IEC 61158-4-24

Edition 1.0 2014-08

INTERNATIONAL STANDARD

NORME INTERNATIONALE

Industrial communication networks – Fieldbus specifications – Part 4-24: Data-link layer protocol specification – Type 24 elements

Réseaux de communication industriels – Spécifications des bus de terrain – Partie 4-24: Spécification du protocole de la couche liaison de données – Éléments de type 24

IEC 61158-4-24:2014-08(en-fr)

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

THIS PUBLICATION IS COPYRIGHT PROTECTED

Copyright © 2014 IEC, Geneva, Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either IEC or IEC's member National Committee in the country of the requester. If you have any questions about IEC copyright or have an enquiry about obtaining additional rights to this publication, please contact the address below or your local IEC member National Committee for further information.

Droits de reproduction réservés. Sauf indication contraire, aucune partie de cette publication ne peut être reproduite ni utilisée sous quelque forme que ce soit et par aucun procédé, électronique ou mécanique, y compris la photocopie et les microfilms, sans l'accord écrit de l'IEC ou du Comité national de l'IEC du pays du demandeur. Si vous avez des questions sur le copyright de l'IEC ou si vous désirez obtenir des droits supplémentaires sur cette publication, utilisez les coordonnées ci-après ou contactez le Comité national de l'IEC de votre pays de résidence.

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

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

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

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

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.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

IEC 61158-4-24

Edition 1.0 2014-08

INTERNATIONAL STANDARD

NORME INTERNATIONALE

Industrial communication networks – Fieldbus specifications – Part 4-24: Data-link layer protocol specification – Type 24 elements

Réseaux de communication industriels – Spécifications des bus de terrain – Partie 4-24: Spécification du protocole de la couche liaison de données – Éléments de type 24

INTERNATIONAL ELECTROTECHNICAL COMMISSION

COMMISSION ELECTROTECHNIQUE INTERNATIONALE

PRICE CODE
CODE PRIX XE

ICS 25.040.40; 35.100.20; 35.110

ISBN 978-2-8322-1730-6

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

® Registered trademark of the International Electrotechnical Commission
Marque déposée de la Commission Electrotechnique Internationale

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

CONTENTS

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

INTERNATIONAL ELECTROTECHNICAL COMMISSION

INDUSTRIAL COMMUNICATION NETWORKS – FIELDBUS SPECIFICATIONS –

Part 4-24: Data-link layer protocol specification – Type 24 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.

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 61158-4-24 has been prepared by subcommittee 65C: Industrial networks, of IEC technical committee 65: Industrial-process measurement, control and automation.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

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 the ISO/IEC Directives, Part 2.

A list of all parts of the IEC 61158 series, published 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.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

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.

NOTE Use of some of the associated protocol types is restricted by their 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 particular data-link layer protocol type to be used with physical layer and application layer protocols in Type combinations as specified explicitly in the profile series. Use of the various protocol types in other combinations may require permission from their respective intellectual-property-right holders.

The International Electrotechnical Commission (IEC) draws attention to the fact that it is claimed that compliance with this document may involve the use of patents as follows, where the [xx] notation indicates the holder of the patent right:

US 8223804 JP 4760978 CN 200880002225.3 EPC 08738862.5 KR 10-2009-7011514 TW 97111183	[YE]	COMMUNICATION DEVICE, SYNCHRONIZED SYSTEM, AND SYCHRONIZED COMMUNICATION METHOD
US 7769935 JP 4683346 US 8046512 EPC 07850686.2 TW 96150287	[YE]	MASTER SLAVE COMMUNICATION SYSTEM AND MASTER SLAVE COMMUNICATION METHOD
JP 4356698	[YE]	COMMUNICATION DEVICE, SYNCHRONIZED COMMUNICATION SYSTEM, AND SYCHRONIZED COMMUNICATION METHOD

IEC takes no position concerning the evidence, validity and scope of this patent right.

The holders of these patent rights have assured IEC that they are willing to negotiate licenses either free of charge or under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holders of these patent rights is registered with IEC. Information may be obtained from

[YE] YASKAWA ELECTRIC CORPORATION
2-1 Kurosakishiroishi, Yahatanishi-ku, Kitakyushu 806-0004, Japan
Attention; Intellectual Property Rights Section.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those identified above. IEC shall not be held responsible for identifying any or all such patent rights.

ISO (www.iso.org/patents) and IEC (http://patents.iec.ch) maintain on-line data bases of patents relevant to their standards. Users are encouraged to consult the data bases for the most up to date information concerning patents.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

INDUSTRIAL COMMUNICATION NETWORKS – FIELDBUS SPECIFICATIONS –

Part 4-24: Data-link layer protocol specification – Type 24 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, or

b) in an acyclic manner, as requested 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 datalink 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 64 ms;

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 standard 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.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

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.

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 61158-2, *Industrial communication networks – Fieldbus specifications – Part 2: Physical layer specification and service definition*

IEC 61158-3-24:2014, *Industrial communication networks – Fieldbus specifications – Part 3-24: Data-link layer service definition – Type 24 elements*

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 9899, *Information technology – Programming languages – C*

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

ISO/IEC 13239:2002, *Information technology – Telecommunications and information exchange between systems – High-level data link control (HDLC) procedures*

ISO/IEC 19501:2005, *Information technology – Open Distributed Processing – Unified Modeling Language (UML) Version 1.4.2*

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

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 **acknowledgement** [ISO/IEC 7498-1]

3.1.2 **correspondent (N)-entities** [ISO/IEC 7498-1]

correspondent DL-entities (N=2)

correspondent Ph-entities (N=1)

3.1.3 **DL-address** [ISO/IEC 7498-3]

3.1.4 **DL-protocol** [ISO/IEC 7498-1]

3.1.5 **DL-protocol-data-unit** [ISO/IEC 7498-1]

3.1.6 **DL-service-data-unit** [ISO/IEC 7498-1]

3.1.7 **DLS-user** [ISO/IEC 7498-1]

3.1.8 **DLS-user-data** [ISO/IEC 7498-1]

3.1.9 **event** [ISO/IEC 19501]

3.1.10 **layer-management** [ISO/IEC 7498-1]

3.1.11 **primitive name** [ISO/IEC 7498-1]

3.1.12 **reset** [ISO/IEC 7498-1]

3.1.13 **segmenting** [ISO/IEC 7498-1]

3.1.14 **state** [ISO/IEC 19501]

3.1.15 **state machine** [ISO/IEC 19501]

3.1.16 **systems-management** [ISO/IEC 7498-1]

3.1.17 **transition** [ISO/IEC 19501]

3.1.18 **(N)-entity** [ISO/IEC 7498-1]

DL-entity (N=2)

Ph-entity (N=1)

3.1.19 **(N)-layer** [ISO/IEC 7498-1]

DL-layer (N=2)

Ph-layer (N=1)

(N)-service [ISO/IEC 7498-1]

DL-service (N=2)

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Ph-service (N=1)

(N)-service-access-point [ISO/IEC 7498-1]

DL-service-access-point (N=2)

Ph-service-access-point (N=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 confirm (primitive)
3.2.2 DL-service-primitive;
3.2.3 DL-service-provider
3.2.4 DL-service-user
3.2.5 indication (primitive)
3.2.6 request (primitive)
3.2.7 requestor
3.2.8 response (primitive)

3.3 Common terms and definitions

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

3.3.1
acyclic transmission
non-periodic exchange of telegrams

3.3.2
C1 master
one of the station type that initiates and control cyclic transmission

3.3.3
C1 message
message communication that C1 master operates as initiator to exchange messages with slave or C2 master

3.3.4
C2 master
one of the station type that has the function of monitoring all process data transmitted through the network and may initiates message communication

3.3.5
C2 message
message communication that C2 master operates as initiator to exchange messages with slave or C1 master

3.3.6
cyclic transmission
periodic exchange of telegrams

3.3.7
data
generic term used to refer to any information carried over a fieldbus

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

3.3.8
device
physical entity connected to the fieldbus composed of at least one communication element (the network element) and which may have a control element and/or a final element (transducer, actuator, etc.)

3.3.9
event driven mode
transmission mode for the application layer protocol of the communication type 24 in which a transaction of command-response-exchanging arises as user's demands

3.3.10
frame
synonym for DLPDU

3.3.11
initiator
station that initiates the exchange of process data or message

3.3.12
interface
shared boundary between two functional units, defined by functional characteristics, signal characteristics, or other characteristics as appropriate

3.3.13
input data
process data sent by the slave and received by the C1 master

3.3.14
message
ordered series of octets intended to convey information

Note 1 to entry: Normally used to convey information between peers at the application layer.

3.3.15
monitor slave
slave that has the function of monitoring all process data transmitted through the network

3.3.16
network
set of nodes connected by some type of communication medium, including any intervening repeaters, bridges, routers and lower-layer gateways

3.3.17
node
a) single DL-entity as it appears on one local link

b) end-point of a link in a network or a point at which two or more links meet

3.3.18
output data
process data sent by the C1 master and received by the slaves

3.3.19
protocol
convention about the data formats, time sequences, and error correction in the data exchange of communication systems

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

3.3.20
real-time communication
transfer of data in real-time

3.3.21
receiving DLS-user
DL-service user that acts as a recipient of DL-user-data

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

3.3.22
responder
station that responds process data or message after it has been initiated by initiator

3.3.23
send data with acknowledge
data transfer service with acknowledge of reception from corresponding DLE

3.3.24
send data without acknowledge
data transfer service without acknowledge of reception from corresponding DLE

3.3.25
slave
one of the station type that accesses the medium only after it has been initiated by C1-maser or C2 master

3.3.26
sending DLS-user
DL-service user that acts as a source of DL-user-data

3.3.27
station
node

3.3.28
topology
physical network architecture with respect to the connection between the stations of the communication system"

3.3.29
transmission cycle
fixed time period of cyclic transmission

3.3.30
time slot
Time period reserved so that initiator and responder may exchange one frame respectively

3.4 Symbols and abbreviations

3.4.1	**DA**	Destination address
3.4.2	**DL-**	Data-link layer (as a prefix)
3.4.3	**DLE**	DL-entity (the local active instance of the data-link layer)
3.4.4	**DLL**	DL-layer
3.4.5	**DLM**	DL-management

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

3.4.6	**DLME**	DL-management entity (the local active instance of DL-management)
3.4.7	**DLMS**	DL-management service
3.4.8	**DLPDU**	DL-protocol-data-unit
3.4.9	**DLS**	DL-service
3.4.10	**DLSAP**	DL-service-access-point
3.4.11	**DLSDU**	DL-service-data-unit
3.4.12	**FIFO**	First-in first-out (queuing method)
3.4.13	**ID**	Identifier
3.4.14	**OSI**	Open systems interconnection
3.4.15	**PDU**	Protocol data unit
3.4.16	**Ph-**	Physical layer (as a prefix)
3.4.17	**PhE**	Ph-entity (the local active instance of the physical layer)
3.4.18	**PhL**	Ph-layer
3.4.19	**PHY**	Physical layer device (specified in ISO/IEC 8802-3)
3.4.20	**QoS**	Quality of service
3.4.21	**RT**	Real-time
3.4.22	**SAP**	Service access point
3.4.23	**SDU**	Service data unit

3.5 Additional type 24 symbols and abbreviations

3.5.1	**ACK**	Acknowledge
3.5.2	**C1MSG**	C1 message
3.5.3	**C2MSG**	C2 message
3.5.4	**I/O**	Input and/or output
3.5.5	**MSG**	Message
3.5.6	**Rx**	Receive
3.5.7	**SDA**	Send data with acknowledge
3.5.8	**SDN**	Send data without acknowledge
3.5.9	**SM**	State machine
3.5.10	**Tcycle**	Transmission cycle
3.5.11	**Tslot**	Time slot
3.5.12	**Tx**	Transmit

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

3.6 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 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 is mandatory for the primitive.

U parameter is 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 parameter-specific constraint:

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

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

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.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

3.7 Additional Type 24 conventions

3.7.1 Primitive conventions

The following notation, a shortened form of the primitive classes defined in 3.2, is used in the figures.

req request primitive

ind indication primitive

cnf confirm primitive (confirmation)

3.7.2 State machine conventions

The protocol sequences are described by means of state machines.

In state diagrams, states are represented as boxes and state transitions are shown as arrows.

Names of states and transitions of the state diagram correspond to the names in the state table. The textual listing of the state transitions is structured as shown in Table 1.

Table 1 – State transition descriptions

No.	Current state	Event /condition =>action	Next state

The description of state machine elements are shown in Table 2.

Table 2 – Description of state machine elements

Description element	Meaning
No	Number of the transition.
Current state, Next state	Names of the originating state and the target state of transition.
Event	Name or description of the trigger event that fire the transition.
/ conditions	Boolean expression, which must be true for the transition to be fired.
=>action	List of assignments and service or function invocations. The action should be atomic. The preceding “=>” is not part of the action.
NOTE “/ conditions” can be omitted.	

The conventions used in the state machines are shown in Table 3.

Table 3 – Conventions used in state machines

Convention	Meaning
+ - * /	Arithmetic operators
:=	Value of an item on the left is replaced by value of an item on the right. If an item on the right is a parameter, it comes from the primitive shown as an input event.
=	A logical condition to indicate an item on the left is equal to an item on the right.
<	A logical condition to indicate an item on the left is less than the item on the right.
>	A logical condition to indicate an item on the left is greater than the item on the right.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Convention	Meaning
<=	A logical condition to indicate an item on the left is less than or equal to the item on the right.
>=	A logical condition to indicate an item on the left is greater than or equal to the item on the right.
<>	A logical condition to indicate an item on the left is not equal to an item on the right.
&&	Logical "AND"
||	Logical "OR"

4 Overview of DL-protocol

4.1 Characteristic feature of DL-protocol

Table 4 shows the characteristic features of DL protocol of Type 24.

Table 4 – Characteristic features of the fieldbus data-link protocol

Profiles	Description
Station type and max. stations	-C1 master (active station with bus access control, 1 station (mandatory)) -C2 master (active station with restricted bus access control, max.1 (optional)) -Slave (passive stations without bus access control, max.62)
Station addressing	1 to 255 (255 = global addresses for broad-cast messages), 8 bit-width address extension for integrated device
Transmission cycle	31,25 us to 64 ms
DLSDU size	8 to 64 octets
Transmission characteristic	-Cyclic data exchange and cyclic event, synchronized with accurate cycle time (jitter below 1 us) -Max 62 times (n times/1 station) retry within cycle time -Acyclic message transmission

There are three types of stations, C1 master, C2 master and slave. Data exchange is executed between one master station (C1 master or C2 master) and N slave stations. This protocol supports 2 communication modes, cyclic transmission and acyclic transmission.

In cyclic transmission mode, transmission is executed cyclically with an accurate period. The transmission cycle is set by the C1 master to a value within a range of 31,25 [μs] to 64 [ms]. Since the set value of transmission cycle is specific to the transmission line, all of the connected slaves shall support that value. It is not permitted to set different transmission cycle values for slaves connected in the same network.

The transmission cycle has I/O data exchange band to transmit process data and message communication band to transmit message. The protocol machine in C1 master controls transmission sequence in cyclic transmission mode. The time period for a master station to exchange with one slave station is called time slot. There are two types of communication sequence, one is “fixed-width time slot type” whose time slot is same width for all stations and the other is “configurable time slot type” whose time slot can be defined for each station. All stations shall use the same-data-length frame when fixed-width time slot type. The width of the time slot is static in both type, and the value is set by DL-management during initialization. Once cyclic communication starts, it shall not be changed.

Acyclic transmission mode is used by DLS-user that operates in event driven mode. In acyclic transmission mode, transmissions are executed sporadically. The same transmission sequence and message communication may be executed in acyclic transmission, as in cyclic transmission mode without fixing the transmission cycle.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

4.2 DL layer component

DL layer is composed of three sublayers, CTC (Cyclic transmission control), SRC (Send Receive Control) and DLM (Data-link management). SRC is positioned at lower layer of CTC and DLM covers both CTC sublayer and SRC sublayer. The data-link layer component is show in Figure 1.

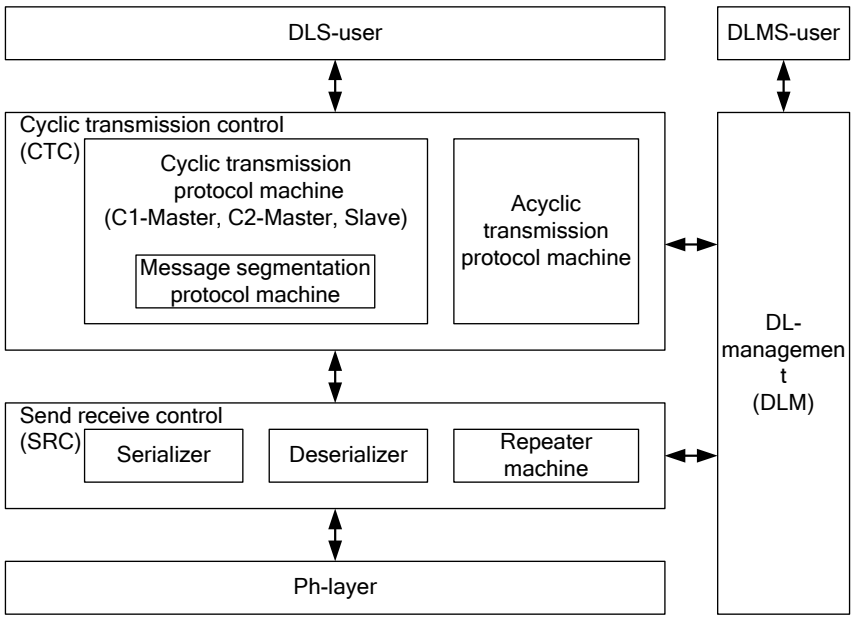


Figure 1 – Data-link layer component

4.2.1 Cyclic transmission control (CTC)

This is a sublayer that builds DLPDU and executes a protocol machine. It has 2 communication modes, i.e. cyclic transmission mode and acyclic transmission mode. CTC executes either of them according to a request from DLMS user.

4.2.2 Send Receive Control (SRC)

SRC sends or receives frames by request of CTC sublayer. It is serialized or de-serialized according to corresponding PHY. When the SRC implements two or more PHY port, the SRC provides frame repeat function between the implemented PHY ports.

4.2.3 DL-management

This is a sublayer that configures DLE operation by setting the internal variables and manages errors detected by each sublayer.

4.3 Timing sequence

4.3.1 Overview

There are two types of transmission mode, cyclic transmission mode and acyclic transmission mode. Cyclic transmission has two types, one is “fixed-width time slot type” whose time slot is same width for all stations and the other is “configurable time slot type” whose time slot can be defined for each station.

The width of the time slot is static in both type, and the value is set by DL-management during initialization. Once cyclic communication starts, it shall not be changed.

