.ind { }	CS_WAIT_SOC
8	CS_WAIT_SOC	/CE_PREQ ( ) = “TRUE” || EXPIRED_TIMER (T(PRES_TT_TIME)) = “TRUE”) => PRESDU:= BUILD_PRES ( ) MA_DATA.req { PRESDU } CSM-Event.ind {E_DLL_LOSS_SOC,3002h,time,AdditionalInfo}	CS_WAIT_SOC
9	CS_WAIT_SOC	/CE_PRES ( ) = “TRUE” || /CE_SOA ( ) = “TRUE” || /EXPIRED_TIMER (T(FRAME_TIME)) = "True") => CSM-Event.ind {E_DLL_LOSS_SOC,3002h,time,AdditionalInfo}	CS_WAIT_SOC
10	CS_WAIT_PREQ	/CE_PRES ( ) = “TRUE” => CSM-RPDO.ind { } -- process PRes (cross traffic) --	CS_WAIT_PREQ
11	CS_WAIT_PREQ	/CE_SOC ( ) = “TRUE” => -- synchroniser avec le début du cycle -- CSM-Event.ind {E_DLL_LOSS_SOA,3002h,time,AdditionalInfo}	CS_WAIT_PREQ

#	État courant	Événement /condition ⇒actions	État suivant
12	CS_WAIT_PREQ	/CE_ASND ( ) = “TRUE” => CSM-Event.ind {E_DLL_LOSS_SOA,3002h,time,AdditionalInfo}	CS_WAIT_PREQ

13	CS_WAIT_PREQ	/CE_SOA ( ) = “TRUE”
		=>
		CSM-TASND.ind { } -- process SoA --
		ASNDDU:= BUILD_ASND ( )
		MA_DATA.req { ASNDDU }
		IF CN <> MULTIPLEXED THEN
		CSM-Event.ind
		{E_DLL_LOSS_PREQ,3002h,time,AdditionalInfo}
		ENDIF
			CS_WAIT_SOC

14	CS_WAIT_PREQ	/EXPIRED_TIMER (T(FRAME_TIME)) = "True") => -- Synchroniser avec la prochaine SoC CSM-Event.ind {E_DLL_LOSS_SOC,3002h,time,AdditionalInfo} CSM-Event.ind {E_DLL_LOSS_SOA,3002h,time,AdditionalInfo}	CS_WAIT_SOC
15	CS_WAIT_SOA	/CE_SOC ( ) = “TRUE” => -- Synchroniser avec la prochaine SoC CSM-Event.ind {E_DLL_LOSS_SOA,3002h,time,AdditionalInfo}	CS_WAIT_PREQ
16	CS_WAIT_SOA	/CE_PRES ( ) = “TRUE” => CSM-RPDO.ind { } -- process PRes (cross traffic) --	CS_WAIT_SOA
17	CS_WAIT_SOA	/CE_ASND ( ) = “TRUE” => CSM-Event.ind {E_DLL_LOSS_SOA,3002h,time,AdditionalInfo}	CS_WAIT_SOA

7.2.4 Fonctions de CSM

Toutes les fonctions utilisées par le CSM sont récapitulées dans le Tableau 35.

Tableau 35 – Table des Fonctions du CSM

Nom de fonction	Entrées	Sorties	Description et opération
BUILD_ASND	Source	ASNDDU	ASNDDU est assemblée comme suit ASNDDU.Destination issu de CSM-TASND.ind ASNDDU.Source:= source ASNDDU.ServiceID issu de CSM-TASND.ind ASNDDU.Payload issu de CSM-TASND.ind SI ASNDDU.ServiceID = IDENT THEN ASNDDU.Payload.PR issu de ATC-Prio.ind ASNDDU.Payload.RS issu de ATC-Prio.ind ASNDDU.Payload.NMTStatus issu de CSM-TNMT.ind ASNDDU.Payload.Type 13 fieldbusVersion:= V(NMT_Type 13 fieldbusVersion_U8) ENDIF SI ASNDDU.ServiceID = STATUS alors ASNDDU.Payload.PR issu de ATC-Prio.ind ASNDDU.Payload.RS issu de ATC-Prio.ind ASNDDU.Payload.NMTStatus issu de CSM-TNMT.ind ASNDDU.Payload.EN issu de CSM-TERR.ind ASNDDU.Payload.EC issu de CSM-TERR.ind ENDIF
BUILD_PREQ	(aucune)	PREQDU	PREQDU est assemblée comme suit PREQDU.Destination:= issu de CSM-TPDO.ind PREQDU.Source:= C_ADDR_MN_DEF_NODE_ID PREQDU.MS issu de ME_SOC_TRIG ( )

Nom de fonction	Entrées	Sorties	Description et opération
			PREQDU.EA issu de CSM-TERR.ind PREQDU.RD issu de CSM-TPDO.ind PREQDU.Payload issu de CSM-TPDO.ind
BUILD_PRES	(aucune)	PRESDU	PRESDU est assemblée comme suit PRESDU.Destination C_ADDR_BROADCAST PRESDU.Source:= NodeID PRESDU.NMTStatus issu de CSM-TNMT.ind PRESDU.MS issu de ME_SOC_TRIG ( ) PRESDU.EN issu de CSM-TERR.ind PRESDU.RD issu de CSM-TPDO.ind PRESDU.PR issu de ATC-Prio.ind PRESDU.RS issu de ATC-Prio.ind PRESDU.Payload issu de CSM-TPDO.ind
BUILD_SOA	(aucune)	SOADU	SOADU est assemblée comme suit SOADU.Destination:= C_ADDR_BROADCAST SOADU.Source:= C_ADDR_MN_DEF_NODE_ID SOADU.NMTStatus issu de CSM-TNMT.ind SOADU.EA issu de CSM-TERR.ind SOADU.ER issu de CSM-TERR.ind SOADU.RequestedServiceID issu de CSM-SoA.ind SOADU.RequestedServiceTarget issu de CSM-SoA.ind SOADU.Type 13 fieldbusVersion = V(NMT_Type 13 fieldbusVersion_U8) SOADU.ER:= ERRDU.ER
BUILD_SOC	(aucune)	SOCDU	SOCDU est assemblée comme suit SOCDU.Destination:= C_ADDR_BROADCAST SOCDU.Source:= C_ADDR_MN_DEF_NODE_ID SOCDU.MC issu de ME_SOC_TRIG ( ) SOCDU.PS issu de ME_SOC_TRIG ( ) SOCDU.NetTime issu de ME_SOC_TRIG ( ) SOCDU.RelativeTime issu de ME_SOC_TRIG ( )
CE_ASND	(aucune)	True (vrai)/False (faux)	Cette fonction signifie qu'une trame ASnd ou une trame qui n'est pas de bus de terrain de Type 13 a été reçue. Étant donné que les types de trames durant la phase asynchrone ne sont pas limités aux types de bus de terrain de Type 13, cet événement est généré à la réception de toutes les trames Ethernet légales
CE_PREQ	(aucune)	True (vrai)/False (faux)	Cette fonction signifie qu'une trame PReq a été reçue du MN.
CE_PRES	(aucune)	True (vrai)/False (faux)	Le CN peut être configuré pour traiter les trames PRes d'autres CN et du MN (trafic transversal). À chaque fois qu'une trame PRes est reçue, un événement DLL_CE_PRES est généré
CE_SOA	(aucune)	True (vrai)/False (faux)	Cette fonction signifie qu'une trame SoA a été reçue du MN. Elle marque la fin de la phase isochrone du cycle et le début de la phase asynchrone
CE_SOC	(aucune)	True (vrai)/False (faux)	Cette fonction signifie qu'une trame SoC de bus de terrain de Type 13 a été reçue du MN. Elle marque le début d'un nouveau cycle et simultanément, le début de la phase isochrone du cycle
ME_ASND	(aucune)	True (vrai)/False (faux)	Cette fonction signifie qu'une trame ASnd ou une trame qui n'est pas de bus de terrain de Type 13 a été reçue.
ME_PRES	(aucune)	True (vrai)/False (faux)	Cette fonction signifie qu'une trame PRes a été reçue
ME_SOA_TRIG	(aucune)	True (vrai)/False (faux)	Cette fonction signifie qu'un nouveau cycle de bus de terrain de Type 13 réduit doit commencer. La fonction peut être générée soit de manière cyclique soit directement après réception d'une trame ASnd pour poursuivre le cycle de bus de terrain de Type 13 réduit le plus vite possible
ME_SOC_TRIG	(aucune)	True (vrai)/False (faux)	Cette fonction déclenche la transmission de la trame SoC et commence un nouveau cycle de bus de terrain de Type 13. La précision temporelle détermine la précision de synchronisation du réseau de bus de terrain de Type 13

Nom de fonction	Entrées	Sorties	Description et opération
EXPIRED_TIMER	T(x)	True (vrai)/False (faux)	Lorsque le temporisateur requis x a expiré, “True” (Vrai) est retourné, dans le cas contraire, “False” (Faux) est retourné
START_TIMER	T(x), V(y)	(aucune)	Le temporisateur x est mis à la valeur de y, et est activé

7.3 Commande TX/RX en transmission isochrone (ITC)

7.3.1 Vue d’ensemble

Le transfert de données en temps réel est effectué à l’aide d’objets de données de processus (PDO). La communication des PDO dans le bus de terrain de Type 13 est toujours effectuée de façon isochrone par les trames PReq et/ou PRes. Les trames PRes sont envoyées comme des diffusions ou des multidiffusions selon le schéma producteur/consommateur. Les trames PReq avec des adresses à diffusion unique satisfont à la relation maître/esclave.

Le type de transmission de PDO est continu. Il n’y a pas de types de transmission “sur événement” ou “sur modification” fournis.

7.3.2 Intervalles de temps multiplexés

7.3.2.1 Généralités

Le bus de terrain de Type 13 prend en charge les classes de communication de CN qui déterminent les cycles dans lesquels les nœuds sont à adresser lors de la phase isochrone:

- Continuous (continu): les données continues doivent être échangées dans chaque cycle de bus de terrain de Type 13.
- Continuous-time-triggered (cadencé en continu): les données cadencées en continu doivent être échangées dans chaque cycle de bus de terrain de Type 13.
- Multiplexed (multiplexé): les données multiplexées émises vers et à partir d’un CN ne doivent pas être échangées dans chaque cycle de bus de terrain de Type 13.

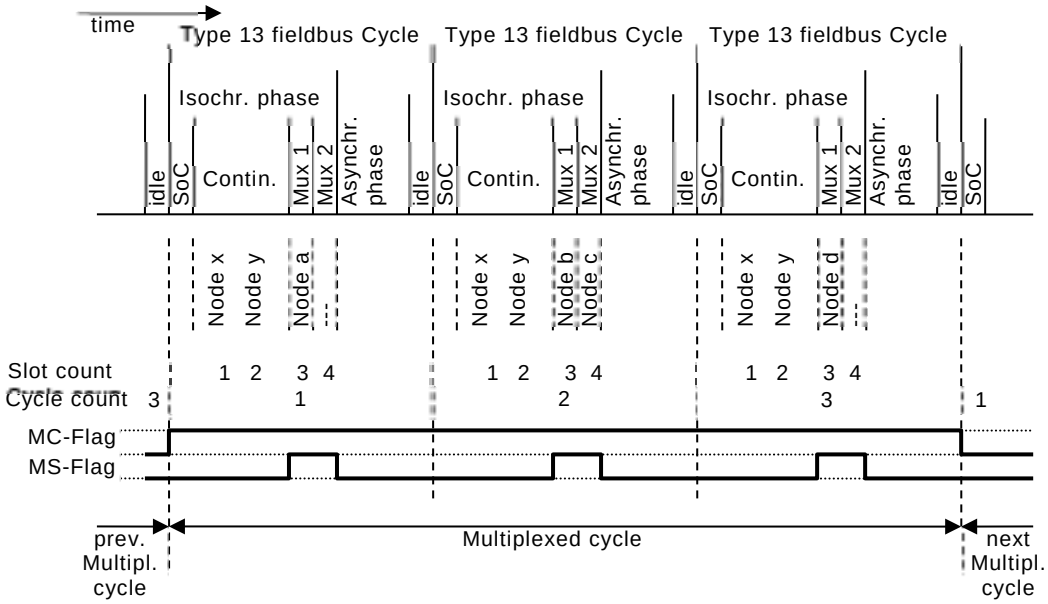
Les accès aux CN multiplexés doivent être dispersés au cycle multiplexé composé d’un certain nombre de cycles de bus de terrain de Type 13. La dispersion permet un accès isochrone à un grand nombre de CN sans allonger le cycle de bus de terrain de Type 13 de façon inacceptable. Cependant, l’accès aux CN multiplexés réduit la fréquence d'échange sur interrogation à un CN particulier.

L’ordre dans lequel les CN sont échangés sur interrogation est spécifique à la mise en œuvre. Il convient qu'une mise en œuvre rassemble les DLPDU PReq/PRes effectuées au début de la phase isochrone. Il convient qu’elle fournisse des moyens pour réagencer l’ordre d’échange sur interrogation, afin d’éviter les lieux des nœuds ayant la valeur de latence SoC la pire à l’intervalle de temps suivant SoC.

7.3.2.2 Répartition des intervalles de temps multiplexés

L’accès continu, cadencé en continu et multiplexé peut fonctionner en parallèle lors d’un cycle de bus de terrain de Type 13.

La Figure 13 montre l’attribution des intervalles de temps multiplexés à des CN. Les intervalles de temps multiplexés sont identifiés par “Mux 1” et “Mux 2”.



Légende

Anglais	Français
time	Temps
Type 13 fieldbus Cycle	Cycle de bus de terrain de Type 13
Isochr. phase	Phase isochrone
idle	Inactif
Contin.	Continu
Asynchr. phase	Phase asynchrone
Node x	Nœud x
Node y	Nœud y
Node a	Nœud a
Node b	Nœud b
Node c	Nœud c
Node d	Nœud d
Slot count	Nombre d'intervalles
Cycle count	Nombre de cycles
MC-Flag	Indicateur MC
MS-Flag	Indicateur MS
prev. Multipl. cycle	Cycle multiplexé précédent
Multiplexed cycle	Cycle multiplexé
Next Multipl. cycle	Prochain cycle multiplexé

Figure 13 – Exemple d'attribution d'intervalles de temps multiples

La capacité d'un nœud activé par un MN à effectuer la commande d'un fonctionnement isochrone multiplexé doit être indiquée par l'élément de description de l'appareil P(D_DLL_MNFeatureMultiplex_BOOL). La capacité d'un nœud activé par un CN à être accessible de façon isochrone de manière multiplexée doit être indiquée par l'élément de description de l'appareil P(D_DLL_CNFeatureMultiplex_BOOL).

L'attribution est contrôlée par V(MultiplCycleCnt_U8) et V(NMT_MultiplCycleAssign_AU8). V(MultiplCycleCnt_U8) définit la longueur du cycle multiplexé dans les nombres de cycle de

bus de terrain de Type 13. Si V(MultiplCycleCnt_U8) est égal à zéro, la méthode d'accès multiplexé ne doit pas être appliquée, par exemple, tous les CN doivent être accessibles de manière continue.

V(NMT_MultiplCycleAssign_AU8) définit le nombre de cycles dans le cycle multiplexé, lorsque le CN respectif doit être échangé sur interrogation par le MN. Si le sous-indice est zéro, le CN doit être accessible de façon continue.

Le nombre d'intervalles de temps isochrones par cycle isochrone est déduit du nombre maximal d'attributions aux CN cumulé à un cycle.

V(NMT_IsochrSlotAssign_AU8) attribue les CN à un intervalle de temps isochrone particulier. Le cycle de bus de terrain de Type 13 isochrone peut être divisé en intervalles de communication, chacun étant composé d'un message PReq et PRes pour un nœud particulier.