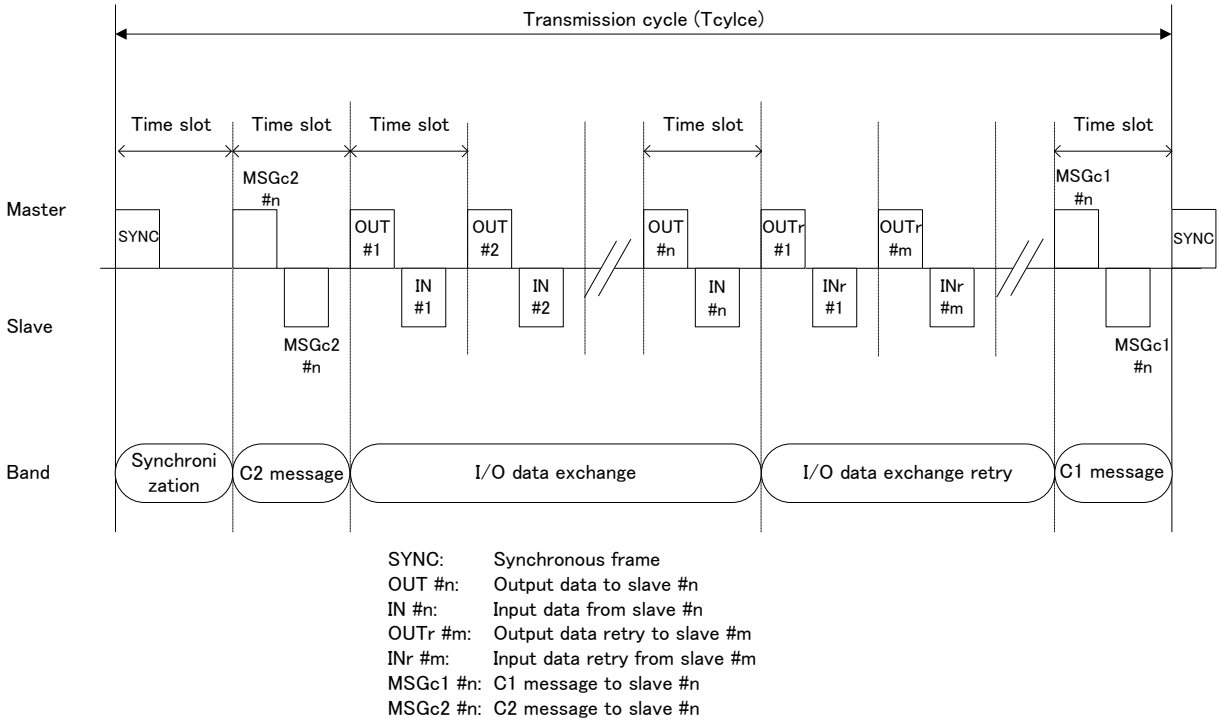
Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

4.3.2 Cyclic transmission mode

4.3.2.1 Fixed-width time slot type

4.3.2.1.1 Overview

Figure 2 shows transmission sequence of fixed-width time slot type. In this type, the width of time slots that are allocated to execute exchange of process data one by one between the master and the slave is identical for all slaves. One or more time slots are allocated to each bandwidth shown in the timing chart.



SYNC: Synchronous frame
OUT #n: Output data to slave #n
IN #n: Input data from slave #n
OUTr #m: Output data retry to slave #m
INr #m: Input data retry from slave #m
MSGc1 #n: C1 message to slave #n
MSGc2 #n: C2 message to slave #n

Figure 2 – Timing chart of fixed-width time slot type cyclic communication

4.3.2.1.2 Detailed description of communication band

4.3.2.1.2.1 Synchronization

This is a bandwidth through which C1 master broadcasts a synchronous frame to slave and C2 master. One time slot is allocated to this bandwidth. Within this bandwidth, only the transition of synchronizing frame from C1 master is allowed; slave and C2 master are prohibited transmitting any frame.

4.3.2.1.2.2 C2 message

This is a bandwidth for a message transmission (C2 message transmission) where C2 master is the client (primary station) and C1 master or slave is the server (secondary station). One time slot is allocated to this bandwidth, and request and response are transmitted once, respectively.

4.3.2.1.2.3 I/O data exchange

This is a bandwidth through which C1 master exchanges I/O data with all slaves that are connected to the network. The time slots of the number of slaves are assigned to this bandwidth. C1 master and one slave station execute I/O data exchange once within one time slot.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

C1 master registers the slave with which it has failed to exchange I/O data through this bandwidth into the retry list as a retry target for re-transmission through a following I/O data exchange retry bandwidth.

4.3.2.1.2.4 I/O data exchange retry

This is a bandwidth through which C1 master retries the I/O data exchange that has not been completed successfully through the I/O data exchange bandwidth. C1 master re-executes the I/O exchange with the retry target slave that has been registered in the I/O data exchange bandwidth. Time slots of the number that is set in C1 master before starting cyclic transmission are allocated to this bandwidth. C1 master retries according to the registered order of the retry list for up to the number of the allocated time slots. DLE quits retry when it uses all allocated time slots even if a slave that is waiting for retry is registered in the retry list.

C1 master retries the I/O data exchange once for each of the registered slave. If the retry is not completed successfully, C1 master will not repeat the retry for the same slave.

4.3.2.1.2.5 C1 message

This is a bandwidth for a message transmission (C1 message transmission) where C1 master is the client (initiator) and C2 master or slave is the server (responder). One time slot is allocated at the maximum within the bandwidth that is allocated to the retry of the I/O data exchange mentioned above, and request and response are transmitted once, respectively. Because one of the bandwidths for retry of I/O data exchange is allocated as this bandwidth, when all of the time slots allocated for retry are used, C1 message can not be executed within the transmission cycle.

4.3.2.1.3 Estimation of cycle time

The transmission cycle of the fixed-width slot type T_{cycle} is calculated as the sum of the bandwidths described in 4.3.2.2, i.e.:

$$T_{cycle} = T_{sync} + T_{C2msg} + T_{io} + T_{retry} + T_{C1msg}$$

where

T_{sync} is the Sync band;
T_{C2msg} is the C2 message band;
T_{io} is the I/O data exchange band;
T_{retry} is the I/O data exchange retry band;
T_{C1msg} is the C1 message band.

An integral multiple of the number of time slots is allocated to the bandwidths. The formula shown above can be transformed by indicating the time slot with T_{slot}, the number of slave stations connected to the network with n, and the number of retry with n_r.

$$T_{cycle} = T_{\text{slot}} + T_{\text{slot}} + n \times T_{\text{slot}} + n_r \times T_{\text{slot}} + T_{\text{slot}}$$

$$= (n + n_r + 3) \times T_{\text{slot}}$$

Time slot T_{slot} can be calculated as shown in the following formula by indicating the frame transmission time (identical to the instruction frame, answer frame, request frame, and response frame) with T_{tr} and the gap between frames with T_{gap}:

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

$$T_{slot} = \max(T_{tr\_c}(n) + T_{dly}(n) + T_{gap} + T_{tr\_r}(n) + T_{dly}(n) + T_{gap})$$
$$= 2 \times \max(T_{tr\_c}(n) + T_{dly}(n) + T_{gap})$$

T_{tr_c} is the instruction transmission time from C1 master to the slave n;

T_{tr_r} is the response transmission time from slave n to C1 master;

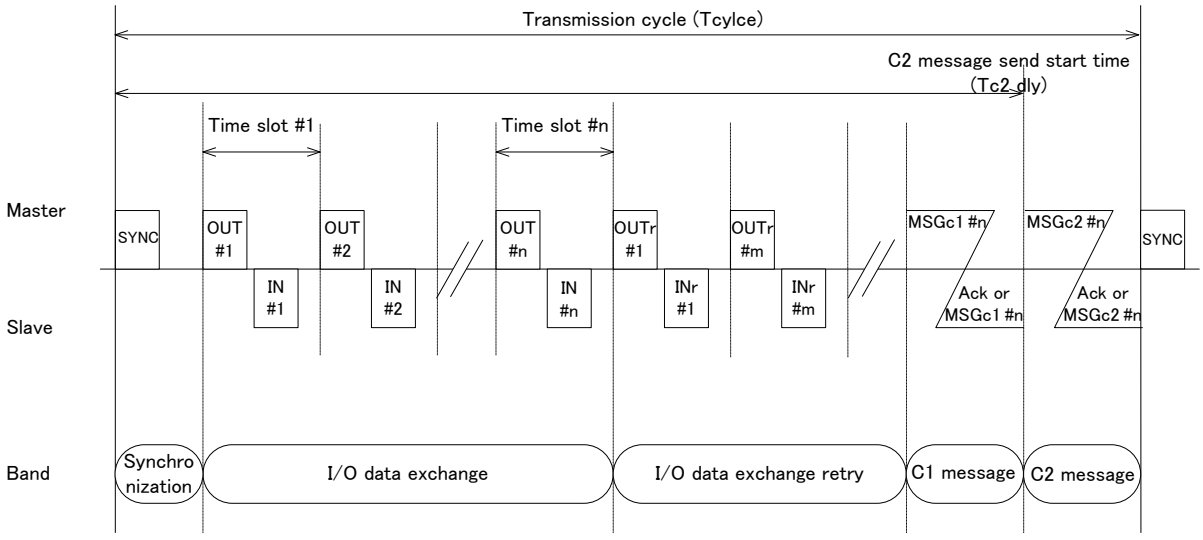
T_{gap} is the gap between the frames;

T_{dly} is the frame transmission delay time between C1 master and slave n.

4.3.2.2 Configurable time slot type

4.3.2.2.1 Overview

Figure 3 shows transmission sequence of configurable time slot type. In this type, the length of time slots that are allocated to execute exchange of command and response one by one between the master and the slave is differ for each slave. DLE manages the residual time of the transmission cycle by using the time slots configured for each slave by the DLMS user. Details of each transmission bandwidth are described in the following subclauses.



SYNC: Syncronous frame
OUT #n: Output data to slave #n
IN #n: Input data from slave #n
OUTr #m: Output data retry to slave #m
INr #m: Input data retry from slave #m
MSGc1 #n: C1 message to slave #n
MSGc2 #n: C2 message to slave #n

Figure 3 – Timing chart of configurable time slot type cyclic communication

4.3.2.2.2 Detailed description of communication phase

4.3.2.2.2.1 Synchronization

See 4.3.2.1.2.1.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

4.3.2.2.2.2 IO data exchange

This is a bandwidth where C1 master executes I/O data exchange with all slaves connected to the network. The time from the end of Synchronization bandwidth in the head of the transmission cycle to the start of the C2 message is allocated for the bandwidth that aggregates this bandwidth, the succeeding I/O data exchange retry bandwidth, and C1 message bandwidth. C1 master and one slave station execute I/O data exchange once within one time slot.

C1 master registers the slave that fails I/O data exchange within this bandwidth into the retry list as a re-transmission target within the succeeding I/O data exchange retry bandwidth.

4.3.2.2.2.3 IO data exchange retry

This band is basically same as the case of fixed-width time slot (see 4.3.2.1.2.4). Subclause 4.3.2.2.2.3 describes the differences.

The time from the end of Synchronization bandwidth in the head of the transmission cycle to the start of the C2 message is allocated for the bandwidth that aggregates the I/O data exchange bandwidth, this bandwidth, and C1 message bandwidth. C1 master executes the retry for the slave registered in the retry list in the order of registration, and when the retry succeed, clears the registration. At the same time, C1 master compares the time required to complete the I/O data exchange with the slave and the residual time until the bandwidth ends (until C2 message starts), then executes the retry if the residual time is longer. If the residual time is shorter than the required time, C1 master ends this bandwidth.

C1 master executes the retry for the slave registered in the retry list in the order of registration, and when the retry succeed, clears the registration. When the bandwidth ends before executing the retries for all of the retry targets, C1 master quits retry.

In the case when the retry executed for a registered slave does not completed successfully, C1 master registers the slave again at the end of the retry list. C1 master repeats retry for the identical slave within the residual time of the bandwidth. When the retries for all of the retry targets within the retry list is executed, C1 master retrieves the slave from the retry list and then execute the retry again. C1 master ends the bandwidth when all of the slaves are cleared from the retry list.

4.3.2.2.2.4 C1 message

This is a bandwidth for a message transmission (C1 message transmission) where C1 master is the client (primary station) and C2 master or slave is the server (secondary station). Residual time from the end of the I/O data exchange retry bandwidth to the start of C2 message transmission is allocated to this bandwidth. C1 master executes the C1 message transmission within the allocated bandwidth. C1 master can repeat the transmission that consists of one request and one response as a pair within this bandwidth. However, C1 master can not execute C1 message transmission if there is no residual time enough for one transmission when the I/O data exchange retry bandwidth ends.

4.3.2.2.2.5 C2 message

This is a bandwidth for a message transmission (C2 message transmission) where C2 master is the client (primary station) and C1 master or slave is the server (secondary station). Time from the start of C2 message to the end of the transmission cycle is allocated to this bandwidth. C2 master executes the C2 message transmission within the allocated bandwidth. C2 master can repeat the transmission that consists of one request and one response as a pair within the bandwidth.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

4.3.2.2.3 Estimation of cycle time

The operation of multiple slaves is synchronized with the command sent by the C1 master. To enable this, the transmission delay time of each slave is measured during initialization. The measured delay time is retained by the C1 master and each slave.

Based on the measured transmission delay time, the C1 master calculates the response monitoring time and the interrupt delay time for each slave to match the communication interrupt timing in the system.

The interrupt delay time is delivered with the synchronous frame (SYN frame in the figure). C1 master and each slave generates communication interrupt timing according to the interrupt delay time and the transmission delay time retained in the local station. As the result, communication interrupt occurs at the same time throughout the system.

By executing data reception processing simultaneously at all slaves at the cyclic event timing, the system can operate synchronously.

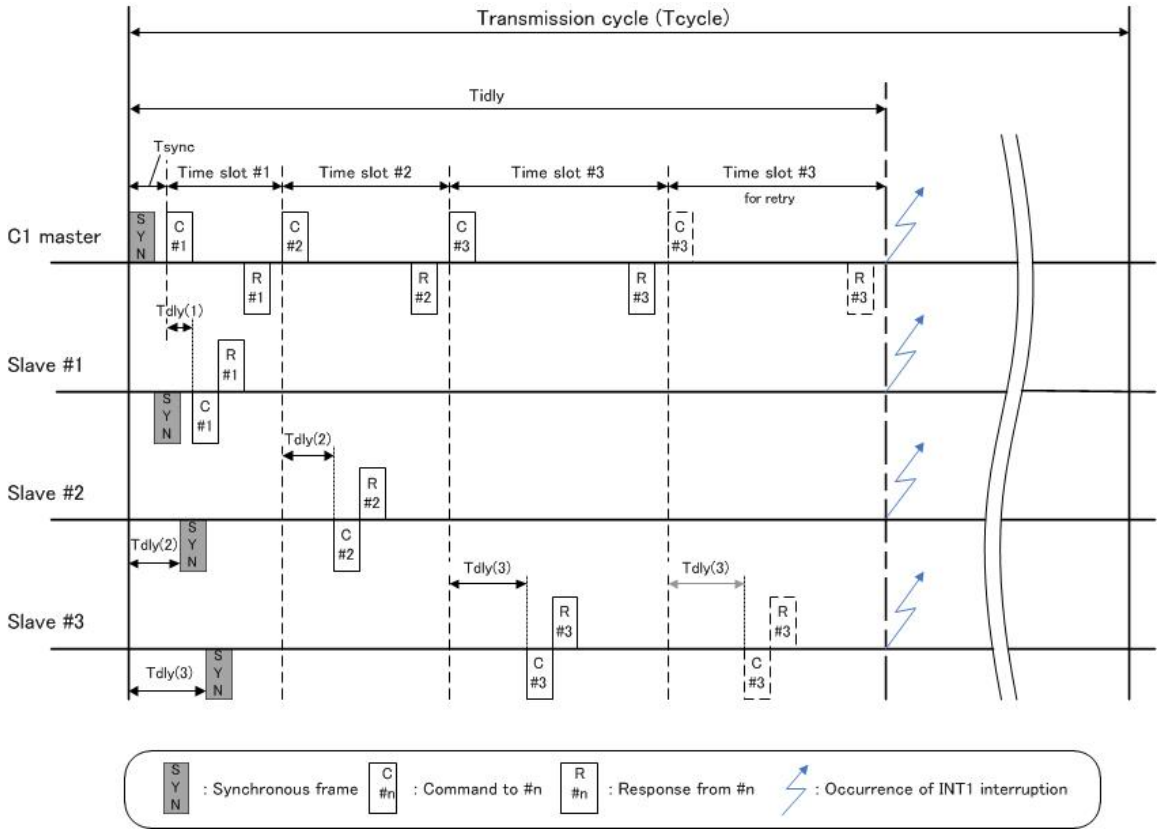


Figure 4 – Schematic Diagram of Communication Interrupt Occurrence

The transmission cycle of variable time slot type T_{cycle} is calculated as an aggregation of the bandwidth described in 4.3.2.3.2, as shown in the following formula:

$$T_{cycle} = T_{sync} + T_{io} + T_{retry} + T_{C1msg} + T_{C2msg}$$

where

T_{sync} is the Sync band;

T_{io} is the I/O data exchange band;

T_{retry} is the I/O data exchange retry band;

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

T_{c1msg} is the C1 message band;

T_{c2msg} is the C2 message band.

The calculation method for the bandwidths mentioned above is shown in the followings.

a) Sync band

Sync bandwidth T_{sync} is calculated as follows: where T_{tr_s} is the transmission time of synchronous frame, and T_{gap} is the gap between the frames:

$$T_{\text{sync}} = T_{\text{tr\_s}} + T_{\text{gap}}$$

b) I/O data exchange band

I/O data exchange bandwidth Tio is calculated as follows: where N is the number of slave connections, $T_{tr\_c}(n)$ is the instruction transmission time from C1 master to the slave of the number n, $T_{dly}(n)$ is the frame transmission delay time between C1 master and the slave of the number n, and T_{gap} is the gap between the frames:

$$T_{io} = \sum_{n} \left\{ T_{tr\_c}(n) + T_{dly}(n) + T_{gap} + T_{tr\_r}(n) + T_{dly}(n) + T_{gap} \right\}$$

$$= \sum_{n} \left\{ T_{tr\_c}(n) + T_{tr\_r}(n) + 2 \times T_{dly}(n) \right\} + 2 \times N \times T_{gap}$$

c) I/O data exchange retry band

I/O data exchange bandwidth Tretry is calculated as follows: where N_r is the number of retry, $T_{tr\_c}(r)$ instruction transmission time from C1 master to the slave of the number r, $T_{dly}(r)$ is the frame transmission delay time between C1 master and the slave of the number r, and Tgap is the gap between the frames:

$$T_{retry} = \sum_{ir} \left\{ T_{tr\_c}(r) + T_{dly}(r) + T_{gap} + T_{tr\_r}(r) + T_{dly}(r) + T_{gap} \right\}$$

$$= \sum_{r} \left\{ T_{tr\_c}(r) + T_{tr\_r}(r) + 2 \times T_{dly}(r) \right\} + 2 \times N_r \times T_{gap}$$

d) C1 message band

C1 message bandwidth T_{c1msg} is calculated as follows: $T_{tr\_c1c}(m_1)$ is the request transmission time from the primary station (C1 master) to the secondary station m_1 (m_1 is the station number of slave or C2 master that becomes the secondary station), $T_{tr\_c1r}(m_1)$ is the response transmission time from the secondary station to the primary station, $T_{dly}(m_1)$ is the transmission delay between the primary station and the secondary station, N_{c1msg} is the number of the C1 messages, and T_{gap} is the gap between the frames:

$$T_{C1msg} = N_{c1msg} \times \left\{ T_{tr\_c1c}(m_1) + T_{dly}(m_1) + T_{gap} + T_{tr\_c1r}(m_1) + T_{dly}(m_1) + T_{gap} \right\}$$

$$= N_{c1msg} \times \left\{ T_{tr\_c1c}(m_1) + T_{tr\_c1r}(m_1) + 2 \times T_{dly}(m_1) + 2 \times T_{gap} \right\}$$

e) C2 message band

C2 message bandwidth T_{c2msg} is calculated as follows: $T_{tr\_c2c}(m_2)$ is the request transmission time from the primary station (C2 master) to the secondary station m_2 (m_2 is the station number of slave or C1 master that becomes the secondary station), $T_{tr\_c2r}(m_2)$ is the response transmission time from the secondary station to the primary station, $T_{dly}(m_2)$ is the

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

transmission delay between the primary station and the secondary station, N_{c2msg} is the number of the C2 messages, and T_{gap} is the gap between the frames:

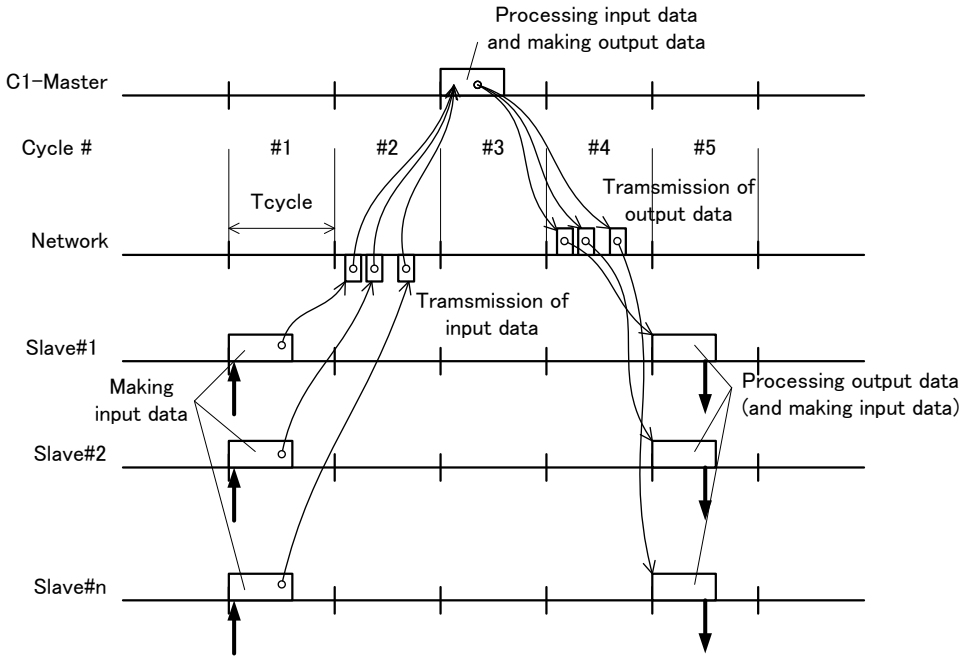
$$T_{c2msg} = N_{c2msg} \times \{T_{tr\_c2c}(m_2) + T_{dly}(m_2) + T_{gap} + T_{tr\_c2r}(m_2) + T_{dly}(m_2) + T_{gap}\}$$

$$= N_{c2msg} \times \{T_{tr\_c2c}(m_2) + T_{tr\_c2r}(m_2) + 2 \times T_{dly}(m_2) + 2 \times T_{gap}\}$$

4.3.2.3 Timing relationship between cyclic transmission and data processing

This subclause explains timing relationship between cyclic transmission and data processing by using Figure 5. In cycle #1, the slaves latch input and make the input data to be sent. The input data that the each slave made is transmitted to C1 master in cycle #2. Though it is received by C1 master, it is not processed by C1 master at this time. C1 master starts to process it at the top of the cycle #3. Therefore, the delay from the timing of slave's latched input to the timing of the master processing it is two transmission cycle.

Similarly, the output data that the C1 master made at cycle #3 is transmitted to all slaves, slave by slave, in cycle #4. Though it is received by each slave in cycle #4, it is not processed by the slave at this time. All slaves start to process it all together at the top of cycle #5. Therefore, the delay from the timing of master's making output data to the timing of the slave processing it is two transmission cycles, that is same as input data.



NOTE Output data and input data are transmitted in every cycle, but these drawings are omitted in this figure to explain easily. Data processing by C1 master and slaves in every cycle are also omitted.

Figure 5 – Timing relationship between cyclic transmission and data processing

4.3.3 Acyclic transmission mode

Acyclic transmission may be used in a system that does not require real-time communication or cyclical data exchange. In acyclic transmission mode, it is possible to execute the same transmission sequence as cyclic transmission mode without fixing the transmission cycle. Although C2 message communication is also possible, the DLS-user shall execute the arbitration of the transmission timing. In acyclic transmission mode, the data length is fixed at 64 octets.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Since acyclic transmission may not use the synchronous frame, slaves executes processing of the output data sent by the master and processing of the data to send the input data at its own timing. (Slaves do not operate simultaneously.)

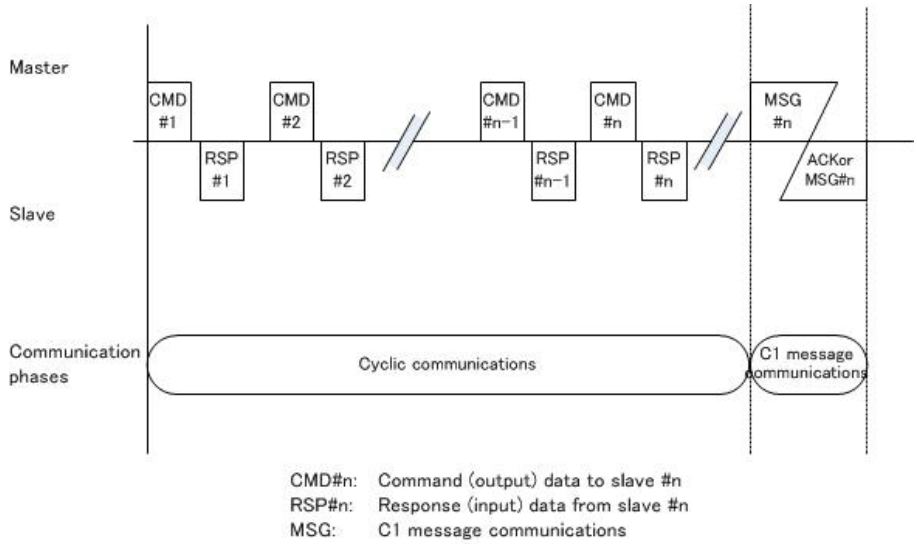


Figure 6 – Timing chart example of acyclic communication

4.4 Service assumed from PhL

4.4.1 General requirement

There are two types of Ph-service which DLE required. One is defined in IEC 61158-2, Type 24 and the other is defined in ISO/IEC 8802-3, Clause 6.

Both PHY requires the following interface elements;

- Transmit Machine
- Receive Machine
- Serializer
- Deserializer

4.4.2 DL_Symbols

The PhL Interface Data Units present at the DLL-PhL interface shall be DL_symbols. DL_symbol shall have one of the following values:

a) ZERO corresponds to a binary "0";

b) ONE corresponds to a binary "1".

4.4.3 Assumed primitives of PhS

The PhS is assumed to provide the following two categories of primitives to the Type 24 DL-protocol.

a) Service primitives for transmitting and receiving frames to / from other peer DLEs.

b) Service primitives that provide information needed by the local DLE to perform the media access functions.

The assumed primitives of PhS are grouped into these two categories:

c) Transfer of Data to the corresponding DLE

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

1) PLS_DATA request
2) PLS_DATA indication

d) Indication from the local PhL

3) PLS_CARRIER indication
4) PLS_SIGNAL indication
5) PLS_DATA_VALID indication

4.5 Local parameters, variable, counters, timers

4.5.1 Overview

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. It also uses local queues Q(...) as a means of ordering certain activities, of clarifying the effects of certain actions, and of clarifying the conditions under which those activities are valid.

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 timers shall be initialized to inactive;
c) all queues shall be initialized to empty.

DL-management may change the values of configuration variables.

4.5.2 Variables, parameters, counters and timers to support DLE function

4.5.2.1 V(MA), P(MA)

This variable is used by the CTC to record station address of this station. This variable shall be implemented by all stations. Its range is 1 to 2^{16}-1. When the station adopts short format DLPDU, the lower 8 bits of this variable is effective.

4.5.2.2 V(Tcycle), P(Tcycle)

This variable is used by the CTC to record transmission period of cyclic communication. This variable shall be implemented by all stations. The time that a set value indicates depends on V(Tunit). Its range is 1 000 to 64 000.

4.5.2.3 V(Nmax_slaves), P(Nmax_slaves)

This variable is used by the CTC to record the number of connectable slave stations. Its range is 1 to 62.

4.5.2.4 V(Cyc_sel), P(Cyc_sel)

This variable is used by the DLM to record the selection of transmission mode, which is cyclic or acyclic. The value is listed at Table 5.

Table 5 – List of the values of variable Cyc_sel

Value	Symbol	Description
0	CMode_Cyclic	Cyclic transmission mode
1	CMode_Acyclic	Acyclic transmission mode

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

4.5.2.5 V(Nmax_dly_cnt), P(Nmax_dly_cnt)

This variable is used by the DLM to record maximum-measurement-count, which limits the number of delay measurement execution. This variable is implemented only by the C1 master adopts configurable time slot. Its default value is 2. Its range is 2 to 31.

4.5.2.6 V(IO_sz), P(IO_sz)

This variable is used by the CTC to hold and designate the data size of send and receive frame, which is transferred in I/O data exchange band. Its range is 8 to 64. When CTC adopts fixed-width time slot, its value shall be same for all stations.

4.5.2.7 V(Pkt_sz), P(Pkt_sz)

This variable is used by message segmentation machine to record the message packet size of cyclic transmission. Its range is 4 to 500. When CTC adopts fixed-width time slot, its value shall be equal to V(IO_sz).

4.5.2.8 V(Nmax_retry), P(Nmax_retry)

This variable is used by the CTC to record maximum-retry-count, which limits the number of retries in the I/O data exchange retry band. This variable is implemented only by the C1 master. Its default value is 0, meaning retry are not permitted. Its range is 0 to 62.

4.5.2.9 V(Tslot), P(Tslot)

This variable is used by the CTC to hold and designate the value of the timeout time for I/O data exchange to each station. This variable shall be implemented by C1 master and C2 master. Its range is 0 to less than V(Tcycle). The time that a set value indicates depends on V(Tunit). When CTC adopts fixed-width time slot, its value shall be same for all stations.

4.5.2.10 V(Tunit), P(Tunit)

This variable is used by the CTC to record the unit of a set value of the variable concerning time, when the CTC adopts configurable time slot. Table 6 shows its range. This variable shall be implemented only by the CTC which adopts configurable time slot. CTC which adopts fixed-width time slot shall not this variable, and shall use 250 ns as time unit for all variables concerning time.

Table 6 – List of the values of variable Tunit

Value	Definition	Tcycle
0	10 ns	31,25 μs to 500 μs (See NOTE)
1	100 ns	More than 500 μs to 4 ms
2	1 μs	More than 4 ms to 64 ms
NOTE Value 0 is default.		

4.5.2.11 V(Tidly), P(Tidly)

This variable is used by the CTC to record the timing of event, which is issued periodically to synchronize DLS-user. This variable shall be implemented by all stations, which adopt configurable time slot. The value is delay time from the start of cyclic transmission. Its range is 0 to less than V(Tcycle). The time that a set value indicates depends on V(Tunit).

4.5.2.12 V(Tc2_dly), P(Tc2_dly)

This variable is used by the CTC to record the timing to start C2 message communication. This variable shall be implemented by C1 master and C2 master, which adopt configurable

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

time slot. The value is delay time from the end of cyclic transmission. Its range is 0 to less than V(Tcycle). The time that a set value indicates depends on V(Tunit).

4.5.2.13 V(Tmsg), P(Tmsg)

This variable is used by the CTC to record the time period for message communication. This variable shall be implemented by C1 master and C2 master. Its range is 0 to less than V(Tcycle). The time that a set value indicates depends on V(Tunit). When CTC adopts fixed-width time slot, its value shall be same for all stations.

4.5.2.14 V(Twrpt)

This variable is used by the SRC to record the time period for transmission delay measurement. This variable shall be implemented by C2 master and slave. Its range is V(Tslot) x 3 to less than 64 ms. The default value is 500 μs. The time that a set value indicates depends on V(Tunit).

4.5.2.15 V(IO_MAP), P(IO_MAP)

This variable is used by the DLM and CTC to record the information of the slaves and C2 master which to be connected in the network. See IEC 61158-3-24, 5.3.2.2.13 for the details.

4.5.2.16 V(Sts_STI), P(Sts_STI)

This variable is used by the DLM to hold the connection status of the stations which to be connected in the network. See IEC 61158-3-24, 5.3.3.2.2.1 for the details.

4.5.2.17 V(Sts_Err), P(Sts_Err)

This variable is used by the DLM to hold the error factor which occurred in the DLE.

4.5.2.18 V(Fc2msg)

This variable is used by the CTC to hold the presence of C2 message communication bandwidth in cyclic transmission mode. This variable shall be implemented by C1 master and C2 master. Its value is 0 or 1. The value shall be set to 1 when C1 message communication bandwidth is assigned, and set to 0 when it is not assigned.

4.5.2.19 V(Nslave)

This variable is used by the CTC to hold and designate slave number, which is being processed in the current time slot. This variable shall be implemented by C1 master and C2 master. Its range is 0 to V(Nmax_slaves). The value is reset to zero at the start of each cycle of cyclic transmission. And it is incremented when I/O data exchange to a slave is executed.

4.5.2.20 V(Nretry)

This variable is used by the CTC to hold and designate the element of retry list, which is being processed in the current time slot. The value is reset to zero at the start of each cycle of cyclic transmission. And it is incremented when I/O data exchange to a slave is fault, and decremented when I/O data exchange retry is executed.

4.5.2.21 V(Nrest_slot)

This variable is used by the CTC to hold and designate the number of rest time slot within the end of this cycle. Its range is 0 to V(Nmax_slave). This variable shall be implemented by C1 master, which adopt fixed-width time slot. The value is decremented in the range of V(Nmax_retry) to 0.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

4.5.2.22 V(Ndly_cnt)

This variable is used by the DLM to hold and designate the count of executed delay measurement. This variable shall be implemented by C1 master, which adopt configurable time slot. The value is reset to zero at the start of each cycle of cyclic transmission, and incremented when delay measurement is executed. Its range is 0 to V(Nmax_dly_cnt).

4.5.2.23 V(PDUType)

This variable is used by the CTC and DLM to hold and designate the selection of DLPDU type, which is basic format DLPDU or short format DLPDU. When the DLE adopts each of them, it may not implement this variable. The value is listed at Table 7.

Table 7 – List of the values of variable PDUType

Value	Symbol	Description
0	PDUBasic	Basic format DLPDU
1	PDUShort	Short format DLPDU

4.5.2.24 V(SlotType)

This variable is used by the CTC and DLM to hold and designate the selection of time slot type, which is fixed-width or configurable. When the DLE adopts each of them, it may not implement this variable. The value is listed at Table 8.

Table 8 – List of the values of variable SlotType

Value	Symbol	Description
0	TSFixed	Fixed-width time slot
1	TSConfig	Configurable time slot

4.5.2.25 V(Nms_1), V(Nms_2)

This variable is used by the CTC to hold the number of segmentations to be sent in the message communication using basic format DLPDU. The suffix "_1" and "_2" show the bandwidth of message communication, and indicate C1 message communication and C2 message communication respectively. This variable shall be implemented by all stations which execute message communication. The value is reset to zero at the first segment of each message. And, it is incremented by one every time CTC received acknowledge of one segment normally from peer station. Its range is 0 to 127.

4.5.2.26 V(Nmr_1), V(Nmr_2)

This variable is used by the CTC to hold the number of segments to be received in message communication using basic format DLPDU. The suffix "_1" and "_2" show the bandwidth of message communication, and indicate C1 message communication and C2 message communication respectively. This variable shall be implemented by all stations which execute message communication. The value is reset to zero at the first segment of each message. And it is incremented when one segment is received from peer station. Its range is 0 to 127.

4.5.2.27 V(Fmp_1), V(Fmp_2)

This variable is used by the CTC to hold and designate the flag of the polling request in message communication using basic format DLPDU. The suffix "_1" and "_2" show the bandwidth of message communication, and indicate C1 message communication and C2 message communication respectively. This variable shall be implemented by all stations

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

which execute message communication. Its range is 0 or 1. The value 1 means the polling request and 0 (default) means the other.

4.5.2.28 V(Fmf_1), V(Fmf_2)

This variable is used by the CTC to hold and designate the flag of the last segment in message communication using basic format DLPDU. The suffix "_1" and "_2" show the bandwidth of message communication, and indicate C1 message communication and C2 message communication respectively. This variable shall be implemented by all stations, which execute message communication. Its range is 0 or 1. The value is reset to zero at the first segment of each message, and set one when the last segment of the message is sent or received.

4.5.2.29 V(Ten)

This variable is used by the DLM to hold the timestamp of delay measurement end when the DLE adopts configurable time slot. The DLE shall implement this variable only when the DLE adopts configurable time slot.

4.5.2.30 V(Tdly)

This variable is used by the DLM to hold the transmission delay measured in CompDly state of DLM when the DLE adopts configurable time slot. The DLE shall implement this variable only when the DLE adopts configurable time slot.

4.5.2.31 V(Tmax_dly)

This variable is used by the DLM to hold the transmission delay measured in CompDly state of DLM when the DLE adopts configurable time slot. The DLE shall implement this variable only when the DLE adopts configurable time slot.

4.5.2.32 V(Tst)

This variable is used by the DLM to hold the timestamp of delay measurement start time when the DLE adopts configurable time slot. The DLE shall implement this variable only when the DLE adopts configurable time slot.

4.5.2.33 T(Tcycle)

T(Tcycle) is used by the CTC to measure the cyclic transmission period. The value is decremented in the range of V(Tcycle) to 0.

4.5.2.34 T(Tslot)

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

4.5.2.35 T(Tmsg)

T(Tmsg) is used by the CTC to measure the time period of message communication band. The value is decremented in the range of V(Tmsg) to 0.

4.5.2.36 T(Twrpt)

T(Twrpt) is used by the SRC to watch repeat function. The value is decremented in the range of V(Twrpt) to 0.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

4.5.2.37 Q(MSGc1s), Q(MSGc2s)

The queue buffer is implemented at the interface between CTC and MSM to transfer the send message. CTC enqueues the message to be sent, then MSM dequeues the message to build DLPDU and send it.

4.5.2.38 Q(MSGc1r), Q(MSGc2r)

The queue buffer is implemented at the interface between CTC and Message segmentation protocol machine to transfer the received message. MSM builds the message from the received DLPDU and enqueues it, then CTC dequeues the message to transfer it to the DLS-user.

5 DLPDU structure

5.1 Overview

5.1.1 Transfer syntax for bit sequences

For transmission across Type 24 DL a bit sequence is reordered into a sequence of octets. Hexadecimal notation is used for octets as specified in ISO/IEC 9899. Let $b = b_0...b_{n-1}$ be a bit sequence. Denote k a non-negative integer such that $8(k - 1) <= n < 8k$. Then b is transferred in k octets assembled as shown in Table 9. The bits b_i, $i > n$ of the highest numbered octet shall be ignored.

Table 9 – Transfer syntax for bit sequences

Octet number	1	2	k
	$b_7...b_0$	$b_{15}..b_8$	$b_{8k-1}..b_{8k-8}$

When the DLE implemented over the PHY defined in IEC 61158-2, Type 24, octet 1 is transmitted first and octet k is transmitted last. Hence the bit sequence is transferred as follows across the network:

$b_0, b_1, ..., b_7, b_8, ..., b_{15}, ...$

5.1.2 Data type encodings

Data of basic data type Unsigned*n* has values in the non-negative integers. The value range is 0, ..., 2^n-1. The data is represented as bit sequences of length n. The bit sequence

$b = b_0 ...b_{n-1}$

is assigned the value

$$\text{Unsigned}(b) = b_{n-1} \times 2^{n-1} + ... + b_1 \times 2^1 + b_0 \times 2^0$$

The bit sequence starts on the left with the least significant octet.

EXAMPLE The value 266 = 0x10A with data type Unsigned16 is transferred in two octets, first 0x0A and then 0x01.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Table 10 – Bit order

Octet number	1	2	3	4	5	6	7	8
Unsigned8	$b_7..b_0$							
Unsigned16	$b_7..b_0$	$b_{15}..b_8$						
Unsigned32	$b_7..b_0$	$b_{15}..b_8$	$b_{23}..b_{16}$	$b_{31}..b_{24}$				
Unsigned64	$b_7..b_0$	$b_{15}..b_8$	$b_{23}..b_{16}$	$b_{31}..b_{24}$	$b_{39}..b_{32}$	$b_{47}..b_{40}$	$b_{55}..b_{48}$	$b_{63}..b_{56}$

The Unsigned*n* data types are transferred as specified in Table 10. Unsigned data types as Unsigned1 to Unsigned7 and Unsigned 9 to Unsigned15 will be used too. In this case the next element will start at the first free bit position as denoted in 5.1.1.