Chaque sous-indice dans la matrice correspond à un intervalle de communication individuel égal au sous-indice. La valeur 0 indique qu'il n'y a pas de nœud particulier attribué à l'intervalle de communication.

Les sous-indices peuvent également être utilisés pour l'attribution des intervalles de temps de nœuds multiplexés. Si des nœuds multiplexés et non-multiplexés doivent être attribués à des intervalles de communication particuliers, il est important de s'assurer que toutes les stations non multiplexées isochrones sont configurées sur les sous-indices inférieurs de V((NMT_IsochrSlotAssign_AU8). Il faut également veiller à ce que les intervalles de temps multiplexes soient mis en correspondance avec les intervalles de communication dans l'ordre croissant. Cela signifie que le premier nœud multiplexé attribué à V(NMT_IsochrSlotAssign_AU8) doit être configuré dans le premier intervalle de temps multiplexe dans V(NMT_MultiplCycleAssign_AU8), et ainsi de suite.

Les espaces dans le NMT_IsochrSlotAssign_AU8 sont autorisés et les intervalles de communication non utilisés sont supprimés.

Le tableau suivant présente les valeurs des paramètres qui contrôlent le système de la Figure 13:

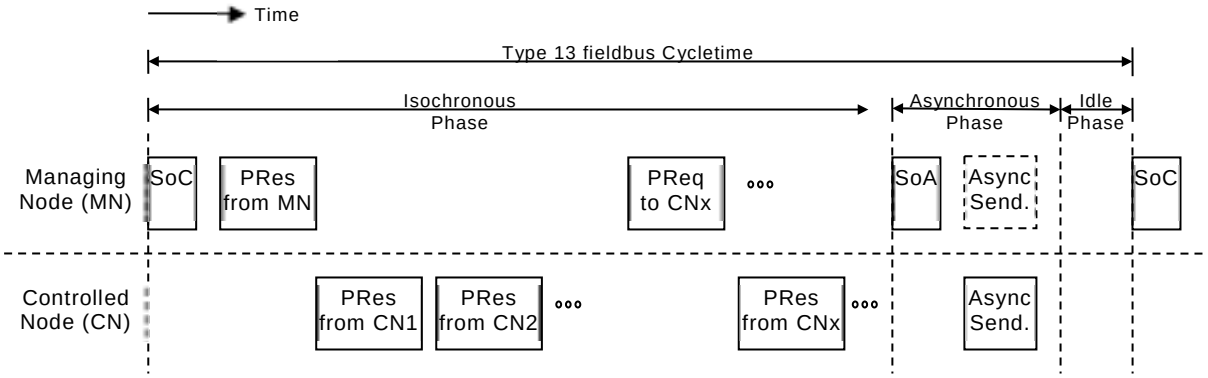
Tableau 36 – Exemple d'attribution d'intervalle de temps isochrone

Variable	Valeur	Variable	Valeur
MultiplCycleCnt_U8	3	NMT_IsochrSlotAssign_AU8[0]	6
NMT_MultiplCycleAssign_AU8[$NodeID_a$]	1	NMT_IsochrSlotAssign_AU8[1 (Slot 1)]	$NodeID_x$
NMT_MultiplCycleAssign_AU8[$NodeID_b$]	2	NMT_IsochrSlotAssign_AU8[2 (Slot 2)]	$NodeID_y$
NMT_MultiplCycleAssign_AU8[$NodeID_c$]	2	NMT_IsochrSlotAssign_AU8[3 (Slot 3)]	$NodeID_a$
NMT_MultiplCycleAssign_AU8[$NodeID_d$]	3	NMT_IsochrSlotAssign_AU8[4 (Slot 4)]	$NodeID_b$
NMT_MultiplCycleAssign_AU8[$NodeID_x$]	0	NMT_IsochrSlotAssign_AU8[5 (Slot 5)]	$NodeID_c$
NMT_MultiplCycleAssign_AU8[$NodeID_y$]	0	NMT_IsochrSlotAssign_AU8[6 (Slot 6)]	$NodeID_d$

7.3.3 PRes cadencée

Au lieu d'envoyer la PRes en réponse à une PReq, les nœuds de la classe de communication "continuous-time-triggered" (cadencé en continu) envoient leur PRes de manière cadencée.

Par conséquent, le MN doit envoyer sa PRes juste après le SoC. Cette PRes provenant du MN fait office de référence pour tous les nœuds pour lancer leur temporisateur d'envoi.



Légende

Anglais	Français
Time	Temps
Type 13 fieldbus Cycletime	Durée de cycle de bus de terrain de Type 13
Isochronous Phase	Phase isochrone
Asynchronous Phase	Phase asynchrone
Idle Phase	Phase d'inactivité
Managing Node (MN)	Nœud gérant (MN)
PRes from MN	PRes du MN
PReq to CNx	PReq au CNx
Async Send.	Envoi asynchrone
SoC	SoC
Controlled Node (CN)	Nœud commandé (CN)
PRes from CN1	PRes du CN1
PRes from CN2	PRes du CN2
PRes from CNx	PRes du CNx
Async Send.	Envoi asynchrone

Figure 14 – Exemple de PRes cadencée

La capacité d'un nœud à effectuer la commande d'un envoi cadencé de PRes doit être indiquée par l'élément de description de l'appareil P(D_DLL_FeaturePResTimeTriggered_BOOL).

L'heure à laquelle envoyer la PRes est commandée par V(PRES_TT_TIMEOUT). V(PRES_TT_TIMEOUT) définit le temps entre la fin de la réception de la PRes du MN et le début de l'envoi de la PRes.

Par conséquent, après réception de la PRes provenant du MN, le temporisateur T(PRES_TT_TIME) est lancé et lorsqu'il expire, le CN envoie sa PRes.

7.3.4 Définitions des primitives

7.3.4.1 Primitive échangée entre l'ITC et l'utilisateur de DLS

Le Tableau 37 récapitule toutes les primitives échangées entre l'ITC et l'utilisateur de DLS.

Tableau 37 – Primitives échangées entre l'ITC et l'utilisateur de DLS

Nom de primitive	Source	Paramètres associés	Description
DL-PDO.req	Utilisateur de DLS	D_addr DLSDU	Transmet (publie) un élément PDO provenant de l'utilisateur de DLS. L'élément PDO est mis en mémoire tampon dans un tampon local dépendant de "D_addr", jusqu'à ce qu'il soit demandé par le biais de la primitive CSM-TPDO du CSM. MN: Un tampon pour chaque nœud actif est disponible. CN: Un seul tampon est disponible. "D_addr" est mis à 255 (Diffusion)
DL-PDO.ind	ITC	S_addr D_addr DLSDU	Reporte un élément PDO reçu avec ses paramètres associés à l'utilisateur de DLS. Ni la mise en mémoire tampon, ni le traitement n'est effectué avec ces éléments

Les paramètres utilisés pour l'échange de primitives entre l'ITC et l'utilisateur de DLS sont décrits dans le Tableau 38.

Tableau 38 – Paramètres utilisés avec les primitives échangées entre l'ITC et l'utilisateur de DLS

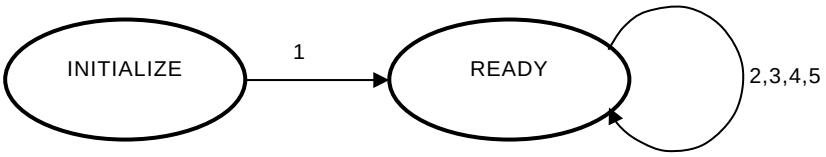
Nom de paramètre	Description
S_addr	Le paramètre "S_addr" spécifie l'adresse de DL de l'éditeur.
D_addr	Le paramètre "D_addr" spécifie l'adresse de DL de l'abonné.
DLSDU	Ce paramètre spécifie l'information qui est transférée par le biais d'un tampon à partir de la DLE locale en tant qu'un éditeur vers les DLE homologues multiples distantes en tant qu'abonnées

7.3.4.2 Primitive échangée entre l'ITC et le CSM

La primitive et ses paramètres associés entre l'ITC et le CSM sont décrits en 7.2.2.1.