5.1.3 Frame format

There are two types of frame format. One is basic format and the other is short format that based on ISO/IEC 13239:2002 (HDLC).

5.2 Basic format DLPDU structure

5.2.1 General

Figure 7 shows the structure of basic frame format.

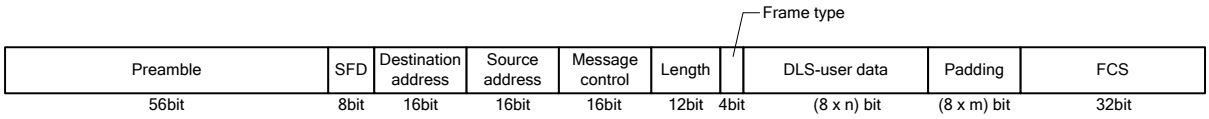


Figure 7 – Basic format DLPDU structure

5.2.1.1 Preamble

The preamble field is identical to Clause 3 of ISO/IEC 8802-3:2000. The preamble field is a 56-bit field that is used to allow the physical signaling part circuitry to reach its steady state synchronization with the receiving frame timing.

The preamble pattern is:

“10101010 10101010 10101010 10101010 10101010 10101010 10101010.”

5.2.1.2 Start frame delimiter (SFD)

The Start Frame Delimiter (SFD) is identical to Clause 3 of ISO/IEC 8802-3:2000. The SFD field is the sequence of bit pattern “10101011”. It immediately follows the preamble pattern and indicates the start of a frame.

5.2.1.3 Destination address (DA)

Destination address shall contain the node address of the destination DLE.

The higher 8 bits of a 16-bit address is used as an extended address and the lower 8 bits is used as a station address (see Table 11).

For both station addresses and extension address, the usable addresses are specified in Table 12 and Table 13.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Table 11 – Destination and Source address format

Octet number	Size	Contents
1	Unsigned8	Station address
2	Unsigned8	Extended address

Table 12 – Station address

Station address	Contents
0x00	Reserved
0x01	C1 master
0x02	C2 master
0x03 to 0xEF	Slave, Gateway
0xF0 to 0xFE	Reserved
0xFF	Broadcast address

Table 13 – Extended address

Extended address	Contents
0x00 to 0xFEh	Device address for the node that has multiple device, or remote address for gateway node
0xFF	Broadcast address[a]
[a] Only synchronous frame may use broadcast address.	

5.2.1.4 Source address (SA)

Source address shall contain the node address of the source DLE. See 5.2.1.3 for its format and its range.

5.2.1.5 Message control

Message control field is used for send data with acknowledge service. This field is effective only when Frame type which follows this field is Message frame. This field shall be zero except Message frame.

This field has two formats. One is Information transfer format and the other is supervisory format (see Table 14 and Table 15). Each field shall have the same function as the field with the same name in ISO/IEC 13239:2000 (HDLC).

NOTE In HDLC and this specification, the order of transmitting the bit is reverse.

Table 14 – Message control field format (Information transfer format)

Octet number	Size	Symbol	Contents
1	Unsigned7	N(R)	Transmitting receive sequence number
	Unsigned1	P/F	Poll bit – primary transmissions Final bit – secondary transmission
2	Unsigned7	N(S)	Transmitting send sequence number

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Octet number	Size	Symbol	Contents
	Unsigned1	-	Reserved (0)[a]
[a] This bit shall be zero.			

Table 15 – Message control field format (Supervisory format)

Octet number	Size	Symbol	Contents
1	Unsigned7	N(R)	Transmitting receive sequence number
	Unsigned1	-	Reserved (1)[a]
2	Unsigned4	-	Reserved (0)[b]
	Unsigned2	S	Supervisory function bits
	Unsigned1	-	Reserved (0)[b]
	Unsigned1	-	Reserved (1)[a]
[a] This bit shall be one. [b] This bits shall be zero.			

Table 16 – The list of Supervisory function bits

Value	Symbol	Description	Note
0	RR	Receive ready (RR) command or RR response	
1	REJ	Reject (REJ) command or REJ response	
2	RNR	Receive not ready (RNR) response	
3	–	Reserved	

5.2.1.6 Type and length field

Frame type and Data length field consists of a higher 4-bit area where the frame type is set and a lower 12-bit area where the data length is set (see Table 17).

The frame type is used for identifying the contents of a frame and also for identifying the type of the protocol for message communication. Table 18 shows the definition of frame type. The data length indicates the size of the data stored in a frame in terms of the number of octets.

Table 17 – Frame type and Data length format

Octet number	Size	Contents
1	Unsigned12	Data length (lower 8 bits)
2		Data length (higher 4 bits)
	Unsigned4	Frame type

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Table 18 – The list of Frame type

Value	Symbol	Description
0	–	Reserved
1	FT_SYNC	Synchronous Frame
2	FT_IO	Output data and Input data frame
3	FT_DLST	Delay measurement start frame
4	FT_DLMS	Delay measurement frame
5	FT_MTKN	Message token frame
6	FT_STS	Status frame
7	FT_CINF	Cyclic information frame
8 to 11	–	Reserved
12	FT_MSG	Message frame
13 to 15	–	Reserved

5.2.1.7 DLS-user data field

The data format varies according to the frame type. With message frames, if the application data cannot be set in one frame, it is possible to set and send the data in multiple frames using message control.

5.2.1.8 Field check sequence field (FCS)

A frame check sequence (FCS) uses 32-bit CRC-CCITT. FCS is calculated in the range from the destination address to the end of the data except the FCS itself.

5.2.2 Synchronous frame

C1 master uses this frame to synchronize slaves and C2 master. Only C1 master may send this frame. C1 master shall set the station address as the broadcast address (0xFF), and the slaves and the C2 master shall receive this frame.

When slave and C2 master receive this frame, they shall refresh the local clock with the time calculated by adding the transmission delay measured in advance (notified with delay measurement frame) to the current time stored this frame.

Table 19 and Table 20 show detailed the data format of this frame.

Table 19 – Data format of Synchronous frame

Octet number	Size	Contents
1 to 4	Unsigned32	Timestamp
5 to 6	Unsigned16	Cyclic event delay time
7 to 8	Unsigned16	Reserved

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Table 20 – The field list of Synchronous frame

Field	Contents	Maximum and minimum value	Recommended value	Note
Timestamp	Time stamp of C1 master at the frame is sent	0 to 2^{32}-1	–	The unit is defined according to the setting of the variable V(Tunit).
Cyclic event delay time	Delay time from the current time stored in this frame to the time when the cyclic event is indicated	0 to 2^{16}-1	0	The unit is defined according to the setting of the variable V(Tunit).

5.2.3 Output data or Input data frame

C1 master uses this frame for the output data to be sent to slaves, and slaves use this frame for the input data to be sent to C1 master, within I/O data exchange bandwidth of cyclic transmission mode. C2 master may only receive this frame and shall not send this frame.

Either of the destination address or the source address of this frame must be C1 master because this frame is exchanged between dataC1 master and slave. The data length must not be changed during normal operation.

Table 21 and Table 22 show the data format of this frame.

Table 21 – Data format of Output data or Input data frame

Octet number	Size	Contents
1 to n	Unsigned8n	Output data or Input data
(n+1) to (n+m)	Unsigned8m	Padding

Table 22 – The field list of Output data or Input data frame

Field	Contents	Maximum and minimum value	Recommended value	Note
Output data or Input data	Output data from C1 master to slave or input data from slave to C1 master	0 to maximum value that can be represented with the octet length of the specified data length	–	
Padding	Area adjusted to become multiple in 32 bits in data length	Don't care	0	

5.2.4 Delay measurement start frame

C1 master uses this frame to specify the targeted station of the delay measurement to measure the transmission delay from C1 master to each slave or C2 master. Only C1 master transmits this frame. After the station receives this frame, the station returns the receipt frame till the number of times that specified with this frame.

Table 23 and Table 24 show detailed the data format of this frame.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Table 23 – Data format of Delay measurement start frame

Octet number	Size	Contents
1 to 2	Unsigned16	Measurement number
3 to 4	Unsigned16	Reserved

Table 24 – The field list of Delay measurement start frame

Field	Contents	Maximum and minimum value	Recommended value	Note
Measurement number	Number of times for sending the transmission delay measurement frame	1 to 32	1	

5.2.5 Delay measurement frame

C1 master uses this frame to measure the transmission delay and to notify the slaves and C2 master of the result. Only C1 master transmits this frame. C1 master shall send this frame to the station to which the delay measurement start frame is sent, and send this frame till the times specified with the delay measurement start frame. C1 master shall notify of the result of the measurement by using this frame.

Slave and C2 master that received delay measurement start frame shall receive and return this frame. They shall stop returning the receipt frame after receiving this frame till the number of times notified with the delay measurement start frame.

Table 25 and Table 26 show detailed the data format of this frame.

Table 25 – Data format of Delay measurement frame

Octet number	Size	Contents
1 to 4	Unsigned32	Timestamp
5 to 6	Unsigned16	Transmission delay
7 to 8	Unsigned16	Reserved

Table 26 – The field list of Delay measurement frame

Field	Contents	Maximum and minimum value	Recommended value	Note
Transmission delay	Time from sending a frame of C1 master to reception of slave or C2 master of it.	0 to 2^{16}-1	–	The unit is defined according to the setting of the time unit. The default unit is 10 ns.

5.2.6 Message token frame

C1 master sends this frame to notify C2 master of the permission of start C2 message communication before C2 message communication start time. C1 master shall send this frame only when C1 master exchanges the command data/response data with all slaves and C1 message communication ends before C2 message communication start time. However, C1 master shall not send this frame when the time from the current time of the local clock to the start time of C2 message communication is shorter than the transmission delay between C1 master and the C2 master.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

C2 master that receives this frame may start C2 message communication unless C2 message communication start time has come.

Only C1 master may send this frame. The destination and source addresses are fixed because only C2 master may receive this frame. This frame contains no data.

5.2.7 Status frame

C1 master uses this frame to inquire the status of slaves and C2 master. The station that received this frame from C1 master shall send this frame to notify of the current status. Slaves and C2 master shall return this frame also when receiving a cycle information frame contains their own address as the destination. Slave and C2 master may return this frame to request C1 master to execute the transmission delay measurement even when they receive the other frame than the synchronous frame from C1 master.

Broadcast address or multicast address shall not be specified as the destination of this frame.

Table 27 to Table 30 show detailed the data format of this frame.

Table 27 – Data format of Status frame

Octet number	Size	Contents
1 to 2	Unsigned16	Status
3 to 4	Unsigned16	Repeater status

Table 28 – The field list of Status frame

Field	Contents	Maximum and minimum value	Recommended value	Note
Status	Current value of the DLE status	(See Table 29)	–	Refreshed by slave or C2 master
Repeater status	Current value of repeater status	(See Table 30)	–	Refreshed by slave or C2 master

Table 29 – The list of the DLE status

Status code	Description
0x0000	Station not exist
0x0023	Station address duplicated
0x0024	Waiting for the delay measurement request by DLS-user
0x0025	Waiting for delay measurement start indication from C1 master
0x0026	Measuring transmission delay
0x0027	Waiting for cyclic information frame from C1 master
0x0050	Operating in cyclic transmission mode
0x0060	Operating in acyclic transmission mode

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Table 30 – The list of Repeater status

Send status			
bit	**Symbol**	**Description**	**Initial**
0	–	Reserved	0
1	MII_RXTXE	Send request detected while receiving data from PhL	0
2	MII_TXRXE	Receive data from PhL detected while sending data to PhL	0
3	MII_RXRXE	Receive data from PhL detected while receiving data from PhL	0
4 to 15	–	Reserved	0

5.2.8 Cycle Information frame

C1 master uses this frame to notify slave and C2 master of the transmission mode. Only C1 master may send this frame.

C1 master may broadcast this frame or send it to slaves or C2 master individually. Slaves and C2 master shall return a status frame when they receive this frame that contains their own address as the destination.

Table 31 and Table 32 show detailed the data format of this frame.

Table 31 – Data format of Delay measurement frame

Octet number	Size	Contents
1 to 2	Unsigned16	Transmission cycle
3 to 4	Unsigned16	C2 message delay
5 to 6	Unsigned16	Maximum delay
7	Unsigned8	Communication mode
8	Unsigned8	Time unit

Table 32 – The field list of Cycle Information frame

Field	Contents	Maximum and minimum value	Note
Transmission cycle	Transmission cycle of the cyclic transmission mode	3 125 to 64 000	a)
C2 message delay	Delay time from the current time stored in the synchronous frame to the time of C2 message communication start	0 to 2^{16}-1	
Maximum delay	Maximum value among the transmission delay measured by C1 master.	0 to 2^{16}-1	a)
Communication mode	Selection of transmission mode to be executed after initialization	0: Cyclic 1: Acyclic	
Time unit	Selection code of time unit	0 : 10 ns 1 : 100 ns 2 : 1 μs	b)

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

a) The unit is defined by the value of the time unit field in this frame. b) When the transmission cycle Tcycle is 31,25 μs <= T_MCYC <= 500 μs, set to 10 ns. In the case of 500 μs < T_MCYC <= 4 ms, set to 100 ns. When 4 ms < T_MCYC <= 64 ms, set to 1 μs.

5.2.9 Message frame

This frame is used for message transmission. All stations may send this frame. When the DLSDU that requested to send is beyond the size that may contained within one frame, the DLSDU shall be divided by using the message control field in this frame.

Table 33 and Table 34 show detailed the data format of this frame.

Table 33 – Data format of Message frame

Octet number	Size	Contents
1 to n	Unsigned8n	Message data
(n+1) to (n+m)	Unsigned8m	Padding

Table 34 – The field list of Message frame

Field	Contents	Maximum and minimum value	Recommended value	Note
Message data	Arbitrary data except output data or input data	0 to maximum value that can be represented with the octet length of the specified data length	–	
Padding	Area adjusted to become multiple in 32 bits in data length	Don't care	0	

5.3 Short format DLPDU structure

5.3.1 General

The short format DLPDU is shown in Figure 8. Transmission sequence shall be from left to right as shown in Figure 8, i.e. station address first, followed by control, DLS-user data and finally CRC.

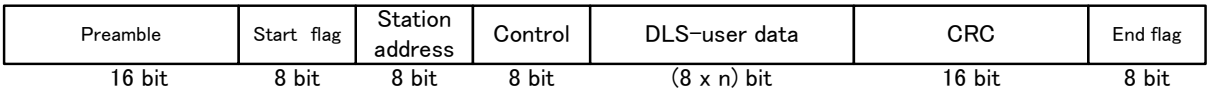


Figure 8 – Short format DLPDU structure

5.3.1.1 Preamble

The preamble field is identical to ISO/IEC 13239:2002 (HDLC). The preamble field is a 16-bits field that is used to allow the physical signaling part circuitry to reach its steady state synchronization with the receiving frame timing.

The preamble pattern is:

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

“10101010 10101010.”

5.3.1.2 Start flag

The start flag is identical to ISO/IEC 13239:2002 (HDLC). This field is the sequence of bit pattern “10101011”. It immediately follows the preamble pattern and indicates the start of a frame.

Due to bit-stuffing, this bit sequence is prevented from occurring inside the DLSDU sequence. The DLE shall only accept a received signal burst as a DLPDU after verifying this sequence and shall remove this sequence before transferring the DLSDU sequence to the DLS-user.

5.3.1.3 Station address field

Table 35 shows the range of the station address field.

Table 35 – Range of Station address field

Station Address	Contents
0x00	Reserved
0x01	C1 maseter
0x02	C2 maseter
0x03 to 0xDE	Slaves
0xDF	Not connected (default)
0xE0 to 0xFE	Reserved
0xFF	Broadcast address

5.3.1.4 Control field (CTL)

Control field is used to identify the frame type. This field has two formats. One is I/O data exchange format and the other is Message format. Table 36 to Table 38 show detailed this field.

Table 36 – Control field format (I/O data exchange format)

Octet number	Size	Symbol	Contents
1	Unsigned4	CMD	Input data or Output data flag; 1: Input data (response) 3: Output data (command)
	Unsigned4	-	Reserved (0)[a]
[a] This bit shall be zero.			

Table 37 – Control field format (Message format)

Octet number	Size	Symbol	Contents
1	Unsigned4	S(n)	Sequence number
	Unsigned1	-	Reserved (1)[a]
	Unsigned1	C1/C2	Primary node flag
	Unsigned1	END	End flag
	Unsigned1	S/D	Sync / data flag
[a] This bit shall be one.			

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Table 38 – The field list of Message format

Symbol	Description
S(n)	Transmitting send or receive sequence number.
C1/C2	0: C1 master 1: C2 master
END	When S/D is 0, 0: Data transfer from primary node to secondary node 1: Data transfer from secondary node to primary node When S/D is 1, 0: Following data exists 1: This data is last data
S/D	0: Handshake frame 1: Data frame

5.3.1.5 DLS-user data field

The length of the DLS-user data field is from 8 octets to 64 octets.

5.3.1.6 CRC field

A frame check sequence (FCS) using a 16-bit CRC-CCITT is used. CRC is calculated based on the following formula.

$$P(x) = X^{16} + X^{12} + X^{5} + 1$$

5.3.1.7 End flag

The sequence of symbols in this field is identical to the start flag (see 5.3.1.2).

5.3.2 Synchronous frame

C1 master uses this frame to synchronize slaves and C2 master. Only C1 master may send this frame. C1 master shall set the station address to the broadcast address (0xFF), and slaves and C2 master shall receive this frame. When they receive this frame, they shall check the CRC field contained in this frame, and then issue a cyclic event if the result of the check has no error.

Table 39 and Table 40 show detailed the data format of this frame.

Table 39 – Data format of Synchronous frame

Octet number	Size	Contents
1 to 2	Unsigned16	Transmission cycle
3 to 4	Unsigned16	Time slot width
5 to 16 or 31	-	Reserved

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Table 40 – The field list of Sync frame

Field	Contents	Maximum and minimum value	Note
Transmission cycle	Transmission cycle notified C2 master	0 to 2^{16}-1	The unit is 0,25 us
Time slot width	Time slot	0 to 2^{16}-1	The unit is 0,25 us

5.3.3 Output data or Input data frame

5.3.3.1 Output data frame

C1 master uses this frame for the output data to be sent to salve within I/O data exchange bandwidth of cyclic transmission mode. C2 master only may receive this frame. The length of the output data shall be 16 or 31 octets.

Table 41 and Table 42 show detailed the data format of this frame.

Table 41 – Data format of Output data frame

Octet number	Size	Contents
1 to 16 or 31	Unsigned8n	Output data

Table 42 – The field list of Output data frame

Field	Contents	Maximum and minimum value	Recommended value	Note
Output data	Output data from C1 master to slave	0 to Maximum value according to the data length	–	

5.3.3.2 Input data frame

Slaves use this frame for the input data to be sent to C1 master within I/O data exchange bandwidth of cyclic transmission mode. C2 master only may receive this frame. The length of the input data shall be 16 or 31 octets.

Table 43 and Table 44 show detailed the data format of this frame.

Table 43 – Data format of Input data frame

Octet number	Size	Contents
1 to 16 or 31	Unsigned8n	Input data

Table 44 – The field list of Input data frame

Field	Contents	Maximum and minimum value	Recommended value	Note
Input data	Input data from slave to C1 master	0 to Maximum value according to the data length	–	

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

5.3.4 Message frame

This frame is used for message transmission. All stations may send this frame. When the DLSDU that requested to send is beyond the size that may contained within one frame, the DLSDU shall be divided by using the message control field in this frame.

6 DLE element procedure

6.1 Overview

In following subclause, the operation of CTC and SRC is described. Though DL-management is a component of DLE, it is described in the next clause.

6.2 Cyclic transmission control sublayer

6.2.1 General

CTC provides the following functions.

- Control of communication sequence and communication band (C1 master)
- Monitoring of all I/O data exchanged between C1 master and slaves (C2 master)
- Sending response as a server (Slave)
- Message communication in cyclic communication
- Sporadic data exchange

6.2.2 DLS-user interface

6.2.2.1 Primitive definitions

The primitives exchanged with the DLS-user between CTC are shown below. Table 45 is for the case that issued by DLS-user and Table 46 is for the case that issued by CTC.

Table 45 – The primitives and parameters for DLS-user interface issued by DLS-user

Primitive	Source	Parameter	Function
DL-WRITE-DATA.req	DLS-user	SAP_ID, DLS-user-data	Send data writing request
DL-READ-DATA.req	DLS-user	SAP_ID	Receive data reading request
DL-SDA.req	DLS-user	SAP_ID, Node_ID, Length, DLS-user-data	Send data request with transmission confirmation
DL-SDN.req	DLS-user	SAP_ID, Node_ID, Length, DLS-user-data	Send data request without transmission confirmation
DL_GET_STATUS.req	DLS-user	Sts_ID	DL parameter referring request

Table 46 – The primitives and parameters for DLS-user interface issued by CTC

Primitive	Source	Parameter
DL-WRITE-DATA.cnf	CTC	Result
DL-READ-DATA.cnf	CTC	Result, DLS-user-data
DL-SDA.ind	CTC	Node_ID, Length, DLS-user-data
DL-SDA.cnf	CTC	Result
DL-SDN.ind	CTC	Node_ID, Length, DLS-user-data
DL-SDN.cnf	CTC	Result

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Primitive	Source	Parameter
DL_GET_STATUS.cnf	CTC	Result, Cur_Status
DL_EVENT.ind	CTC	Event_ID

6.2.2.2 Overview of the interactions

See IEC 61158-3-24, 4.3.

6.2.2.3 Detailed definitions of the primitives and parameters

See IEC 61158-3-24, 4.4.

6.2.3 Protocol machines in CTC

6.2.3.1 Overview

There are three protocol machines in CTC, which are cyclic transmission protocol machine, message segmentation protocol machine, and acyclic transmission protocol machine.

The cyclic communication protocol machine has two kinds of protocol machines corresponding to the adopted frame format. One is “fixed-width time slot type” whose transmission sequence is composed of same-width time slot, and the other is “configurable time slot type” whose transmission sequence for each station may be different time slot. In addition, each of two protocol machines of “fixed-width time slot type” and “fixed-width time slot type” has three kinds of protocol machines corresponding to the station type (the C1 master, the C2 master, and the slave).

The cyclic communication protocol machine has two protocol machine, sender and receiver.

The message communication protocol machine works only while the cyclic transmission protocol machine is active.

6.2.3.2 Cyclic transmission protocol machine

6.2.3.2.1 Protocol machine for cyclic communication consists of fixed-width time slot

6.2.3.2.1.1 C1 master

Figure 9 and Table 47 show the state diagram and the state table of C1 master that adopts fixed-width time slot.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

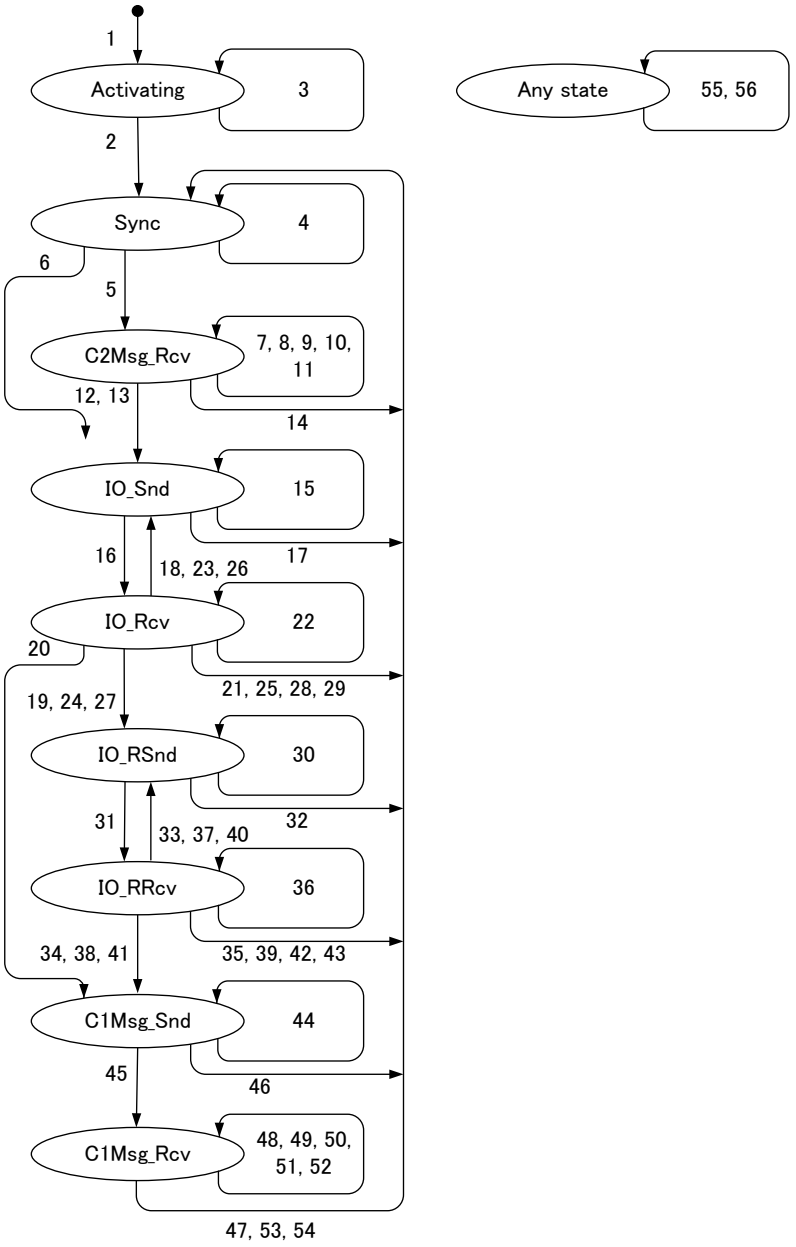


Figure 9 – The state diagram of C1 master for fixed-width time slot

Table 47 – The state table of C1 master for fixed-width time slot

No.	Current state	Event /condition =>action	Next state
1	Any state	Power on or CTC_Reset.req => (none)	Activating
2	Activating	CTC_Start.req / CHECK_VARS() = True => START_TIMER(T(Tcycle), V(Tcycle)) CTC_Start.cnf { OK }	Sync
3	Activating	CTC_Start.req / CHECK_VARS() <> True => CTC_Start.cnf { NG }	Activating

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
4	Sync	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } SRCSDU := BUILD_PDU_SYNC() Node_ID := GET_DA(SRCSDU) Length := GET_LEN(SRCSDU) SRC_Send_Frame.req { Node_ID, Length, SRCSDU} START_TIMER(T(Tslot), V(Tslot)) V(Nslave) := 1 V(Nretry) := 0 V(Nrest_slot) := V(Nmax_retry)	Sync
5	Sync	SRC_Send_Frame.cnf { Result } / V(Fc2msg) = “True” => (none)	C2Msg_Rcv
6	Sync	SRC_Send_Frame.cnf { Result } / V(Fc2msg) = “Flase” => (none)	IO_Snd
7	C2Msg_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && GET_MC(RcvSRCSDU) = Ec2msg)) => EXEC_MSPM_R(Ec2msg, Rcv_sts, RcvSRCSDU, SndSRCSDU) Node_ID := GET_DA(SndSRCSDU) Length := GET_LEN(SndSRCSDU) SRC_Send_Frame.req { Node_ID, Length, SRCSDU}	C2Msg_Rcv
8	C2Msg_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && GET_MC(RcvSRCSDU) = Ec1msg)) => (none)	C2Msg_Rcv
9	C2Msg_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) <> FT_MSG)) => (none)	C2Msg_Rcv
10	C2Msg_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) <> V(MA))) => (none)	C2Msg_Rcv
11	C2Msg_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / (Rcv_sts <> OK) => (none)	C2Msg_Rcv
12	C2Msg_Rcv	SRC_Send_Frame.cnf { Result } => (none)	IO_Snd
13	C2Msg_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" => SRCSDU := BUILD_PDU_OUT(V(Nslave)) Node_ID := GET_DA(SndSRCSDU) Length := GET_LEN(SndSRCSDU)	IO_Snd

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		SRC_Send_Frame.req { Node_ID, Length, SRCSDU}	
14	C2Msg_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot))	Sync
15	IO_Snd	EXPIRED_TIMER (T(Tslot)) = "True" => SRCSDU := BUILD_PDU_OUT(V(Nslave)) Node_ID := GET_DA(SndSRCSDU) Length := GET_LEN(SndSRCSDU) SRC_Send_Frame.req { Node_ID, Length, SRCSDU}	IO_Snd
16	IO_Snd	SRC_Send_Frame.cnf { } => (none)	IO_Rcv
17	IO_Snd	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot)	Sync
18	IO_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) = GET_NODE_ID(V(Nslave)) && (V(Nslave) < V(Nmax_slave)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts) V(Nslave) := V(Nslave) + 1	IO_Snd
19	IO_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) = GET_NODE_ID(V(Nslave)) && (V(Nslave) >= V(Nmax_slave) && (V(Nrest_slot) > 0) && (V(Nretry) > 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts)	IO_RSnd
20	IO_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) = GET_NODE_ID(V(Nslave)) && (V(Nslave) >= V(Nmax_slave) && (V(Nrest_slot) > 0) && (V(Nretry) = 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts)	C1Msg_Snd
21	IO_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) = GET_NODE_ID(V(Nslave)) && (V(Nslave) >= V(Nmax_slave) && (V(Nrest_slot) <= 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts)	Sync
22	IO_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) <> GET_NODE_ID(V(Nslave))) => (none)	IO_Rcv
23	IO_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts <> OK) && (V(Nslave) < V(Nmax_slave)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts) PUT_RETRY_LIST(V(Nslave)) V(Nslave) := V(Nslave) + 1	IO_Snd
24	IO_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts <> OK) && (V(Nslave) >= V(Nmax_slave) && (V(Nrest_slot) > 0))	IO_RSnd

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		=> STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts) PUT_RETRY_LIST(V(Nslave)) V(Nslave) := V(Nslave) + 1	
25	IO_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts <> OK) && (V(Nslave) >= V(Nmax_slave) && (V(Nrest_slot) <= 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts) PUT_RETRY_LIST(V(Nslave)) V(Nslave) := V(Nslave) + 1	Sync

26	IO_Rcv	EXPIRED_TIMER (T(Tslot)) = "True"
		/ V(Nslave) < V(Nmax_slave)
		=> STORE_PDU_IN(V(Nslave), SRCSDU, NG)
		If V(Nrest_slot) > 0 then
		PUT_RETRY_LIST(V(Nslave))
		endif
		V(Nslave) := V(Nslave) + 1
			IO_Snd

27	IO_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) > 0) => STORE_PDU_IN(V(Nslave), SRCSDU, NG) PUT_RETRY_LIST(V(Nslave)) V(Nslave) := GET_RETRY_LIST() SRCSDU := BUILD_PDU_OUT(V(Nslave)) Node_ID := GET_DA(SRCSDU) Length := GET_LEN(SRCSDU) SRC_Send_Frame.req { Node_ID, Length, SRCSDU}	IO_RSnd
28	IO_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) = 0) => STORE_PDU_IN(V(Nslave), SRCSDU, NG)	Sync
29	IO_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot) STORE_PDU_IN(V(Nslave), SRCSDU, NG)	Sync
30	IO_RSnd	EXPIRED_TIMER (T(Tslot)) = "True" => V(Nslave) := GET_RETRY_LIST() SRCSDU := BUILD_PDU_OUT(V(Nslave)) Node_ID := GET_DA(SRCSDU) Length := GET_LEN(SRCSDU) SRC_Send_Frame.req { Node_ID, Length, SRCSDU}	IO_RSnd
31	IO_RSnd	SRC_Send_Frame.cnf { } => (none)	IO_RRcv
32	IO_RSnd	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot)	Sync
33	IO_RRcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU}	IO_RSnd

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		/ ((Rcv_sts = OK) && (GET_DA(SRCSDU) = GET_NODE_ID(V(Nslave)) && (V(Nrest_slot) >= 1) && (V(Nretry) > 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts) V(Nrest_slot) := V(Nrest_slot) – 1	
34	IO_RRcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) = GET_NODE_ID(V(Nslave)) && (V(Nrest_slot) >= 1) && (V(Nretry) <= 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts)	C1Msg_Snd
35	IO_RRcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) = GET_NODE_ID(V(Nslave)) && (V(Nrest_slot) < 1)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts)	Sync
36	IO_RRcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) <> GET_NODE_ID(V(Nslave))) => (none)	IO_RRcv
37	IO_RRcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts <> OK) && (V(Nrest_slot) >= 1) && (V(Nretry) > 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts) V(Nrest_slot) := V(Nrest_slot) – 1	IO_RSnd
38	IO_RRcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts <> OK) && (V(Nrest_slot) >= 1) && (V(Nretry) <= 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts) V(Nrest_slot) := V(Nrest_slot) – 1	C1Msg_Snd
39	IO_RRcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts <> OK) && (V(Nrest_slot) < 1)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts)	Sync
40	IO_RRcv	EXPIRED_TIMER (T(Tslot)) = "True" / ((V(Nrest_slot) >= 1) && (V(Nretry) > 0)) => STORE_PDU_IN(V(Nslave), None, TOUT) V(Nrest_slot) := V(Nrest_slot) – 1 V(Nslave) := GET_RETRY_LIST() SRCSDU := BUILD_PDU_OUT(V(Nslave)) Node_ID := GET_DA(SRCSDU) Length := GET_LEN(SRCSDU) SRC_Send_Frame.req { Node_ID, Length, SRCSDU}	IO_RSnd
41	IO_RRcv	EXPIRED_TIMER (T(Tslot)) = "True" / ((V(Nrest_slot) >= 1) && (V(Nretry) <= 0)) => STORE_PDU_IN(V(Nslave), None, TOUT) V(Nrest_slot) := V(Nrest_slot) – 1 EXEC_MSPM_IS(Ec1msg, SRCSDU) Node_ID := GET_DA(SRCSDU) Length := GET_LEN(SRCSDU)	C1Msg_Snd

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		SRC_Send_Frame.req { Node_ID, Length, SRCSDU}	
42	IO_RRcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nrest_slot) < 1) => STORE_PDU_IN(V(Nslave), None, TOUT)	Sync
43	IO_RRcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot) STORE_PDU_IN(V(Nslave), SRCSDU, TOUT)	Sync
44	C1Msg_Snd	EXPIRED_TIMER (T(Tslot)) = "True" => EXEC_MSPM_IS(Ec1msg, SndSRCSDU) Node_ID := GET_DA(SndSRCSDU) Length := GET_LEN(SndSRCSDU) SRC_Send_Frame.req { Node_ID, Length, SndSRCSDU}	C1Msg_Snd
45	C1Msg_Snd	SRC_Send_Frame.cnf { } => (none)	C1Msg_Rcv
46	C1Msg_Snd	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot)	Sync
47	C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, RcvSRCSDU} / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && GET_MC(RcvSRCSDU) = Ec1msg)) => STOP_TIMER(T(Tslot)) EXEC_MSPM_IR(E1msg, Rcv_sts, RcvSRCSDU)	Sync
48	C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, RcvSRCSDU} / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && GET_MC(RcvSRCSDU) = Ec2msg)) => (none)	C1Msg_Rcv
49	C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, RcvSRCSDU} / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && GET_MC(RcvSRCSDU) = Ec2msg)) => (none)	C1Msg_Rcv
50	C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, RcvSRCSDU} / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) <> FT_MSG)) => (none)	C1Msg_Rcv
51	C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, RcvSRCSDU} / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) <> V(MA))) => (none)	C1Msg_Rcv
52	C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, RcvSRCSDU} / ((Rcv_sts <> OK) => (none)	C1Msg_Rcv
53	C1Msg_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" => STOP_TIMER(T(Tslot))	Sync
54	C1Msg_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True"	Sync

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		=> DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot))	
55	Any state	DL-WRITE-DATA.req(SAP_ID, DLSDU) => Result := SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	Same state
56	Any state	DL-READ-DATA.req(SAP_ID) => Result := GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	Same state

6.2.3.2.1.2 C2 master

Figure 10 and Table 48 show the state diagram and the state table of C2 master that adopts fixed-width time slot.

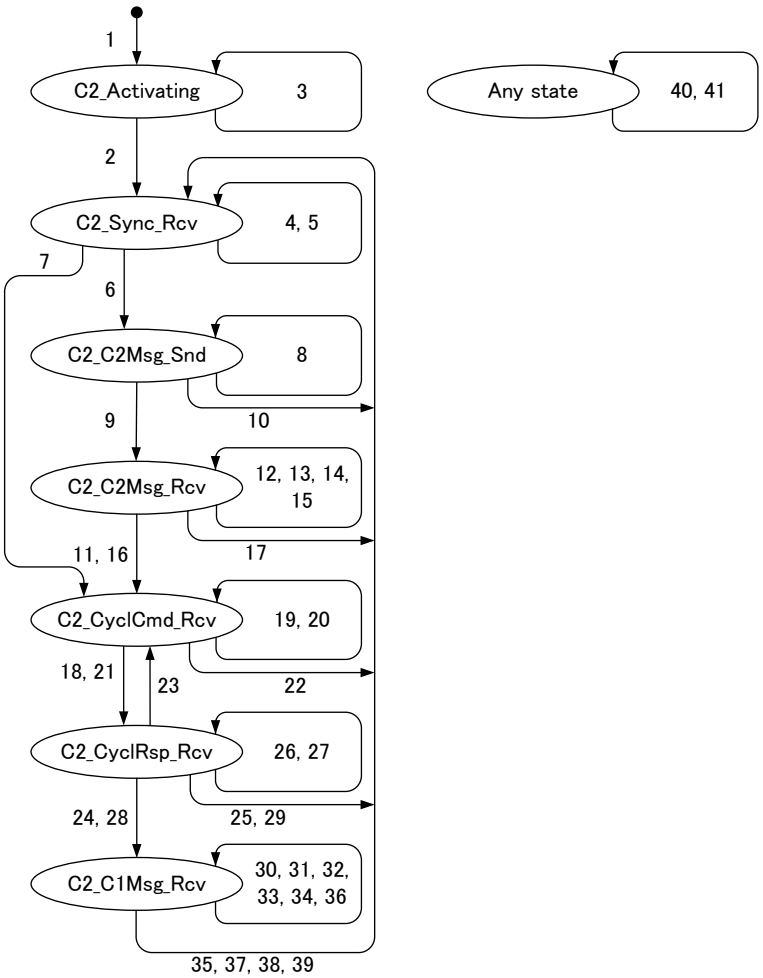


Figure 10 – The state diagram of C2 master for fixed-width time slot

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Table 48 – The state table of C2 master for fixed-width time slot