7.3.5 Table des états de l'ITC

La Figure 15 représente le diagramme états-transitions de l'ITC, et la table des états de l'ITC est présentée dans le Tableau 39.



Légende

Anglais	Français
INITIALIZE	INITIALIZE (INITIALISATION)
READY	READY (PRÊT)

Figure 15 – Diagramme états-transitions de l'ITC

Tableau 39 – Transitions de l'ITC

#	État courant	Événement /condition ⇒actions	État suivant
1	INITIALIZE	POWER-ON ou RESET / =>	READY
2	READY	CSM-RPDO.ind { PDODU } / => DL-PDO.ind { PDODU }	READY
3	READY	DL-PDO.req {addr:= D_addr, PDODU } / ((CN = "TRUE" && addr = C_ADDR_BROADCAST) || MN = "TRUE") => BUFFER_PDODATA (addr , PDODU) -- DL-PDO.cnf { Status:="OK" }	READY
4	READY	DL-PDO.req {addr:= D_addr, PDODU } / (CN = "TRUE" && addr <> C_ADDR_BROADCAST) => -- DL-PDO.cnf { Status:="KO" }	READY

5	READY	IF MN = "TRUE" THEN
		addr:= SLOT_CTR ( )
		ELSE
		addr:= C_ADDR_BROADCAST
		ENDIF
		/
		=>
		PDODU:= DEBUFFER_PDODATA (addr)
		CSM-TPDO.ind { D_addr:= addr, PDODU }
			READY

7.3.6 Fonctions de l'ITC

Une synthèse de toutes les fonctions utilisées par l'ITC est présentée dans le Tableau 40.

Tableau 40 – Table des Fonctions de l'ITC

Nom de fonction	Entrée	Sortie	Opération
BUFFER_PDOPDATA	Addr, PDODU	(aucune)	Mise en mémoire tampon des données d'entrée dans le tampon (addr) PDODATA
DEBUFFER_PDODATA	Addr	PDODU	Récupération des données de sortie du tampon (addr) PDODATA
SLOT_CTR	(aucune)	(addr)	Détermine le prochain ID de Nœud, qui doit être servi au prochain intervalle de temps isochrone

7.4 Commande TX/RX en transmission asynchrone (ATC)

7.4.1 Vue d'ensemble

La commande TX/RX en transmission asynchrone (ATC) est chargée de la mise en mémoire tampon et de la répartition dans le temps des DLSDU reçues de la trame ASND entre l'utilisateur de DLS et le CSM.

7.4.2 Définitions des primitives

7.4.2.1 Primitive échangée entre l'ATC et l'utilisateur de DLS

Le Tableau 41 récapitule toutes les primitives échangées entre l'ATC et l'utilisateur de DLS.

Tableau 41 – Primitives échangées entre l'ATC et l'utilisateur de DLS

Nom de primitive	Source	Paramètres associés	Description
DL-SDO.req	Utilisateur de DLS	SDO_C_addr SDO_S_addr SDO_Prio SDO_DLSDU	Transmet la Demande du Client au Serveur.
DL-SDO.ind	ATC	SDO_C_addr SDO_S_addr SDO_DLSDU	Reçoit la demande du Client au Serveur
DL-SDO.res	Utilisateur de DLS	SDO_C_addr SDO_S_addr SDO_Prio SDO_DLSDU	Transmet la réponse du Serveur au Client
DL-SDO.cnf	ATC	SDO_C_addr SDO_S_addr SDO_DLSDU	Réponse du Serveur destinataire
DL-UDT.req	Utilisateur de DLS	UDT_S_addr UDT_D_addr UDT_Prio UDT_DLSDU	Transmet des données non spécifiées
DL-UDT.ind	ATC	UDT_S_addr UDT_D_addr UDT_DLSDU	Reçoit des données non spécifiées
DL-STA.req	Utilisateur de DLS	STA_S_addr	Demande la réponse de statut de l'esclave
DL-STA.ind	ATC	STA_S_addr	Reçoit la demande du maître et prépare la réponse du statut
DL-STA.res	Utilisateur de DLS	STA_DLSDU	Transmet la réponse du statut au maître
DL-STA.cnf	ATC	STA_S_addr STA_DLSDU	Reçoit la réponse de statut de l'esclave
DL-IDE.req	Utilisateur de DLS	IDE_S_addr	Demande la réponse d'identification de l'esclave
DL-IDE.ind	ATC	IDE_S_addr	Reçoit la demande du maître et prépare la réponse d'identification
DL-IDE.res	Utilisateur de DLS	IDE_DLSDU	Transmet la réponse d'identification au maître
DL-IDE.cnf	ATC	IDE_S_addr IDE_DLSDU	Reçoit la réponse d'identification de l'esclave
DL-CMD.req	Utilisateur de DLS	CMD_D_addr CMD_DLSDU	Transmet la commande NMT
DL-CMD.ind	ATC	CMD_DLSDU	Reçoit la commande NMT
DL-SYN.req	Utilisateur de DLS	SYN_S_addr SYN_DLSDU	Demande la réponse de synchronisation de l'esclave
DL-SYN.ind	ATC	SYN_S_addr SYN_DLSDU	Reçoit la demande du maître et prépare la réponse de synchronisation
DL-SYN.res	Utilisateur de DLS	SYN_DLSDU	Transmet la réponse d'identification au maître
DL-SYN.cnf	ATC	SYN_S_addr SYN_DLSDU	Reçoit la réponse de synchronisation de l'esclave

Les paramètres utilisés pour l'échange de primitives entre l'ATC et l'utilisateur de DLS sont décrits dans le Tableau 42.

Tableau 42 – Paramètres utilisés avec les primitives échangées entre l'ATC et l'utilisateur de DLS

Nom de paramètre	Description
SDO_C_addr	Le paramètre "client-address" spécifie l'adresse de DL relative à la DLE client
SDO_S_addr	Le paramètre "server-address" spécifie l'adresse de DL relative à la DLE serveur
SDO_Prio	Ce paramètre spécifie la priorité de la trame dans la file d'attente d'envoi asynchrone
SDO_DSLDU	Ce paramètre spécifie les données de l'utilisateur de DLS qui doivent être envoyées par la DLE
UDT_S_addr	Le paramètre "source-address" spécifie l'adresse de DL relative à la DLE serveur.
UDT_D_addr	Le paramètre "destination-address" spécifie l'adresse de DL relative à la DLE serveur.
UDT_Prio	Ce paramètre spécifie la priorité de la trame dans la file d'attente d'envoi asynchrone
UDT_DLSDU	Ce paramètre spécifie les données de l'utilisateur de DLS qui doivent être envoyées par la DLE
STA_S_addr	Le paramètre "slave-adress" spécifie l'adresse de DL relative à la DLE esclave, qu'il convient qu'elle réponde avec son message de statut
STA_DLSDU	Ce paramètre spécifie les données de l'utilisateur de DLS (message de statut) qui doivent être envoyées par la DLE NOTE Certains des paramètres de statut sont insérés/supprimés par la DLL en raison de l'apparition dupliquée de ces informations dans des trames normales et dans le message de statut.
IDE_S_addr	Le paramètre "slave-adress" spécifie l'adresse de DL relative à la DLE esclave, qu'il convient qu'elle réponde avec son message d'identification
IDE_DLSDU	Ce paramètre spécifie les données de l'utilisateur de DLS (message d'identification) qui doivent être envoyées par la DLE NOTE Certains des paramètres d'identification sont insérés/supprimés par la DLL en raison de l'apparition dupliquée de ces informations dans des trames normales et dans le message d'identification.
CMD_D_addr	Le paramètre "destination-address" spécifie l'adresse de DL relative à la DLE abonnée. La valeur 255, utilisée pour un message de diffusion, indique ce message pour tous les nœuds connectés.
CMD_DLSDU	Ce paramètre spécifie l'information qui est transférée par le biais d'un tampon à partir de la DLE locale en tant qu'un éditeur vers les DLE homologues multiples distantes en tant qu'abonnées. La différenciation ultérieure entre les différentes commandes NMT possibles et l'interprétation des données incluses dans la commande NMT sont à la charge de l'utilisateur de DLS.
SYN_S_addr	Le paramètre "slave-adress" spécifie l'adresse de DL relative à la DLE esclave, qu'il convient qu'elle réponde avec son message de synchronisation
SYN_DLSDU	Ce paramètre spécifie les données de l'utilisateur de DLS (message de synchronisation) qui doivent être envoyées par la DLE NOTE Certains des paramètres de synchronisation sont insérés/supprimés par la DLL en raison de l'apparition dupliquée de ces informations dans des trames normales et dans le message de synchronisation.