No.	Current state	Event /condition =>action	Next state
1	Any state	Power on or CTC_Reset.req => (none)	C2_Activating
2	C2_Activating	CTC_Start.req / CHECK_VARS() = True => START_TIMER(T(Tcycle), V(Tcycle)) CTC_Start.cnf { OK }	C2_Sync_Rcv
3	C2_Activating	CTC_Start.req / CHECK_VARS() <> True => CTC_Start.cnf { NG }	C2_Activating
4	C2_Sync_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) <> Cgaddr)) => (none)	C2_Sync_Rcv
5	C2_Sync_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / (Rcv_sts <> OK) => (none)	C2_Sync_Rcv
6	C2_Sync_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" / V(Fc2msg) = “True” => DL_Event.ind {DL_Ev_Tcycle } START_TIMER(T(Tslot), V(Tslot)) V(Nslave) := 1	C2_C2Msg_Snd
7	C2_Sync_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" / V(Fc2msg) = “False” => DL_Event.ind {DL_Ev_Tcycle } START_TIMER(T(Tslot), V(Tslot)) V(Nslave) := 1	C2_Out_Rcv
8	C2_C2Msg_Snd	EXPIRED_TIMER (T(Tslot)) = "True" => EXEC_MSPM_IS(Ec2msg, SndSRCSDU) Node_ID := GET_DA(SndSRCSDU) Length := GET_LEN(SndSRCSDU) SRC_Send_Frame.req { Node_ID, Length, SndSRCSDU}	C2_C2Msg_Snd
9	C2_C2Msg_Snd	SRC_Send_Frame.cnf { } => (none)	C2_C2Msg_Rcv
10	C2_C2Msg_Snd	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot)	C2_Sync_Rcv
11	C2_C2Msg_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && GET_MC(RcvSRCSDU) = Ec2msg)) => EXEC_MSPM_IR(E1msg, Rcv_sts, RcvSRCSDU)	C2_Out_Rcv
12	C2_C2Msg_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && GET_MC(RcvSRCSDU) = Ec1msg))	C2_C2Msg_Rcv

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		=> (none)	
13	C2_C2Msg_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) <> FT_MSG)) => (none)	C2_C2Msg_Rcv
14	C2_C2Msg_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) <> V(MA))) => (none)	C2_C2Msg_Rcv
15	C2_C2Msg_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / (Rcv_sts <> OK) => (none)	C2_C2Msg_Rcv
16	C2_C2Msg_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" => (none)	C2_Out_Rcv
17	C2_C2Msg_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot))	C2_Sync_Rcv
18	C2_Out_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_FT(RcvSRCSDU) = FT_OUT)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts)	C2_In_Rcv
19	C2_Out_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_FT(RcvSRCSDU) <> FT_OUT)) => (none)	C2_Out_Rcv
20	C2_Out_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / (Rcv_sts <> OK) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts)	C2_Out_Rcv
21	C2_Out_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" => (none)	C2_In_Rcv
22	C2_Out_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot))	C2_Sync_Rcv
23	C2_In_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_FT(RcvSRCSDU) = FT_IN) && (V(Nslave) < V(Nmax_slave))) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts) V(Nslave) := V(Nslave) + 1	C2_Out_Rcv
24	C2_In_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_FT(RcvSRCSDU) = FT_IN) && (V(Nslave) >= V(Nmax_slave)) && (V(Nmax_retry) > 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts) V(Nslave) := V(Nslave) + 1 V(Nretry) := V(Nmax_retry)	C2_C1Msg_Rcv
25	C2_In_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_FT(RcvSRCSDU) = FT_IN) && (V(Nslave) >= V(Nmax_slave)) && (V(Nmax_retry) <= 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts)	C2_Sync_Rcv

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		V(Nslave) := V(Nslave) + 1	
26	C2_In_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_FT(RcvSRCSDU) <> FT_IN)) => (none)	C2_In_Rcv
27	C2_In_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts <> OK) && (V(Nslave) < V(Nmax_slave))) => (none)	C2_In_Rcv
28	C2_In_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts <> OK) && (V(Nslave) >= V(Nmax_slave)) && (V(Nmax_retry) > 0)) => V(Nretry) := V(Nmax_retry)	C2_C1Msg_Rcv
29	C2_In_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts <> OK) && (V(Nslave) >= V(Nmax_slave)) && (V(Nmax_retry) <= 0)) => (none)	C2_Sync_Rcv
30	C2_C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && (GET_MC(RcvSRCSDU) = Ec1msg)) => EXEC_MSPM_R(Ec1msg, Rcv_sts, RcvSRCSDU, SndSRCSDU) Node_ID := GET_DA(SndSRCSDU) Length := GET_LEN(SndSRCSDU) SRC_Send_Frame.req { Node_ID, Length, SRCSDU}	C2_C1Msg_Rcv
31	C2_C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && (GET_MC(RcvSRCSDU) = Ec2msg)) => (none)	C2_C1Msg_Rcv
32	C2_C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) <> FT_MSG)) => (none)	C2_C1Msg_Rcv
33	C2_C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) <> V(MA))) => (none)	C2_C1Msg_Rcv
34	C2_C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / (Rcv_sts <> OK) => (none)	C2_C1Msg_Rcv
35	C2_C1Msg_Rcv	SRC_Send_Frame.cnf { } => (none)	C2_Sync_Rcv
36	C2_C1Msg_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / V(Nretry) > 0 => V(Nretry) := V(Nmax_retry) – 1	C2_C1Msg_Rcv
37	C2_C1Msg_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / V(Nretry) <= 0 => STOP_TIMER(T(Tslot)	C2_Sync_Rcv

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
38	C2_C1Msg_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot))	C2_Sync_Rcv
39	Any state	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = Cgaddr) && (GET_FT(RcvSRCSDU) = FT_SYNC)) => DL_Event.ind {DL_Ev_Tcycle } RESTART_TIMER(T(Tcycle)) START_TIMER(T(Tslot), V(Tslot)) V(Nslave) := 1	C2_Sync_Rcv
40	Any state	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = Cgaddr) && (GET_FT(RcvSRCSDU) <> FT_SYNC)) => (none)	Same state
41	Any state	DL-READ-DATA.req(SAP_ID) => Result := GET_CYC_DATA(SAP_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	Same state

6.2.3.2.1.3 Slave

Figure 11 and Table 49 show the state diagram and the state table of slave that adopts fixed-width time slot.

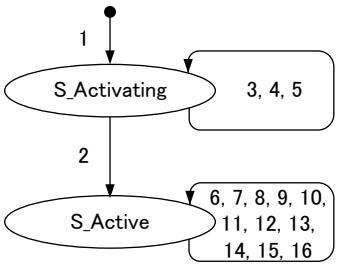


Figure 11 – The state diagram of slave for fixed-width time slot

Table 49 – The state table of slave for fixed-width time slot

No.	Current state	Event /condition =>action	Next state
1	Any states	Power on or CTC_Reset.req => (none)	S_Activating
2	S_Activating	CTC_Start.req / CHECK_VARS() = True => START_TIMER(T(Tcycle), V(Tcycle)) CTC_Start.cnf { OK }	S_Active
3	S_Activating	CTC_Start.req / CHECK_VARS() <> True => CTC_Start.cnf { NG }	Activating

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
4	S_Activating	DL-WRITE-DATA.req(SAP_ID, DLSDU) => Result := SET_CYC_DATA(SAP_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	S_Activating
5	S_Activating	DL-READ-DATA.req(SAP_ID) => Result := GET_CYC_DATA(SAP_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	S_Activating
6	S_Active	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle }	S_Active
7	S_Active	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = Cgaddr) && (GET_FT(RcvSRCSDU) = FT_SYNC)) => DL_Event.ind {DL_Ev_Tcycle } RESTART_TIMER(T(Tcycle))	S_Active
8	S_Active	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = Cgaddr) && (GET_FT(RcvSRCSDU) <> FT_SYNC)) => (none)	S_Active
9	S_Active	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_OUT)) => SndSRCSDU := BUILD_PDU_IN(1) Node_ID := V(MA) Length := GET_LEN(SndSRCSDU) SRC_Send_Frame.req { Node_ID, Length, SndSRCSDU} STORE_PDU_OUT(1, RcvSRCSDU, Rcv_sts)	S_Active
10	S_Active	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && GET_MC(RcvSRCSDU) = Ec1msg)) => EXEC_MSPM_R(Ec1msg, Rcv_sts, RcvSRCSDU, SndSRCSDU) Node_ID := MA Length := GET_LEN(SndSRCSDU) SRC_Send_Frame.req { Node_ID, Length, SndSRCSDU}	S_Active
11	S_Active	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && GET_MC(RcvSRCSDU) = Ec2msg)) => EXEC_MSPM_R(Ec2msg, Rcv_sts, RcvSRCSDU, SndSRCSDU) Node_ID := MA Length := GET_LEN(SndSRCSDU) SRC_Send_Frame.req { Node_ID, Length, SndSRCSDU}	S_Active
12	S_Active	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) <> FT_MSG) && (GET_FT(RcvSRCSDU) <> FT_OUT)) => (none)	S_Active

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
13	S_Active	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) <> V(MA))) => (none)	S_Active
14	S_Active	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / (Rcv_sts <> OK) => (none)	S_Active
15	S_Active	DL-WRITE-DATA.req(SAP_ID, DLSDU) => Result := SET_CYC_DATA(SAP_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	S_Active
16	S_Active	DL-READ-DATA.req(SAP_ID) => Result := GET_CYC_DATA(SAP_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	S_Active

6.2.3.2.2 Protocol machine for cyclic communication consists of configurable time slot

6.2.3.2.2.1 C1 master

Figure 12 and Table 50 show the state diagram and the state table of C1 master that adopts configurable time slot.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

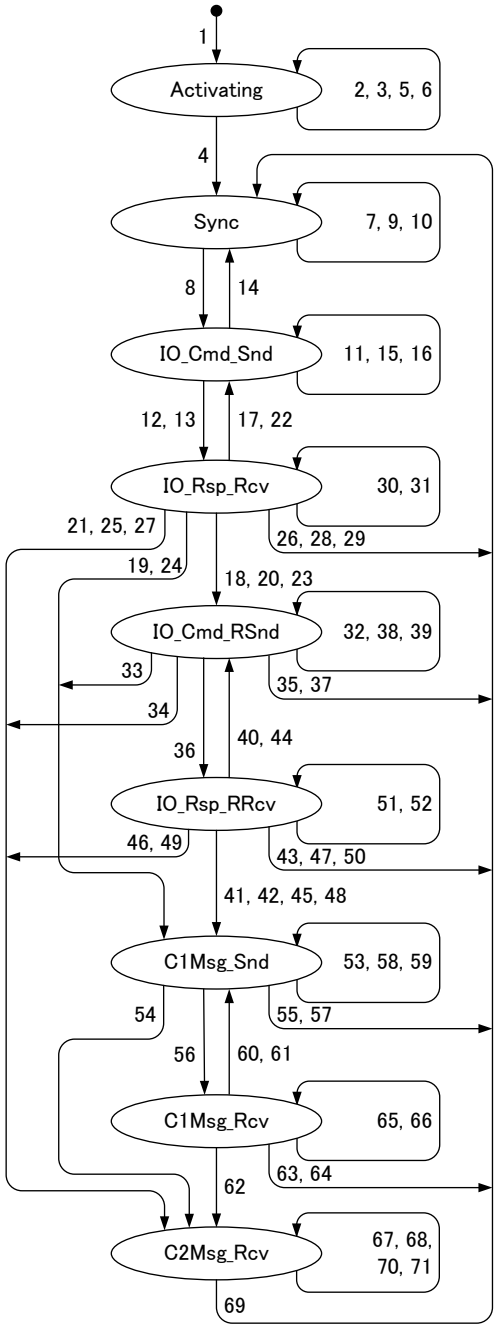


Figure 12 – The state diagram of C1 master for configurable time slot

Table 50 – The state table of C1 master for configurable time slot

No.	Current state	Event /condition =>action	Next state
1	Any state	Power on or CTC_Reset.req => (none)	Activating
2	Activating	CTC_Set_Par.req { Par_ID, Val } => CTC_SET_PAR(Par_ID, Val, Result) CTC_Set_Par.cnf { Result }	Activating
3	Activating	CTC_Get_Par.req { Par_ID } => CTC_GET_PAR(Par_ID, Val, Result)	Activating

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		CTC_Get_Par.cnf { Result, Par_ID, Val }	
4	Activating	CTC_Start.req => START_TIMER(T(Tcycle), V(Tcycle)) CTC_Start.cnf	Sync
5	Activating	DL-WRITE-DATA.req(SAP_ID, DLSDU) => Result := SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	Activating
6	Activating	DL-READ-DATA.req(SAP_ID) => Result := GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	Activating
7	Sync	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } SRCSDU := BUILD_PDU_SYNC() SRC_Send_Frame.req {SRCSDU} V(Nslave) := 0 START_TIMER(T(Tslot), V(Tslot(V(Nslave))))	Sync
8	Sync	SRC_Send_Frame.cnf { } => V(Nslave) := 1 V(Nretry) := 0 V(Nrest_slot) := 0	IO_Cmd_Snd
9	Sync	DL-WRITE-DATA.req(SAP_ID, DLSDU) => Result := SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	Sync
10	Sync	DL-READ-DATA.req(SAP_ID) => Result := GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	Sync
11	IO_Cmd_Snd	EXPIRED_TIMER (T(Tslot)) = "True" => SRCSDU := BUILD_PDU_CYCC(V(Nslave)) SRC_Send_Frame.req {SRCSDU} STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot), V(Tslot(V(Nslave))))	IO_Cmd_Snd
12	IO_Cmd_Snd	SRC_Send_Frame.cnf { } / V(Nslave) < V(Nmax_slave) => (none)	IO_Rsp_Rcv
13	IO_Cmd_Snd	SRC_Send_Frame.cnf { } / V(Nslave) >= V(Nmax_slave) => V(Nrest_slot) := V(Nmax_retry)	IO_Rsp_Rcv
14	IO_Cmd_Snd	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle} STOP_TIMER(T(Tslot)) SRCSDU := BUILD_PDU_SYNC() SRC_Send_Frame.req {SRCSDU} V(Nslave) := 0 START_TIMER(T(Tslot), V(Tslot(V(Nslave))))	Sync

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
15	IO_Cmd_Snd	DL-WRITE-DATA.req(SAP_ID, DLSDU) => result := SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	IO_Cmd_Snd
16	IO_Cmd_Snd	DL-READ-DATA.req(SAP_ID) => result := GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	IO_Cmd_Snd

17	IO_Rsp_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU}
		/ ((Rcv_sts = OK) && V(Nslave) < V(Nmax_slave))
		=> STORE_PDU_CYCR(V(Nslave), SRCSDU, Rcv_sts)
		If ((Rcv_sts <> OK) && (V(Nrest_slot) > 0)) then
		PUT_RETRY_LIST(V(Nslave))
		Endif
		V(Nslave) := V(Nslave) + 1
			IO_Cmd_Snd

18	IO_Rsp_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU}
		/ ((Rcv_sts = OK) && (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) > 0) && (V(Nretry) > 0))
		=> STORE_PDU_CYCR(V(Nslave), SRCSDU, Rcv_sts)
		If (Result <> OK) then
		PUT_RETRY_LIST(V(Nslave))
		Endif
			IO_Cmd_RSnd

19	IO_Rsp_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) > 0) && (V(Nretry) = 0)) => STORE_PDU_CYCR(V(Nslave), SRCSDU, Rcv_sts)	C1Msg_Snd
20	IO_Rsp_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) > 0) && (V(Nretry) = 0)) => STORE_PDU_CYCR(V(Nslave), SRCSDU, Rcv_sts) PUT_RETRY_LIST(V(Nslave))	IO_Cmd_RSnd
21	IO_Rsp_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) <= 0)) => STORE_PDU_CYCR(V(Nslave), SRCSDU, Result) SND_MSG_TOKEN()	C2Msg_Rcv

22	IO_Rsp_Rcv	EXPIRED_TIMER (T(Tslot)) = "True"
		/ V(Nslave) < V(Nmax_slave)
		=> STORE_PDU_CYCR(V(Nslave), SRCSDU, NG)
		If V(Nrest_slot) > 0 then
		PUT_RETRY_LIST(V(Nslave))
		Endif
		V(Nslave) := V(Nslave) + 1
			IO_Cmd_Snd

23	IO_Rsp_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) > 0) && CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) > 0 => STORE_PDU_CYCR(V(Nslave), SRCSDU, NG)	IO_Cmd_RSnd

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		PUT_RETRY_LIST(V(Nslave)) V(Nslave) := GET_RETRY_LIST() SRCSDU := BUILD_PDU_CYCC(V(Nslave)) SRC_Send_Frame.req {SRCSDU} STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot), V(Tslot(V(slave))))	
24	IO_Rsp_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) > 0) && CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) = 0 && CHECK_MSG_EN(C1) <> 0 => STORE_PDU_CYCR(V(Nslave), SRCSDU, NG) EXEC_MSGC_SEG(Ec1msg, OK, None, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU } slot := GET_MSG_SLOT(C1) STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot),slot))	C1Msg_Snd
25	IO_Rsp_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) > 0) && CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) = 0 && CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) <> 0 => STORE_PDU_CYCR(V(Nslave), SRCSDU, NG) SND_MSG_TOKEN()	C2Msg_Rcv
26	IO_Rsp_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) > 0) && CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) = 0 && CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) = 0 => STORE_PDU_CYCR(V(Nslave), SRCSDU, NG)	Sync
27	IO_Rsp_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) <= 0) && CHECK_MSG_EN(C2) <> 0 => STORE_PDU_CYCR(V(Nslave), SRCSDU, NG) SND_MSG_TOKEN()	C2Msg_Rcv
28	IO_Rsp_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) <= 0) && CHECK_MSG_EN(C2) = 0 => STORE_PDU_CYCR(V(Nslave), SRCSDU, NG)	Sync
29	IO_Rsp_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot)) STORE_PDU_CYCR(V(Nslave), SRCSDU, NG) SRCSDU := BUILD_PDU_SYNC() SRC_Send_Frame.req {SRCSDU}	Sync

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		V(Nslave) := 0 START_TIMER(T(Tslot), V(Tslot(V(Nslave))))	
30	IO_Rsp_Rcv	DL-WRITE-DATA.req(SAP_ID, DLSDU) => result := SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	IO_Rsp_Rcv
31	IO_Rsp_Rcv	DL-READ-DATA.req(SAP_ID) => result := GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	IO_Rsp_Rcv
32	IO_Cmd_RSnd	EXPIRED_TIMER (T(Tslot)) = "True" / CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) > 0 => V(Nslave) := GET_RETRY_LIST() SRCSDU := BUILD_PDU_CYCC(V(Nslave)) SRC_Send_Frame.req {SRCSDU} STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot), V(Tslot(V(slave))))	IO_Cmd_RSnd
33	IO_Cmd_RSnd	EXPIRED_TIMER (T(Tslot)) = "True" / CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) > 0 && CHECK_MSG_EN(C1) <> 0 =>EXEC_MSGC_SEG(Ec1msg, OK, None, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU } slot := GET_MSG_SLOT(C1) STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot),slot))	C1Msg_Snd
34	IO_Cmd_RSnd	EXPIRED_TIMER (T(Tslot)) = "True" / CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) > 0 && CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) <> 0 => SND_MSG_TOKEN()	C2Msg_Rcv
35	IO_Cmd_RSnd	EXPIRED_TIMER (T(Tslot)) = "True" / CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) > 0 && CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) = 0 =>(none)	Sync
36	IO_Cmd_RSnd	SRC_Send_Frame.cnf { } => (none)	IO_Rsp_RRcv
37	IO_Cmd_RSnd	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot)) SRCSDU := BUILD_PDU_SYNC() SRC_Send_Frame.req {SRCSDU} V(Nslave) := 0 START_TIMER(T(Tslot), V(Tslot(V(Nslave))))	Sync
38	IO_Cmd_RSnd	DL-WRITE-DATA.req(SAP_ID, DLSDU) => result := SET_CYC_DATA(SAP_ID, Node_ID, DLSDU)	IO_Cmd_RSnd