7.4.2.2 Primitive échangée entre l'ATC et l'ES

Le Tableau 43 récapitule toutes les primitives échangées entre l'ATC et l'ES.

Tableau 43 – Primitives échangées entre l'ATC et l'ES

Nom de primitive	Source	Paramètres associés	Description
ES-STA.req	ES	Destination	Demande un transfert des Données de Statut par le nœud spécifié. Cette primitive n'est disponible qu'au MN

Les paramètres utilisés pour l'échange de primitives entre l'ATC et l'ES sont décrits dans le Tableau 44.

Tableau 44 – Paramètres utilisés avec les primitives échangées entre l'ATC et l'ES

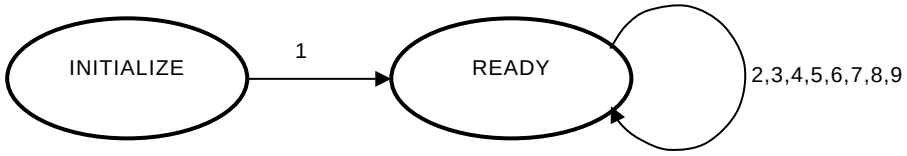
Nom de paramètre	Description
Destination	Ce paramètre indique l'ID de nœud du nœud émettant

7.4.2.3 Primitive échangée entre l'ATC et le CSM

La primitive et ses paramètres associés entre l'ATC et le CSM sont décrits en 7.2.2.2.

7.4.3 Table des états de l'ATC

La Figure 16 représente le diagramme états-transitions de l'ATC, et la table des états de l'ATC est présentée dans le Tableau 45.



Légende

Anglais	Français
INITIALIZE	INITIALIZE (INITIALISATION)
READY	READY (PRÊT)

Figure 16 – Diagramme états-transitions de l'ATC

Tableau 45 – Transitions de l'ATC

#	État courant	Événement /condition ⇒actions	État suivant
1	INITIALIZE	POWER-ON ou RESET / =>	READY

2	READY	CSM-RASND.ind { ASNDDU }
		/
		=>
		IF ASNDDU.SERVICEID = IDENT_RESPONSE THEN
		DL-IDE.cnf { ASNDDU.payload }
		ELSE IF ASNDDU.SERVICEID = STATUS_RESPONSE
		THEN DL-STA.cnf { ANSDDU.payload }
		ELSE IF ASNDDU.SERVICEID = NMT_REQUEST THEN
		DL-IDE.req { ASNDDU.payload }
		ELSE IF ASNDDU.SERVICEID = NMT_COMMAND THEN
		DL-CMD.ind { ASNDDU.payload }
		ELSE IF ASNDDU.SERVICEID = SDO THEN
		DL-SDO.ind { ASNDDU.payload }
		DL-SDO.cnf { ASNDDU.payload }
		ELSE IF ASNDDU.SERVICEID = SYNC_RESPONSE
		THEN
		DL-SYN.cnf { ASNDDU.payload }
		ELSE
		DL-UDT.ind { ASNDDU.payload }
		ENDIF
			READY

#	État courant	Événement /condition ⇒actions	État suivant

3	READY	DL-SDO.req { prio,SDODU }
		DL-SDO.res { prio,SDODU }
		/
		=>
		IF CHECK_ASNDDATAQ (prio) <> “Full” THEN
		QUEUE_ASNDDATA { prio,SDODU }
		-- DL-SDO.cnf { Status:=”OK” }
		ELSE
		-- DL-SDO.cnf { Status:="KO" }
		ENDIF
			READY

4	READY	DL-UDT.req { prio, IPDU }
		/
		=>
		IF CHECK_ASNDDATAQ (prio) <> “Full” THEN
		QUEUE_ASNDDATA { prio,IPDU }
		-- DL-UDT.cnf { Status:=”OK” }
		ELSE
		--DL-UDT.cnf { Status:="KO" }
		ENDIF
			READY

5	READY	DL-STA.req { STADU }
		ES-STA.res { STADU }
		/
		=>
		IF MN = “TRUE” THEN
		IF CHECK_ASNDDATAQ (8) <> “Full” THEN
		QUEUE_ASNDDATA { 8,STADU }
		-- DL-STA.cnf { Status:=”OK” }
		ELSE
		--DL-STA.cnf { Status:="KO" }
		ENDIF
		ELSE
		IF CHECK_ASNDDATAQ (11) <> “Full” THEN
		NMTDU:= BUILD_NMTREQUEST
		(STATUS_REQUEST,src)
		QUEUE_ASNDDATA { 11,NMTDU }
		-- DL-IDE.cnf { Status:=”OK” }
		ELSE
		-- DL-IDE.cnf { Status:=”KO” }
		ENDIF
		ENDIF
			READY

6	READY	DL-IDE.req { IDEDU }
		/
		=>
		IF MN = “TRUE” THEN
		IF CHECK_ASNDDATAQ (9) <> “Full” THEN
		QUEUE_ASNDDATA { 9,STADU }
		-- DL-IDE.cnf { Status:=”OK” }
		ELSE
		--DL-IDE.cnf { Status:="KO" }
		ENDIF
		ELSE
		IF CHECK_ASNDDATAQ (11) <> “FULL” THEN
		NMTDU:= BUILD_NMTREQUEST
		(IDENT_REQUEST,src)
		QUEUE_ASNDDATA { 11,NMTDU }
		-- DL-IDE.cnf { Status:=”OK” }
		ELSE
		-- DL-IDE.cnf { Status:=”KO” }
		ENDIF
		ENDIF
			READY

7	READY	DL-CMD.req { NMTDU }
		/
		=>
		IF CHECK_ASNDDATAQ (11) <> “Full” THEN
		QUEUE_ASNDDATA { 11,NMTDU }
		-- DL-CMD.cnf { Status:=”OK” }
		ELSE
		-- DL-CMD.cnf { Status:=”KO” }
		ENDIF
			READY

#	État courant	Événement /condition ⇒actions	État suivant

8	READY	CSM-TASND.ind { dest,ServiceID,ASNDDU }
		/
		=>
		IF ServiceID = NO_SERVICE THEN
		ASNDDU:= BUILD_ASND (NO_SERVICE)
		ELSE IF ServiceID = IDENT_REQUEST THEN
		dummy:= DEQUEUE_ASNDDATA(9)
		DL-IDE.ind { }
		DL-IDE.res { ASNDDU }
		ELSE IF ServiceID = STATUS_REQUEST THEN
		dummy:= DEQUEUE_ASNDDATA(8)
		DL-STA.ind { }
		DL-STA.res { ASNDDU }
		ELSE IF ServiceID = SYNC_REQUEST THEN
		dummy:= DEQUEUE_ASNDDATA(10)
		DL-SYN.ind { }
		DL-SYN.res { ASNDDU }
		ELSE IF ServiceID = NMT_REQUEST_INVITE THEN
		ASNDDU:= DEQUEUE_ASNDDATA(11)
		ELSE
		prio:= FIND_HIGHESTPRIO ( )
		ASNDDU:= DEQUEUE_ASNDDATA(prio)
		ENDIF
		CSM_TASND.ind { ASNDDU }
			READY

9	READY	ATC-Prio.req { } / => Priority:= FIND_HIGHESTPRIO ( ) RequestToSend:= FIND_HIGHESTPRIO_RS ( ) ATC-Prio.ind {Priority , RequestToSend }	READY

10	READY	DL-SYN.req { SYNDU }
		/
		=>
		IF MN = “TRUE” THEN
		IF CHECK_ASNDDATAQ (10) <> “Full” THEN
		QUEUE_ASNDDATA {10,SYNDU }
		-- DL-SYN.cnf { Status:=”OK” }
		ELSE
		--DL-SYN.cnf { Status:="KO" }
		ENDIF
		ENDIF
			READY

7.4.4 Fonctions de l’ATC

Une synthèse de toutes les fonctions utilisées par l’ITC est présentée dans le Tableau 46.