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		DL-WRITE-DATA.cnf (Result)	
39	IO_Cmd_RSnd	DL-READ-DATA.req(SAP_ID) => result := GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	IO_Cmd_RSnd
40	IO_Rsp_RRcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / (V(Nrest_slot) > 1) && (V(Nretry) > 0) => STORE_PDU_CYCR(V(Nslave), SRCSDU, Result) V(Nrest_slot) := V(Nrest_slot) – 1	IO_Cmd_RSnd
41	IO_Rsp_RRcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / (V(Nrest_slot) > 1) && (V(Nretry) <= 0) => STORE_PDU_CYCR(V(Nslave), SRCSDU, Result)	C1Msg_Snd
42	IO_Rsp_RRcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / (V(Nrest_slot) <= 1) => STORE_PDU_CYCR(V(Nslave), SRCSDU, Result)	C1Msg_Snd
43	IO_Rsp_RRcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot) STORE_PDU_CYCR(V(Nslave), SRCSDU, NG) SRCSDU := BUILD_PDU_SYNC() SRC_Send_Frame.req {SRCSDU} V(Nslave) := 0 START_TIMER(T(Tslot), V(Tslot(V(Nslave))))	Sync
44	IO_Rsp_RRcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nrest_slot) > 1) && CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) > 0 => STORE_PDU_CYCR(V(Nslave), SRCSDU, NG) V(Nrest_slot) := V(Nrest_slot) – 1 V(Nslave) := GET_RETRY_LIST() SRCSDU := BUILD_PDU_CYCC(V(Nslave)) SRC_Send_Frame.req {SRCSDU}	IO_Cmd_RSnd
45	IO_Rsp_RRcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nrest_slot) <= 1) && CHECK_MSG_EN(C1) <> 0 =>EXEC_MSGC_SEG(Ec1msg, OK, None, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU } slot := GET_MSG_SLOT(C1) STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot),slot))	C1Msg_Snd
46	IO_Rsp_RRcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nrest_slot) <= 1) && CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) <> 0 => SND_MSG_TOKEN()	C2Msg_Rcv
47	IO_Rsp_RRcv	EXPIRED_TIMER (T(Tslot)) = "True"	Sync

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		/ (V(Nrest_slot) <= 1) && CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) = 0 =>(none)	
48	IO_Rsp_RRcv	EXPIRED_TIMER (T(Tslot)) = "True" /CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) = 0 && CHECK_MSG_EN(C1) <> 0 =>EXEC_MSGC_SEG(Ec1msg, OK, None, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU } slot := GET_MSG_SLOT(C1) STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot),slot))	C1Msg_Snd
49	IO_Rsp_RRcv	EXPIRED_TIMER (T(Tslot)) = "True" /CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) = 0 && CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) <> 0 => SND_MSG_TOKEN()	C2Msg_Rcv
50	IO_Rsp_RRcv	EXPIRED_TIMER (T(Tslot)) = "True" /CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) = 0 && CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) = 0 =>(none)	Sync
51	IO_Rsp_RRcv	DL-WRITE-DATA.req(SAP_ID, DLSDU) => result := SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	IO_Rsp_RRcv
52	IO_Rsp_RRcv	DL-READ-DATA.req(SAP_ID) => result := GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	IO_Rsp_RRcv
53	C1Msg_Snd	EXPIRED_TIMER (T(Tslot)) = "True" / CHECK_MSG_EN(C1) <> 0 => EXEC_MSGC_SEG(Ec1msg, OK, None, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU } slot := GET_MSG_SLOT(C1) STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot),slot))	C1Msg_Snd
54	C1Msg_Snd	EXPIRED_TIMER (T(Tslot)) = "True" / CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) <> 0 => SND_MSG_TOKEN()	C2Msg_Rcv
55	C1Msg_Snd	EXPIRED_TIMER (T(Tslot)) = "True" / CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) = 0 =>(none)	Sync
56	C1Msg_Snd	SRC_Send_Frame.cnf { }	C1Msg_Rcv

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		=> (none)	
57	C1Msg_Snd	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot) SRCSDU := BUILD_PDU_SYNC() SRC_Send_Frame.req {SRCSDU} V(Nslave) := 0 START_TIMER(T(Tslot), V(Tslot(V(Nslave))))	Sync
58	C1Msg_Snd	DL-WRITE-DATA.req(SAP_ID, DLSDU) => result := SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	C1Msg_Snd
59	C1Msg_Snd	DL-READ-DATA.req(SAP_ID) => result := GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	C1Msg_Snd
60	C1Msg_Rcv	SRC_Recv_Frame.ind {Result, RcvSRCSDU} => STOP_TIMER(T(Tslot)) EXEC_MSGC_SEG(Ec1msg, Result, RcvSRCSDU, None)	C1Msg_Snd
61	C1Msg_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / CHECK_MSG_EN(C1) <> 0 => EXEC_MSGC_SEG(Ec1msg, OK, None, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU } slot := GET_MSG_SLOT(C1) STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot),slot))	C1Msg_Snd
62	C1Msg_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) <> 0 => SND_MSG_TOKEN()	C2Msg_Rcv
63	C1Msg_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) = 0 =>(none)	Sync
64	C1Msg_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot)) SRCSDU := BUILD_PDU_SYNC() SRC_Send_Frame.req {SRCSDU} V(Nslave) := 0 START_TIMER(T(Tslot), V(Tslot(V(Nslave))))	Sync
65	C1Msg_Rcv	DL-WRITE-DATA.req(SAP_ID, DLSDU) => result := SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	C1Msg_Rcv
66	C1Msg_Rcv	DL-READ-DATA.req(SAP_ID) => result := GET_CYC_DATA(SAP_ID, Node_ID, DLSDU)	C1Msg_Rcv

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		DL-READ-DATA.cnf (Result, DLSDU)	
67	C2Msg_Rcv	SRC_Recv_Frame.ind { Result, RcvSRCSDU } => EXEC_MSGS_SEG(Ec2msg, Result, RcvSRCSDU, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU }	C2Msg_Rcv
68	C2Msg_Rcv	SRC_Send_Frame.cnf { } => (none)	C2Msg_Rcv
69	C2Msg_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot)) SRCSDU := BUILD_PDU_SYNC() SRC_Send_Frame.req {SRCSDU} V(Nslave) := 0 START_TIMER(T(Tslot), V(Tslot(V(Nslave))))	Sync
70	C2Msg_Rcv	DL-WRITE-DATA.req(SAP_ID, DLSDU) => result := SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	C2Msg_Rcv
71	C2Msg_Rcv	DL-READ-DATA.req(SAP_ID) => result := GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	C2Msg_Rcv

6.2.3.2.2.2 C2Master

Figure 13 and Table 51 show the state diagram and the state table of C2 master hat adopts configurable time slot.

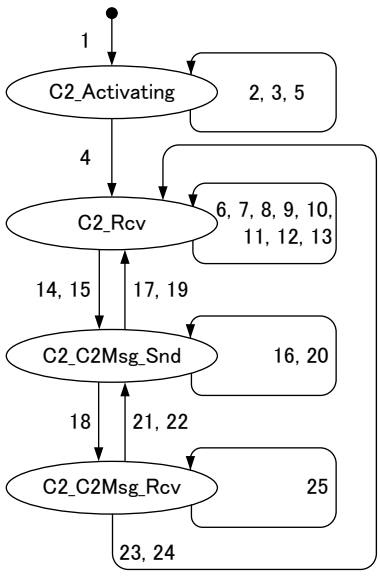


Figure 13 – The state diagram of C2 master for configurable time slot

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Table 51 – The state table of C2 master for configurable time slot

No.	Current state	Event /condition =>action	Next state
1	Any state	Power on or CTC_Reset.req => (none)	C2_Activating
2	C2_Activating	CTC_Set_Par.req { Par_ID, Val } => CTC_SET_PAR(Par_ID, Val, Result) CTC_Set_Par.req { Result }	C2_Activating
3	C2_Activating	CTC_Get_Par.req { Par_ID } => CTC_GET_PAR(Par_ID, Val, Result) CTC_Get_Par.cnf { Result, Par_ID, Val }	C2_Activating
4	C2_Activating	CTC_Start.req => START_TIMER(T(Tcycle), V(Tcycle)) START_TIMER(T(Tmsg), V(Tmsg)) CTC_Start.cnf	C2_Rcv
5	C2_Activating	DL-READ-DATA.req(SAP_ID) => result := GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	C2_Activating
6	C2_Rcv	SRC_Recv_Frame.ind { Result, SRCSDU } / (SRCSDU.TYPLEN = TYP1) => DL_Event.ind {DL_Ev_Tcycle } RESTART_TIMER(T(Tcycle)) RESTART_TIMER(T(Tmsg))	C2_Rcv
7	C2_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tcycle)) STOP_TIMER(T(Tmsg)) START_TIMER(T(Tcycle), V(Tcycle)) START_TIMER(T(Tmsg), V(Tmsg))	C2_Rcv
8	C2_Rcv	SRC_Recv_Frame.ind { Result, RcvSRCSDU } / RcvSRCSDU. SA = C1 RcvSRCSDU. TYPLEN = TYP2 => V(Nslave) := EX_ADD(DA) STORE_PDU_CYCC(V(Nslave), SRCSDU, Result)	C2_Rcv
9	C2_Rcv	SRC_Recv_Frame.ind { Result, RcvSRCSDU } / RcvSRCSDU. SA = Slave RcvSRCSDU. TYPLEN = TYP2 => V(Nslave) := EX_ADD(DA) STORE_PDU_CYCR(V(Nslave), SRCSDU, Result)	C2_Rcv
10	C2_Rcv	SRC_Recv_Frame.ind { Result, RcvSRCSDU } / SRCSDU. SA = C1 SRCSDU.TYPLEN = TYP12 => EXEC_MSGS_SEG(Ec1msg, Result, RcvSRCSDU, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU }	C2_Rcv
11	C2_Rcv	SRC_Send_Frame.cnf { }	C2_Rcv

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		=> (none)	
12	C2_Rcv	EXPIRED_TIMER (T(Tmsg)) = "True" / CHECK_MSG_EN(C2) = 0 => (none)	C2_Rcv
13	C2_Rcv	DL-READ-DATA.req(SAP_ID) => result := GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	C2_Rcv
14	C2_Rcv	SRC_Recv_Frame.ind { Result, RcvSRCSDU } / RcvSRCSDU. SA = V(MA) RcvSRCSDU. TYPLEN = TYP5 => EXEC_MSGC_SEG(Ec2msg, OK, None, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU } slot := GET_MSG_SLOT(C1) STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot),slot))	C2_C2Msg_Snd
15	C2_Rcv	EXPIRED_TIMER (T(Tmsg)) = "True" / CHECK_MSG_EN(C2) <> 0 => EXEC_MSGC_SEG(Ec2msg, OK, None, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU } slot := GET_MSG_SLOT(C1) STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot),slot))	C2_C2Msg_Snd
16	C2_C2Msg_Snd	EXPIRED_TIMER (T(Tslot)) = "True" / CHECK_MSG_EN(C2) <> 0 => EXEC_MSGC_SEG(Ec2msg, OK, None, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU } slot := GET_MSG_SLOT(C1) STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot),slot))	C2_C2Msg_Snd
17	C2_C2Msg_Snd	EXPIRED_TIMER (T(Tslot)) = "True" / CHECK_MSG_EN(C2) = 0 => (none)	C2_Rcv
18	C2_C2Msg_Snd	SRC_Send_Frame.cnf { } => (none)	C2_C2Msg_Rcv
19	C2_C2Msg_Snd	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot)) STOP_TIMER(T(Tcycle)) STOP_TIMER(T(Tmsg)) START_TIMER(T(Tcycle), V(Tcycle)) START_TIMER(T(Tmsg), V(Tmsg))	C2_Rcv
20	C2_C2Msg_Snd	DL-READ-DATA.req(SAP_ID) => result := GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	C2_C2Msg_Snd

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
21	C2_C2Msg_Rcv	SRC_Recv_Frame.ind { Result, RcvSRCSDU } => EXEC_MSGC_SEG(Ec2msg, Result, RcvSRCSDU, None)	C2_C2Msg_Snd
22	C2_C2Msg_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / CHECK_MSG_EN(C2) <> 0 => EXEC_MSGC_SEG(Ec2msg, OK, None, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU } slot := GET_MSG_SLOT(C1) STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot),slot))	C2_C2Msg_Snd
23	C2_C2Msg_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / CHECK_MSG_EN(C2) = 0 => (none)	C2_Rcv
24	C2_C2Msg_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot)) STOP_TIMER(T(Tcycle)) STOP_TIMER(T(Tmsg)) START_TIMER(T(Tcycle), V(Tcycle)) START_TIMER(T(Tmsg), V(Tmsg))	C2_Rcv
25	C2_C2Msg_Rcv	DL-READ-DATA.req(SAP_ID) => result := GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	C2_C2Msg_Rcv

6.2.3.2.2.3 Slave

Figure 14 and Table 52 show the state diagram and the state table of slave which adopts configurable time slot.

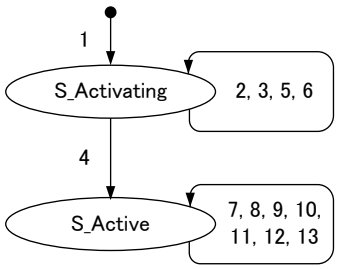


Figure 14 – The state diagram of slave for configurable time slot

Table 52 – The state table of slave for configurable time slot

No.	Current state	Event /condition =>action	Next state
1	Any states	Power on or CTC_Reset.req => (none)	S_Activating
2	S_Activating	CTC_Set_Par.req { Par_ID, Val } => CTC_SET_PAR(Par_ID, Val, Result)	S_Activating

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		CTC_Set_Par.req { Result }	
3	S_Activating	CTC_Get_Par.req { Par_ID } => CTC_GET_PAR(Par_ID, Val, Result) CTC_Get_Par.cnf { Result, Par_ID, Val }	S_Activating
4	S_Activating	CTC_Start.req => START_TIMER(T(Tcycle), V(Tcycle)) CTC_Start.cnf	S_Active
5	S_Activating	DL-WRITE-DATA.req(SAP_ID, DLSDU) => result := SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	S_Activating
6	S_Activating	DL-READ-DATA.req(SAP_ID) => result := GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	S_Activating
7	S_Active	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle }	S_Active
8	S_Active	SRC_Recv_Frame.ind { Result, RcvSRCSDU } / (SRCSDU.TYPLEN = TYP1) => DL_Event.ind {DL_Ev_Tcycle } RESTART_TIMER(T(Tcycle))	S_Active
9	S_Active	SRC_Recv_Frame.ind { Result, RcvSRCSDU } / RcvSRCSDU.DA = V(MA) && RcvSRCSDU. TYPLEN = TYP2 => STORE_PDU_CYCC(1, SRCSDU, Result) SndSRCSDU := BUILD_PDU_CYCR(V(MA)) SRC_Send_Frame.req { SndSRCSDU}	S_Active
10	S_Active	SRC_Recv_Frame.ind { Result, RcvSRCSDU } / RcvSRCSDU.DA = V(MA) RcvSRCSDU.SA = C1 && RcvSRCSDU. TYPLEN = TYP12 => EXEC_MSGS_SEG(Ec1msg, Result, RcvSRCSDU, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU }	S_Active
11	S_Active	SRC_Recv_Frame.ind { Result, RcvSRCSDU } / RcvSRCSDU.DA = V(MA) RcvSRCSDU.SA = C2 && RcvSRCSDU. TYPLEN = TYP12 => EXEC_MSGS_SEG(Ec2msg, Result, RcvSRCSDU, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU }	S_Active
12	S_Active	DL-WRITE-DATA.req(SAP_ID, DLSDU) => result := SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	S_Active
13	S_Active	DL-READ-DATA.req(SAP_ID) => result := GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	S_Active

6.2.3.2.3 Functions used by cyclic transmission machine

All the functions used by the fixed-width time slot protocol machine are summarized in Table 53.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Table 53 – The list of functions used by cyclic transmission machine

Function name	Parameter		Return	Operation
	Input	Output	Value	
BUILD_PDU_SYNC	none	None	SRCSDU	This function builds SRCSDU to request SRC to send synchronous frame.
BUILD_PDU_OUT	Slave_index, DLSDU	None	SRCSDU	This function builds SRCSDU to request SRC to output data frame. Slave_Index is the number of slave and its range is 1 to V(Nmax_slave).
BUILD_PDU_IN	Slave_index, DLSDU	None	SRCSDU	This function builds SRCSDU to request SRC to input data frame. Slave_Index is the number of slave and its range is 1 to V(Nmax_slave).
STORE_PDU_OUT	Slave_index, SRCSDU, Rcv_Sts	None	None	This function stores SRCSDU retrieved from SRC and receives status Rcv_Sts as received output data. Slave_Index is the number of slave and its range is 1 to V(Nmax_slave).
STORE_PDU_IN	Slave_index, SRCSDU, Rcv_Sts	None	None	This function stores SRCSDU retrieved from SRC and receives status Rcv_Sts as received input data. Slave_Index is the number of slave and its range is 1 to V(Nmax_slave).
PUT_RETRY_LIST	Slave_index	None	None	This function registers the slave number specified with Slave_index to retry list, and increments the value of V(Nretry_list) that contains the number of registered slave number.
GET_RETRY_LIST	none	None	Slave_index	This function retrieves a slave number from retry list, and decrements the value of V(Nretry_list) that contains the number of registered slave number.
EXEC_MSGC_SEG	Fc1c2, Rcv_sts, RcvSRCSDU	SndSRCSDU	none	This function calls the message segmentation machine for the initiator of message communication. The parameter Fc1c2 is flag to specify C1 message or C2 message. Rcv_sts and RcvSRCSDU are receive status and received SRCSDU respectively that has been passed by SRC. SndSRCSDU is SRCSDU to be sent.
EXEC_MSGS_SEG	Fc1c2, Rcv_sts, RcvSRCSDU	SndSRCSDU	none	This function calls the message segmentation machine for the responder of message communication. The parameters of this function are same as the function EXEC_MSGC_SEG.
EXEC_MSPM_IR	Ec1c2, Rcv_sts, RcvSRCSDU	None	None	This function calls the message segmentation protocol machine for the initiator of message communication to process a received frame. The parameters of this function are same as the parameters with the same name of the function EXEC_MSPM_R.
START_TIMER	Tim_ID, Val	None		This function start the timer specified with Tim_ID in set value Val. The timer is auto-reload type, which load

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Function name	Parameter		Return	Operation
	Input	Output	Value	
				the set value again when the timer is timed up.
STOP_TIMER	Tim_ID			This function stop the timer specified with Tim_ID.
RESTART_TIMER	Tim_ID			This function restart the timer specified with Tim_ID with the set value specified by START_TIMER.
EXPIRED_TIMER	Tim_ID			This function indicates the status of the timer specified with Tim_ID.
CTC_SET_PAR	Var_ID, Val			This function updates the variable specified with Var_ID in the DLE in the value specified with Val.
CTC_GET_PAR	Var_ID	Val		This function reads the current value of the variable specified with Var_ID in the DLE.
SET_CYC_DATA	SRCSDU			This function updates the send buffer for the I/O data exchange that exists in CTC in the specified data.
GET_CYC_DATA	SRCSDU			This function updates the receive buffer for the I/O data exchange that exists in CTC in the specified data.
CMP_RTIM	Tim_ID, Val			This function compares the remainder time of the timer specified with Tim_ID with set value Val. If the remainder time is large, True is returned and if it is small, False is returned.
CHECK_MSG_EN	Msg_mst	Enable_flag		This function judges whether the message communication that the station specified with Msg_mst operates as an initiator may be executed. When the following three conditions are satisfied, enable_flag is turned on: - The message communication band is configured; - DLS-user issued a request; - Time for message communication remains.
GET_MSG_SLOT	Msg_mst	msg_slot		This function outputs the period of time slot for the message communication that the station specified with Msg_mst operates as an initiator.
EX_ADD	Node_address	Slave_index		This function converts the station address to slave number.
SND_MSG_TOKEN				This function transmits the token frame when time that Message Token frame can be transmitted remains by the C2 message beginning time.

6.2.3.3 Message segmentation protocol machine

6.2.3.3.1 General

Message segmentation and assembly specification of the message of Message segmentation machine have two kinds of each frame formats. It is described respectively in the following subclause.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

In the message communication, it is classified into two kinds by the allocated band. One is C1 message communication that the C1 master operates as primary station (initiator) and the slave or the C2 master operates as secondary station (responder). The other is C2 message communication that the C2 master operates a primary station and the slave or the C1 master becomes a secondary station. The operation of the primary station and the secondary station is the same though these are different the allocated band.