Tableau 46 – Table des Fonctions de l'ATC

Nom de fonction	Entrée	Sortie	Opération
BUILD_ASND	ServiceID	ASNDDU	ASNDDU est assemblé avec le "ServiceID" requis
BUILD_NMTREQUEST	ServiceID ServiceTarget	NMTDU	NMTDU est assemblée avec le "ServiceID" et le "ServiceTarget" requis
CHECK_ASNDDATAQ	Addr	Status	Vérifie le statut de la file d'attente (addr) pour PDOPDATA. Le statut retourné est, "Full", "Empty" ou "Queue"
DEQUEUE_ASNDDATA	Addr	ASNDDU	Retire de la file d'attente PDODATA (addr) sur une base FIFO
FIND_HIGHESTPRIO	(aucune)	PR	Retourne la plus haute priorité de la file d'attente, qui n'est pas vide
FIND_HIGHESTPRIO_RS	(aucune)	RS	Retourne le nombre d'entrées de la file d'attente, qui n'est pas vide
QUEUE_ASNDDATA	Addr, ASNDDU	(aucune)	Met en file d'attente les données d'entrée dans la file d'attente PDODATA (addr) sur une base FIFO

7.5 Programmateur d'intervalle de temps asynchrone (ASS)

7.5.1 Vue d'ensemble

L'ASS programme l'intervalle de temps asynchrone. Il recueille les demandes de tous les CN et du MN, les met dans une file d'attente à priorité interne et décide de manière juste, du nœud qui a le droit de transmettre des données à l'intérieur du prochain intervalle asynchrone.

Cette procédure n'est disponible que sur le MN

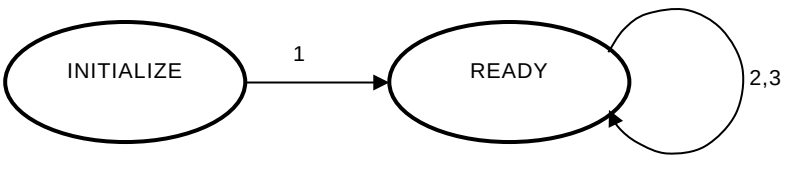
7.5.2 Définitions des primitives

7.5.2.1 Primitive échangée entre l'ASS et le CSM

La primitive et ses paramètres associés entre l'ASS et le CSM sont décrits en 7.2.2.3.

7.5.3 Table des états de l'ASS

La Figure 17 représente le diagramme états-transitions de l'ASS, et la table des états de l'ASS est présentée dans le Tableau 47.



Légende

Anglais	Français
INITIALIZE	INITIALIZE (INITIALISATION)
READY	READY (PRÊT)

Figure 17 – Diagramme états-transitions de l'ASS

Tableau 47 – Transitions de l'ASS

#	État courant	Événement /condition ⇒actions	État suivant
1	INITIALIZE	POWER-ON ou RESET / =>	READY
2	READY	CSM-Prio.ind { Source, Priority, RequestToSend } /CHECK_SOADATAQ(Source,Priority) <> "Full" && (MN = "TRUE" && Priority <= 9) || (CN = "TRUE" && Priority <= 7) => QUEUE_SOADATA(Source,Priority,RequestToSend)	READY

3	READY	CSM-SoA.req { }
		/
		=>
		Source:= SCHEDULE_SOURCE ( )
		Priority:= SCHEDULE_PRIORITY ( )
		dummy:= DEQUEUE_SOADATA(Source,Priority)
		IF Priority < 7 THEN
		RequestedServiceID:= UNSPECIFIED_INVITE
		ELSE IF Priority = 7 THEN
		RequestedServiceID:= NMT_REQUEST_INVITE
		ELSE IF Priority = 8 THEN
		RequestedServiceID:= STATUS_REQUEST
		ELSE IF Priority = 9 THEN
		RequestedServiceID:= IDENT_REQUEST
		ELSE IF Priority = 10 THEN
		RequestedServiceID:= SYNC_REQUEST
		ELSE
		RequestedServiceID:= NO_SERVICE
		ENDIF
		RequestedServiceTarget:= Source
		CSM-SoA.ind { RequestedServiceID,
		RequestedServiceTarget }
			READY

7.5.4 Fonctions de l'ASS

Une synthèse de toutes les fonctions utilisées par l'ASS est présentée dans le Tableau 48.

Tableau 48 – Table des Fonctions de l'ASS

Nom de fonction	Entrée	Sortie	Opération
QUEUE_SOADATA	Addr1 Addr2 RequestToSend	(aucune)	Met en file d'attente les données d'entrée dans la file d'attente SOADATA (addr1,addr2) sur une base FIFO
DEQUEUE_SOADATA	Addr1 Addr2	RequestToSend	Retire de la file d'attente SOADATA (addr1,addr2) sur une base FIFO
SCHEDULE_SOURCE	(aucune)	Source	Retourne l'adresse du nœud, qui obtient le droit de transmettre sur le prochain intervalle de temps asynchrone
SCHEDULE_PRIORITY	(aucune)	Priority	Retourne la priorité du nœud, qui obtient le droit de transmettre sur le prochain intervalle de temps asynchrone

7.6 Signalisation d'exception (ES)

7.6.1 Vue d'ensemble

La signalisation d'exception prend en charge l'initialisation du système de signalisation d'exception du MN aux CN et la signalisation de nouvelles exceptions des CN au MN.

7.6.2 Définitions des primitives

7.6.2.1 Primitive échangée entre l'ES et l'utilisateur de DLS

Le Tableau 49 récapitule toutes les primitives échangées entre l'ES et l'utilisateur de DLS.

Tableau 49 – Primitives échangées entre l'ES et l'utilisateur de DLS

Nom de primitive	Source	Paramètres associées	Description
DL-IERR.req	Vers la DLE	D_addr	Demande l'initialisation de la signalisation d'exception. Uniquement autorisé pour le MN
DL-IERR.cnf	À partir de la DLE	D_addr	Confirmation de l'initialisation de signalisation d'exception
DL-ERR.req	Vers la DLE		Demande la signalisation d'exception. Uniquement autorisé pour les CN
DL-ERR.ind	À partir de la DLE	S_addr	Indication de la signalisation d'exception
DL-ERR.cnf	À partir de la DLE		Confirmation de la signalisation d'exception

Les paramètres utilisés pour l'échange de primitives entre l'ES et l'utilisateur de DLS sont décrits dans le Tableau 50.

Tableau 50 – Paramètres utilisés avec les primitives échangées entre l'ES et l'utilisateur de DLS

Nom de paramètre	Description
D_addr	Le paramètre "destination-address" spécifie l'adresse de DL relative à la DLE abonnée. L'adresse globale (255) pour la diffusion et l'adresse du MN (240) n'est pas autorisée.
S_addr	Le paramètre "source-address" spécifie l'adresse de DL relative à la DLE demandée. L'adresse globale (255) pour la diffusion et l'adresse du MN (240) n'est pas autorisée.

7.6.2.2 Primitive échangée entre l'ES et l'ATC

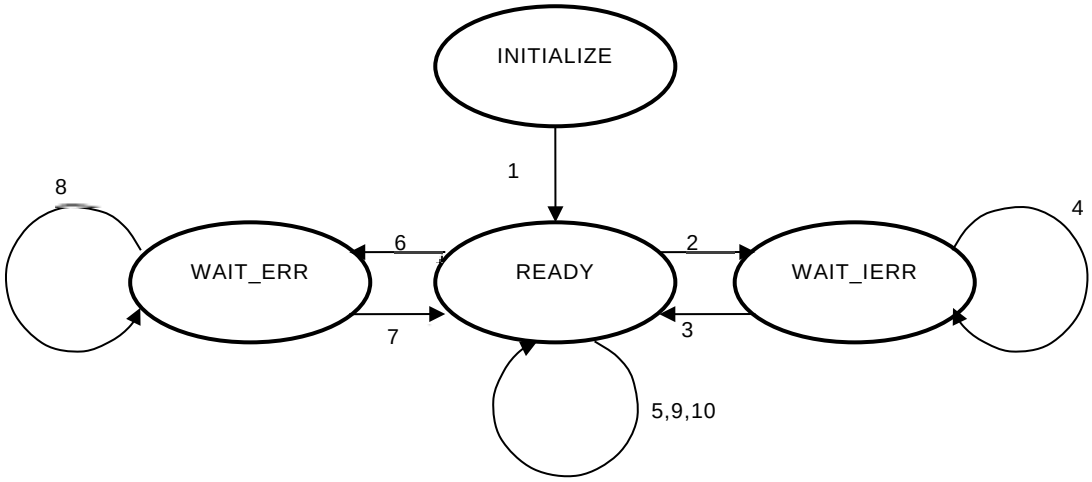
La primitive et ses paramètres associés entre l'ES et le CSM sont décrits en 7.4.2.2.

7.6.2.3 Primitive échangée entre l'ES et le CSM

La primitive et ses paramètres associés entre l'ES et le CSM sont décrits en 7.2.2.4.

7.6.3 Table des états de l'ES

La Figure 18 représente le diagramme états-transitions de l'ES, et la table des états de l'ES est présentée dans le Tableau 51.



Légende

Anglais	Français
INITIALIZE	INITIALIZE (INITIALISATION)
READY	READY (PRÊT)

Figure 18 – Diagramme états-transitions de l'ES

Tableau 51 – Transitions de l'ES

#	État courant	Événement /condition ⇒actions	État suivant
1	INITIALIZE	POWER-ON ou RESET / =>	READY
2	READY	DL-IERR.req { source } / MN = „TRUE“ => ER:= “TRUE” CSM-TERR.ind { source , ER } ES-STA.req { source }	WAIT_IERR
3	WAIT_IERR	CSM-RERR.ind {source , EC } / EC = “TRUE” && MN = “TRUE” => ER:= “FALSE” CSM-TERR.ind {source , EC } DL-IERR.cnf { source } ExceptionFlag(source) = “FALSE”	READY
4	WAIT_IERR	CSM-RERR.ind {source, EC } / EC = “FALSE” && MN = “TRUE” =>	WAIT_IERR
5	READY	CSM-RERR.ind { source , ER } / CN = “TRUE” => ExceptionFlag = “FALSE” EC:= ER CSM-TERR.req { MN , EC }	READY
6	READY	DL-ERR.req { } / CN = “TRUE” => ExceptionFlag:= !ExceptionFlag EN:= ExceptionFlag CSM-TERR.ind { source , EN }	WAIT_ERR

#	État courant	Événement /condition ⇒actions	État suivant
7	WAIT_ERR	CSM-RERR.ind {source , EA } / CN = “TRUE” && EA = ExceptionFlag => DL-ERR.cnf { }	READY
8	WAIT_ERR	CSM-RERR.ind {source, EA } / CN = “TRUE” && EA <> ExceptionFlag =>	WAIT_ERR
9	READY	CSM-RERR.ind {source, EN} / MN = “TRUE” && EN <> ExceptionFlag(source) => ExceptionFlag:= EN CSM-TERR.ind { source , EN } ES-STA.req { source } DL-ERR.ind { source }	READY
10	READY	CSM-RERR.ind {source, EN} / MN = “TRUE” && EN = ExceptionFlag(source) =>	READY

7.6.4 Fonctions de l’ES

Aucune fonction n’est utilisée.

7.7 Signalisation NMT (NS)

7.7.1 Vue d’ensemble

NS est utilisé pour transmettre le statut NMT du propre utilisateur de DLS au CSM et à tous les nœuds et pour recevoir le statut NMT de tous les nœuds.

7.7.2 Définitions des primitives

7.7.2.1 Primitive échangée entre le NS et l’utilisateur de DLS

Le Tableau 52 récapitule toutes les primitives échangées entre le NS et l’utilisateur de DLS.

Tableau 52 – Primitives échangées entre le NS et l’utilisateur de DLS

Nom de primitive	Source	Paramètres associés	Description
DL-NMT.req	Vers la DLE	S_addr	Transmet le statut NMT
DL-NMT.ind	À partir de la DLE	S_addr NMTStatus	Reçoit le statut NMT

Les paramètres utilisés pour l'échange de primitives entre le NS et l'utilisateur de DLS sont décrits dans le Tableau 53.

Tableau 53 – Paramètres utilisés avec les primitives échangées entre le NS et l’utilisateur de DLS

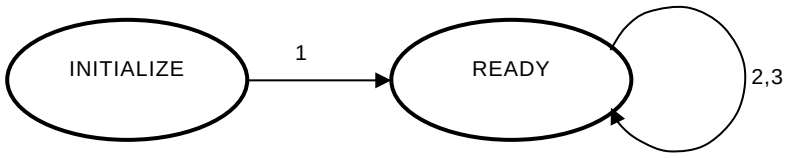
Nom de paramètre	Description
S_addr	Le paramètre “source-address” spécifie l'adresse de DL relative à la DLE publiée.
NMTStatus	Ce paramètre indique le “NMTStatus” actuel du nœud correspondant.

7.7.2.2 Primitive échangée entre le NS et le CSM

La primitive et ses paramètres associés entre le NS et le CSM sont décrits en 7.2.2.5.

7.7.3 Table des états de NS

La Figure 19 représente le diagramme états-transitions du NS, et la table des états du NS est présentée dans le Tableau 54.



Légende

Anglais	Français
INITIALIZE	INITIALIZE (INITIALISATION)
READY	READY (PRÊT)

Figure 19 – Diagramme états-transitions du NS

Tableau 54 – Transitions du NS

#	État courant	Événement /condition ⇒actions	État suivant
1	INITIALIZE	POWER-ON ou RESET / =>	READY
2	READY	DL-NMT.req { NMTStatus } / => CSM-TNMT.ind { NMTStatus }	READY
3	READY	CSM-RNMT.ind { source, NMTStatus } / => DL-NMT.ind { source, NMTStatus }	READY

7.7.4 Fonctions du NS

Aucune fonction n'est utilisée.

7.8 Protocole de gestion de la DLL

7.8.1 Vue d'ensemble

Le protocole d'interface entre la DLM et l'utilisateur de DLMS est décrit dans ce paragraphe.

7.8.2 Définitions des primitives

7.8.2.1 Primitive échangée entre la DLM et l'utilisateur de DLS

Le Tableau 55 récapitule toutes les primitives échangées entre le DLM et l'utilisateur de DLS.

Tableau 55 – Primitives échangée entre la DLM et l'utilisateur de DLS

Nom de primitive	Source	Paramètres associés	Description
DLM-Reset.req	Utilisateur de DLS	(aucun)	Cette primitive "request" entraîne la réinitialisation de la DLE par le DLMS
DLM-Reset.cnf	DLM	(sortant Status)	Ceci indique le statut de la réinitialisation
DLM-Set-value.req	Utilisateur de DLS	(entrant Variable-name, Desired-value)	Ce service est utilisé pour affecter de nouvelles valeurs aux variables de la DLE
DLM-Set-value confirm	DLM	(sortant Status)	L'utilisateur de DLMS reçoit une confirmation que la variable spécifiée a été mise à la nouvelle valeur.
DLM-Get-value.req	Utilisateur de DLS	(entrant Variable-name)	Ce service est utilisé pour lire la valeur d'une variable DLE
DLM-Get-value.cnf	DLM	(sortant Status, Current-value)	Ce service retourne la valeur réelle de la variable spécifiée
DLM-Event.ind	DLM	(sortant Event-identifier, Entry Type, Time Stamp, Additional-information)	Ce service est utilisé pour informer l'utilisateur de DLMS sur certains événements ou certaines erreurs dans la DLL
DLM-Frame.ind	DLM	(sortant DLM-frame-identifier MC PS Time Reltime)	Ce service est utilisé pour informer l'utilisateur de DLMS sur le type de trame actuelle en cours de traitement.

Les paramètres utilisés pour l'échange de primitives entre la DLM et l'utilisateur de DLS sont décrits dans le Tableau 56.

Tableau 56 – Paramètres utilisés avec les primitives échangées entre la DLM et l'utilisateur de DLS

Nom de paramètre	Description
DLM_Status	Ce paramètre permet à l'utilisateur de DLMS de déterminer si le DLMS demandé a été fourni avec succès, ou a échoué pour la raison spécifiée. La valeur transmise dans ce paramètre est comme suit: "OK – parachevé avec succès"; "Échec – arrêté avant l'achèvement"
Variable-name	Ce paramètre spécifie la variable au sein de la DLE dont la valeur doit être mise ou lue
Desired-value	Ce paramètre spécifie la valeur souhaitée pour la variable sélectionnée
Status	Ce paramètre permet à l'utilisateur de DLMS de déterminer si le DLMS demandé a été fourni avec succès, ou a échoué pour la raison spécifiée. La valeur transmise dans ce paramètre est comme suit: "OK – succès – la variable a pu être mise à jour"; "Échec – la variable n'existe pas ou n'a pas pu prendre la nouvelle valeur"; "Échec – paramètres non valides dans la demande".
Additional-information	Ce paramètre facultatif fournit des informations complémentaires spécifiques à l'événement.
DLM-frame-identifier	Ce paramètre spécifie la primitive au sein de la DLE dont l'occurrence est annoncée. Les valeurs possibles sont définies dans la partie correspondante de la CEI 61158-4-13.
MC	Ce paramètre est basculé par le MN dès que le cycle multiplexé final a été terminé.
PS	Ce paramètre est basculé par le MN chaque énième cycle. Ce signal préétalonné est utile pour les nœuds "lents" qui ne peuvent pas réagir à chaque cycle

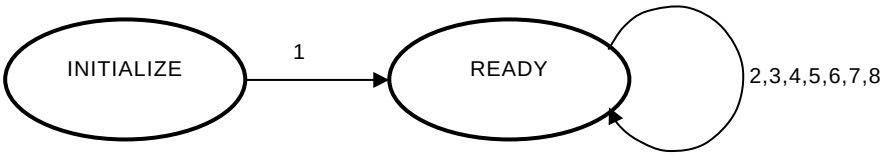
Nom de paramètre	Description
Time	Ce paramètre est distribué par le MN et indique l'heure de début du cycle de bus de terrain de Type 13
Reltime	Ce paramètre est distribué par le MN et indique le temps relatif, qui est incrémenté du temps du cycle de Type 13 lorsqu'une DLPDU SoC est générée. L'unité du temps relatif est µs
DLM-event-identifier	Ce paramètre spécifie la primitive ou l'événement composé au sein de la DLE dont l'occurrence est annoncée. Les valeurs possibles sont définies dans la partie correspondante de la CEI 61158-4-13.
EntryType	Informations de mode et de profil concernant l'erreur survenue
TimeStamp	"NetTime" à partir du cycle de Type 13 lorsque l'erreur/l'événement a été détecté(e).
AdditionalInformation	Ce champ contient des informations supplémentaires relatives aux erreurs et spécifiques à un vendeur d'appareil ou à un profil d'appareil

7.8.2.2 Primitive échangée entre la DLM et le CSM

La primitive et ses paramètres associés entre la DLM et le CSM sont décrits en 7.2.2.6.

7.8.3 Table des états de la DLM

La Figure 20 représente le diagramme états-transitions de la DLM, et la table des états de la DLM est présentée dans le Tableau 57.



Légende

Anglais	Français
INITIALIZE	INITIALIZE (INITIALISATION)
READY	READY (PRÊT)

Figure 20 – Diagramme états-transitions de la DLM

Tableau 57 – Transitions de la DLM

#	État courant	Événement /condition ⇒actions	État suivant
1	INITIALIZE	POWER-ON ou RESET / =>	READY
2	READY	DLM-Reset.req { } => RESET = "TRUE" Status:= "Success" DLM-Reset.Cnf { Status }	READY
3	READY	DLM-Set-value.req {Variable-name, Desired-value } / CHECK_VALUE (Variable-name, Desired-value) = "Valid"	READY

#	État courant	Événement /condition ⇒actions	État suivant
		=> SET_VALUE (Variable-name, Desired-value) Status:= "Success" DLM-Set-value.cnf {Status}	
4	RE ADY	DLM-Set-value.req {Variable-name, Desired-value } / CHECK_VALUE (Variable-name, Desired-value) <> "Valid" => Status:= "Failure" DLM-Set-value.cnf {Status}	READY
5	READY	DLM-Get-value.req {Variable-name } / CHECK_VAR (Variable-name) = "Valid" => Current-value:= GET_CURRENT_VAL (Variable-name) Status:= "Success" DLM-Get-value.cnf { Current-value, Status }	READY
6	READY	DLM-Get-value.req {Variable-name } / CHECK_VAR (Variable-name) <> "Valid" => Current-value:= NIL Status:= "Failure" DLM-Get-value.cnf { Current-value, Status }	READY
7	READY	CSM-Frame.ind { DLM-frame-identifier,MS,PS,Time,Reltime} / => DLM-Frame.ind { DLM-frame-identifier,MS,PS,Time,Reltime}	READY
8	READY	CSM-Event.ind { DLM-Event-identifier, EntryType, TimeStamp, Additional-Information} / => DLM-Event.ind { DLM-Event-identifier, EntryType, TimeStamp, Additional-Information}	READY

7.8.4 Fonctions de la DLM

Une synthèse de toutes les fonctions utilisées par la DLM est présentée dans le Tableau 58.

Tableau 58 – Table des Fonctions de la DLM

Nom de fonction	Description et opération
CHECK_VALUE (Variable-name, Desired-value)	Vérifie si la variable demandée avec la valeur souhaitée est valide. Les variables possibles avec la valeur sont définies en 4.5.
CHECK_VAR (Variable-name)	Vérifie si la variable demandée est valide. Les variables possibles sont définies en 4.5.
GET_CURRENT_VAL (Variable-name)	Récupère la valeur de la variable demandée
SET_VALUE	Établit la valeur de la variable demandée

Bibliographie

NOTE Toutes les parties de la série CEI 61158, ainsi que la CEI 61784-1 et la CEI 61784-2 font l'objet d'une maintenance simultanée. Les références croisées à ces documents dans le texte se rapportent par conséquent aux éditions datées dans la présente liste de références normatives.

CEI 61158-1:2014, *Réseaux de communication industriels – Spécifications des bus de terrain – Partie 1: Présentation et lignes directrices des séries CEI 61158 et CEI 61784*

CEI 61158-3-13:2014, *Réseaux de communication industriels – Spécifications des bus de terrain – Partie 3-13: Définition des services de la couche liaison de données – Eléments de type 13*

CEI 61158-5-13:2014, *Réseaux de communication industriels – Spécifications des bus de terrain – Partie 5-13: Définition des services de la couche application – Eléments de type 13*

CEI 61158-6-13:2014, *Réseaux de communication industriels – Spécifications des bus de terrain – Partie 6-13: Spécification du protocole de la couche application – Eléments de type 13*

CEI 61784-1, *Réseaux de communication industriels – Profils – Partie 1: Profils de bus de terrain*

CEI 61784-2, *Réseaux de communication industriels – Profils – Partie 2: Profils de bus de terrain supplémentaires pour les réseaux en temps réel basés sur l'ISO/CEI 8802-3*

EPSG DS 301 V1.2.0, *Ethernet POWERLINK Communication Profile Specification, Draft Standard Version 1.2.0, EPSG 2013*, disponible à l'adresse http://www.ethernet-powerlink.org/

_____

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

3, rue de Varembé
PO Box 131
CH-1211 Geneva 20
Switzerland

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