Message segmentation protocol machine is executed by the cyclic transmission protocol machine with the function call which name has prefix “EXEC_MSPM_”. In the state table of the following subclause, the event “Call” means the function calls.

6.2.3.3.2 Segmentation for basic format DLPDU

6.2.3.3.2.1 Initiator for basic format DLPDU

Figure 15 and Table 54 show the state diagram and the state table of an initiator of the message segmentation machine when DLE adopts basic format DLPDU.

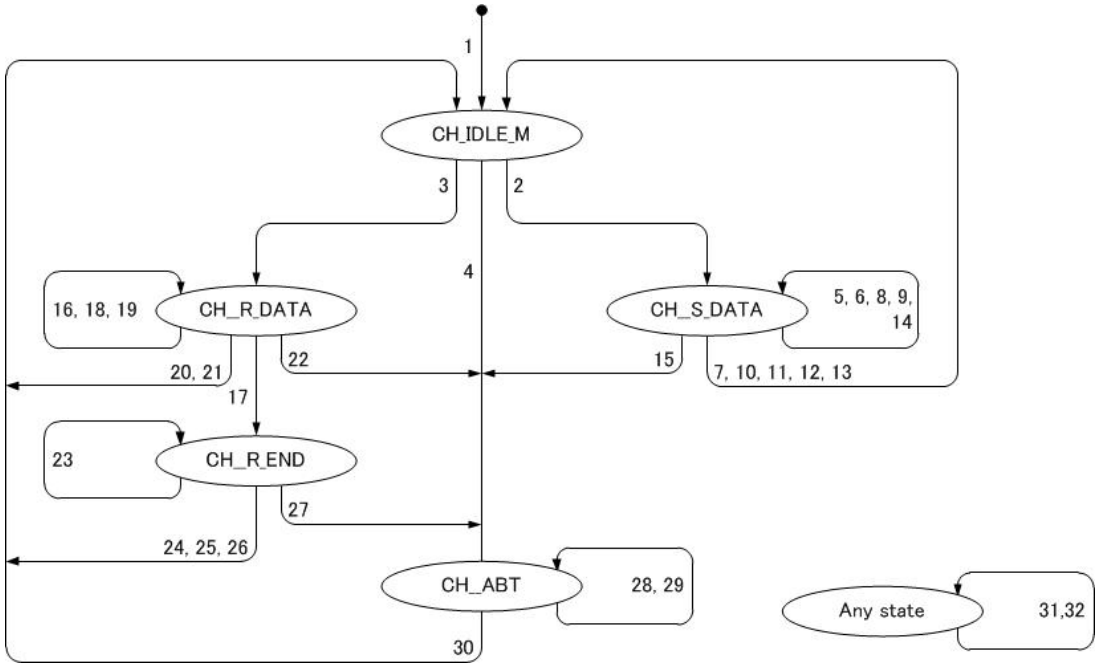


Figure 15 – The state diagram of message initiator for basic format

Table 54 – The state table of message initiator for basic format

No.	Current state	Event /condition =>action	Next state
1	Any states	Power on or CTC_Reset.req => V(Nms_n) := 0 V(Nmr_n) := 0	CH_IDLE_M
2	CH_IDLE_M	DL-SDA.req{SAP_ID,Node_ID,Length,SndDLSDU} => V(Nms_n) := 0 STORE_DLSDU(Ec1c2, Node_ID, SndDLSDU) V(Nmsg_len_n) := Length V(Nmsg_rem_len_n) := Length V(Fmsg_sending_n) := True	CH_S_DATA
3	CH_IDLE_M	EXEC_MSPM_IS(Ec1c2, SndSRCSDU) => V(Nmr_n) := 0 V(Fmsg_sending_n) := False	CH_R_DATA

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		V(Nmp_n) := 1 SndSRCSDU := BUILD_PDU_BMSG(MF_S_RR, 0, V(Nmr_n), V(Nmp_n))	
4	CH_IDLE_M	DL-ABORT-SDA.req { SAP_ID } => V(Fmsg_sending_n) := False	CH_ABT
5	CH_S_DATA	EXEC_MSPM_IS(Ec1c2, SndSRCSDU) / V(Npkt_len_n) >= V(Nmsg_rem_len_n) => V(Nmp_n) = 1 SndSRCSDU := BUILD_PDU_BMSG(MF_I, V(Nms_n), V(Nmr_n), V(Nmp_n))	CH_S_DATA
6	CH_S_DATA	EXEC_MSPM_IS(Ec1c2, SndSRCSDU) / V(Npkt_len_n) < V(Nmsg_rem_len_n) => V(Nmp_n) = 0 SndSRCSDU := BUILD_PDU_BMSG(MF_I, V(Nms_n), V(Nmr_n),V(Nmp_n)) V(Nmsg_rem_len_n) := V(Nmsg_rem_len_n) - V(Npkt_len_n)	CH_S_DATA
7	CH_S_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RR && GET_MC_NR(RcvSRCSDU) = V(Nms_n)+1 && V(Nmp_n) = 1 => V(Nms_n) := V(Nms_n) + 1 DL-SDA.cnf{ SND_OK }	CH_IDLE_M
8	CH_S_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RR && GET_MC_NR(RcvSRCSDU) = V(Nms_n)+1 && V(Nmp_n) = 0 => V(Nms_n) := V(Nms_n) + 1	CH_S_DATA
9	CH_S_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RR && GET_MC_NR(RcvSRCSDU) <> V(Nms_n)+1 => (none)	CH_S_DATA
10	CH_S_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RNR && GET_MC_NR(RcvSRCSDU) = V(Nms_n)+1 && V(Nmp_n) = 1 => V(Nms_n) := V(Nms_n) + 1 DL-SDA.cnf{ SND_OK }	CH_IDLE_M
11	CH_S_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RNR && GET_MC_NR(RcvSRCSDU) = V(Nms_n)+1 && V(Nmp_n) = 0 => DL-SDA.cnf{ SND_BUSY }	CH_IDLE_M
12	CH_S_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RNR	CH_IDLE_M

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		&& GET_MC_NR(RcvSRCSDU) <> V(Nms_n)+1 => DL-SDA.cnf{ SND_BUSY }	
13	CH_S_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_REJ => V(Nms_n) := 0, V(Nmr_n) := 0 DL-SDA.cnf{ SND_ABT }	CH_IDLE_M
14	CH_S_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I => (none)	CH_S_DATA
15	CH_S_DATA	DL-ABORT-SDA.req { SAP_ID } => (none)	CH_ABT
16	CH_R_DATA	EXEC_MSPM_IS(Ec1c2, SndSRCSDU) => V(Nmp_n) := 1 SndSRCSDU := BUILD_PDU_BMSG(MF_S_RR,0,V(Nmr_n),V(Nmp_n))	CH_R_DATA
17	CH_R_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I && GET_MC_NS(RcvSRCSDU) = V(Nmr_n) && GET_MC_F(RcvSRCSDU) = 1 => V(Nmr_n) := V(Nmr_n) + 1 STORE_PDU_MSG(Ec1c2, RcvSRCSDU)	CH_R_END
18	CH_R_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I && GET_MC_NS(RcvSRCSDU) = V(Nmr_n) && GET_MC_F(RcvSRCSDU) = 0 => V(Nmr_n) := V(Nmr_n) + 1 STORE_PDU_BMSG(RcvSRCSDU)	CH_R_DATA
19	CH_R_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I && GET_MC_NS(RcvSRCSDU) <> V(Nmr_n) => (none)	CH_R_DATA
20	CH_R_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG) && { (GET_MC_FMT(RcvSRCSDU) = MF_S_RR) || (GET_MC_FMT(RcvSRCSDU) = MF_S_RNR) } => (none)	CH_IDLE_M
21	CH_R_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_REJ => V(Nms_n) := 0, V(Nmr_n) := 0	CH_IDLE_M
22	CH_R_DATA	DL-ABORT-SDA.req { SAP_ID } => (none)	CH_ABT

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
23	CH_R_END	EXEC_MSPM_IS(Ec1c2, SndSRCSDU) => V(Nmp_n) = 1 SndSRCSDU := BUILD_PDU_BMSG(MF_S_RR,0,V(Nmr_n),V(Nmp_n))	CH_R_END
24	CH_R_END	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && { (GET_MC_FMT(RcvSRCSDU) = MF_S_RR) || (GET_MC_FMT(RcvSRCSDU) = MF_S_RNR) } => Node_ID := GET_SA(RcvSRCSDU) DLSDU := GET_DLSDU(Ec1c2) Length := GET_DLSDU_LEN(DLSDU) DL-SDA.ind{ SAP_ID, Rcv_sts , Node_ID, Length, DLSDU}	CH_IDLE_M
25	CH_R_END	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_REJ => V(Nms_n) := 0, V(Nmr_n) := 0	CH_IDLE_M
26	CH_R_END	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I => (none)	CH_R_END
27	CH_R_END	DL-ABORT-SDA.req { SAP_ID } => (none)	CH_ABT
28	CH_ABT	EXEC_MSPM_IS(Ec1c2, SndSRCSDU) => V(Nmp_n) = 1 SndSRCSDU := BUILD_PDU_BMSG(MF_S_REJ,0,0,V(Nmp_n))	CH_ABT
29	CH_ABT	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && { (GET_MC_FMT(RcvSRCSDU) = MF_S_RR) || (GET_MC_FMT(RcvSRCSDU) = MF_I) } => (none)	CH_ABT

30	CH_ABT	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU)
		
		/ Rcv_sts = OK
		&& GET_FTYP(RcvSRCSDU) = FT_MSG
		&& { (GET_MC_FMT(RcvSRCSDU) = MF_S_RNR) ||
		(GET_MC_FMT(RcvSRCSDU) = MF_S_REJ) }
		
		=> V(Nms_n) := 0, V(Nmr_n) := 0
		DL-ABORT-SDA.cnf
		
		if (V(Fmsg_sending_n) = True) then
		DL-SDA.cnf { SND_ABT }
		endif
			CH_IDLE_M

31	Any state	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) <> FT_MSG => (none)	Same state
32	Any state	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts <> OK => (none)	Same state

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

6.2.3.3.2.2 Responder for basic format DLPDU

Figure 16 and Table 55 show the state diagram and the state table of a responder of the message segmentation machine when DLE adopts basic format DLPLDU.

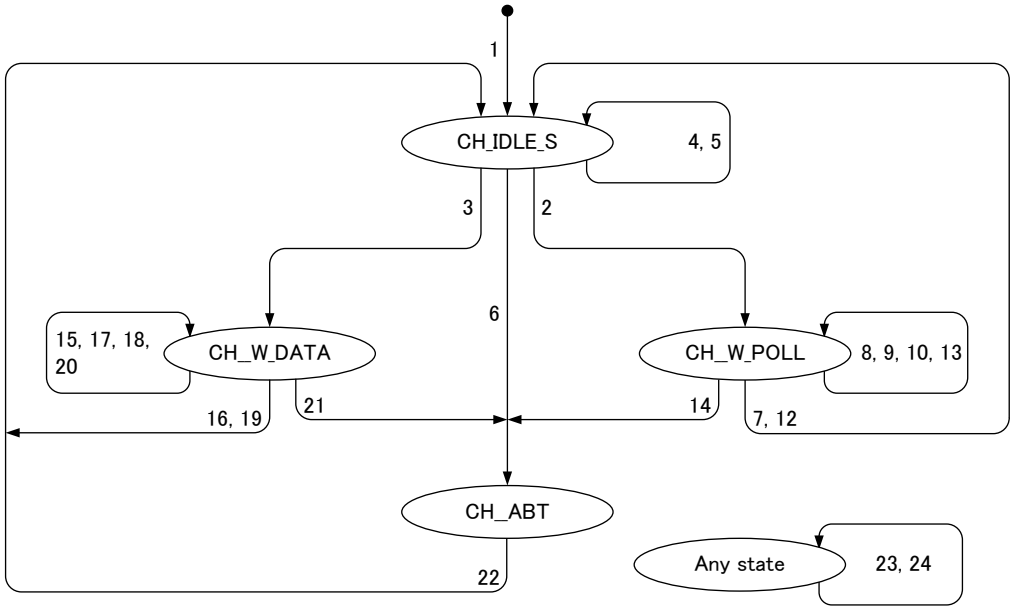


Figure 16 – The state diagram of message responder for basic format

Table 55 – The state table of message responder for basic format

No.	Current state	Event /condition =>action	Next state
1	Any states	Power on or CTC_Reset.req => V(Nms_n) := 0 V(Nmr_n) := 0	CH_IDLE_S
2	CH_IDLE_S	DL-SDA.req{SAP_ID,Node_ID,Length,SndDLSDU} => V(Nms_n) := 0 STORE_DLSDU(Ec1c2, Node_ID, SndDLSDU) V(Nmsg_len_n) := Length V(Nmsg_rem_len_n) := Length V(Fmsg_sending_n) := True	CH_W_POLL
3	CH_IDLE_S	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I && GET_MC_NS(RcvSRCSDU) = 0 && GET_MCTL(RcvSRCSDU,MCTL_P) = 0 => V(Nmr_n) := 1 V(Fmsg_sending_n) := False STORE_PDU_BMSG(Ec1c2, RcvSRCSDU) V(Nmr_n) := 1 V(Nmf_n) = 1 SndSRCSDU := BUILD_PDU_BMSG(MF_S_RR,0,V(Nmr_n),V(Nmf_n))	CH_W_DATA
4	CH_IDLE_S	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I && GET_MC_NS(RcvSRCSDU) = 0 && GET_MCTL(RcvSRCSDU,MCTL_P) = 1 => V(Nmr_n) := 1	CH_IDLE_S

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		V(Nmf_n) = 1 SndSRCSDU := BUILD_PDU_BMSG(MF_S_RNR,0,V(Nmr_n),V(Nmf_n)) STORE_PDU_MSG(Ec1c2, RcvSRCSDU) Node_ID := GET_SA(RcvSRCSDU) DLSDU := GET_DLSDU(Ec1c2) Length := GET_DLSDU_LEN(DLSDU) DL-SDA.ind{ SAP_ID, Rcv_sts , Node_ID, Length, DLSDU}	
5	CH_IDLE_S	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && CHK_MCTL_IS(RcvSRCSDU,MF_I) <> OK => V(Nmf_n) = 1 SndSRCSDU := BUILD_PDU_BMSG(MF_S_RNR,0,V(Nmr_n),V(Nmf_n))	CH_IDLE_S
6	CH_IDLE_S	DL-ABORT-SDA.req { SAP_ID } => V(Fmsg_sending_n) := False	CH_ABT
7	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RR && GET_MC_NR(RcvSRCSDU) = V(Nms_n) && V(Nmsg_rem_len_n) = 0 => V(Nmf_n) = 1 SndSRCSDU := BUILD_PDU_BMSG(MF_S_RNR,0,V(Nmr_n),V(Nmf_n)) DL-SDA.cnf{ SND_OK }	CH_IDLE_S
8	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RR && GET_MC_NR(RcvSRCSDU) = V(Nms_n) && V(Npkt_len_n) >= V(Nmsg_rem_len_n) => V(Nmf_n) = 1 SndSRCSDU := BUILD_PDU_BMSG(MF_I,V(Nms_n),V(Nmr_n),V(Nmf_n)) V(Nmsg_rem_len_n) := 0	CH_W_POLL
9	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RR && GET_MC_NR(RcvSRCSDU) = V(Nms_n) && V(Npkt_len_n) < V(Nmsg_rem_len_n) => V(Nmf_n) = 0 SndSRCSDU := BUILD_PDU_BMSG(MF_I,V(Nms_n),V(Nmr_n),V(Nmf_n)) V(Nms_n) = V(Nms_n) + 1 V(Nmsg_rem_len_n) := V(Nmsg_rem_len_n) - V(Npkt_len_n)	CH_W_POLL
10	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RR && GET_MC_NR(RcvSRCSDU) <> V(Nms_n) => SndSRCSDU := BUILD_PDU_BMSG(MF_I,V(Nms_n)-1,V(Nmr_n),V(Nmf_n))	CH_W_POLL
11	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RNR	CH_W_POLL

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		=> SndSRCSDU := BUILD_PDU_BMSG(MF_I,V(Nms_n)-1,V(Nmr_n),V(Nmf_n))	
12	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_REJ => V(Nms_n) := 0, V(Nmr_n) := 0, V(Nmf_n) = 1 DL-SDA.cnf{ SND_ABT } SndSRCSDU := BUILD_PDU_BMSG(MF_S_RNR,0,V(Nmr_n),V(Nmf_n))	CH_IDLE_S
13	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I => V(Nmf) = 1 SndSRCSDU := BUILD_PDU_BMSG(MF_S_RR,0,V(Nmr_n),V(Nmf_n))	CH_W_POLL
14	CH_W_POLL	DL-ABORT-SDA.req { SAP_ID } => (none)	CH_ABT
15	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I && GET_MC_NS(RcvSRCSDU) = V(Nmr_n) && GET_MCTL(RcvSRCSDU,MCTL_P) = 0 => STORE_PDU_BMSG(Ec1c2, RcvSRCSDU) V(Nmr_n) := V(Nmr_n) + 1 V(Nmf_n) = 1 SndSRCSDU := BUILD_PDU_MSG(MF_S_RR,0,V(Nmr_n),V(Nmf_n))	CH_W_DATA
16	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I && GET_MC_NS(RcvSRCSDU) = V(Nmr_n) && GET_MCTL(RcvSRCSDU,MCTL_P) = 1 => V(Nmr_n) := V(Nmr_n) + 1 V(Nmf_n) = 1 SndSRCSDU := BUILD_PDU_MSG(MF_S_RNR,0,V(Nmr_n),V(Nmf_n)) STORE_PDU_MSG(Ec1c2, RcvSRCSDU) Node_ID := GET_SA(RcvSRCSDU) DLSDU := GET_DLSDU(Ec1c2) Length := GET_DLSDU_LEN(DLSDU) DL-SDA.ind{ SAP_ID, Rcv_sts , Node_ID, Length, DLSDU}	CH_IDLE_S
17	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I && GET_MC_NS(RcvSRCSDU) <> V(Nmr_n) =>V(Nmf_n) = 1 SndSRCSDU := BUILD_PDU_MSG(MF_S_RR,0,V(Nmr_n),V(Nmf_n))	CH_W_DATA
18	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RR => V(Nmf_n) = 1 SndSRCSDU :=	CH_W_DATA

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		BUILD_PDU_MSG(MF_S_RR,0,V(Nmr_n),V(Nmf_n))	
19	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_REJ => V(Nms_n) := 0, V(Nmr_n) := 0, V(Nmf_n) = 1 SndSRCSDU := BUILD_PDU_MSG(MF_S_RNR),0,V(Nmr_n),V(Nmf_n))	CH_IDLE_S
20	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RNR => V(Nms_n) := 0, V(Nmr_n) := 0, V(Nmf_n) = 1 SndSRCSDU := BUILD_PDU_MSG(MF_S_RR),0,V(Nmr_n),V(Nmf_n))	CH_IDLE_S
21	CH_W_DATA	DL-ABORT-SDA.req { SAP_ID } => (none)	CH_ABT

22	CH_ABT	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU)
		/ Rcv_sts = OK
		=> V(Nms_n) := 0, V(Nmr_n) := 0, V(Nmf_n) = 1
		DL-ABORT-SDA.cnf
		if (V(Fmsg_sending_n) = True) then
		DL-SDA.cnf{SND_ABT}
		endif
		SndSRCSDU := BUILD_PDU_MSG(MF_S_REJ,0,0,V(Nmf_n))
			CH_IDLE_S

23	Any state	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) <> FT_MSG => (none)	Same state
24	Any state	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts <> OK => (none)	Same state

6.2.3.3.3 Segmentation for short format DLPDU

6.2.3.3.3.1 Initiator for short format DLPDU

Figure 17 and Table 56 show the state diagram and the state table of an initiator of the message segmentation machine when DLE adopts short format DLPDU.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

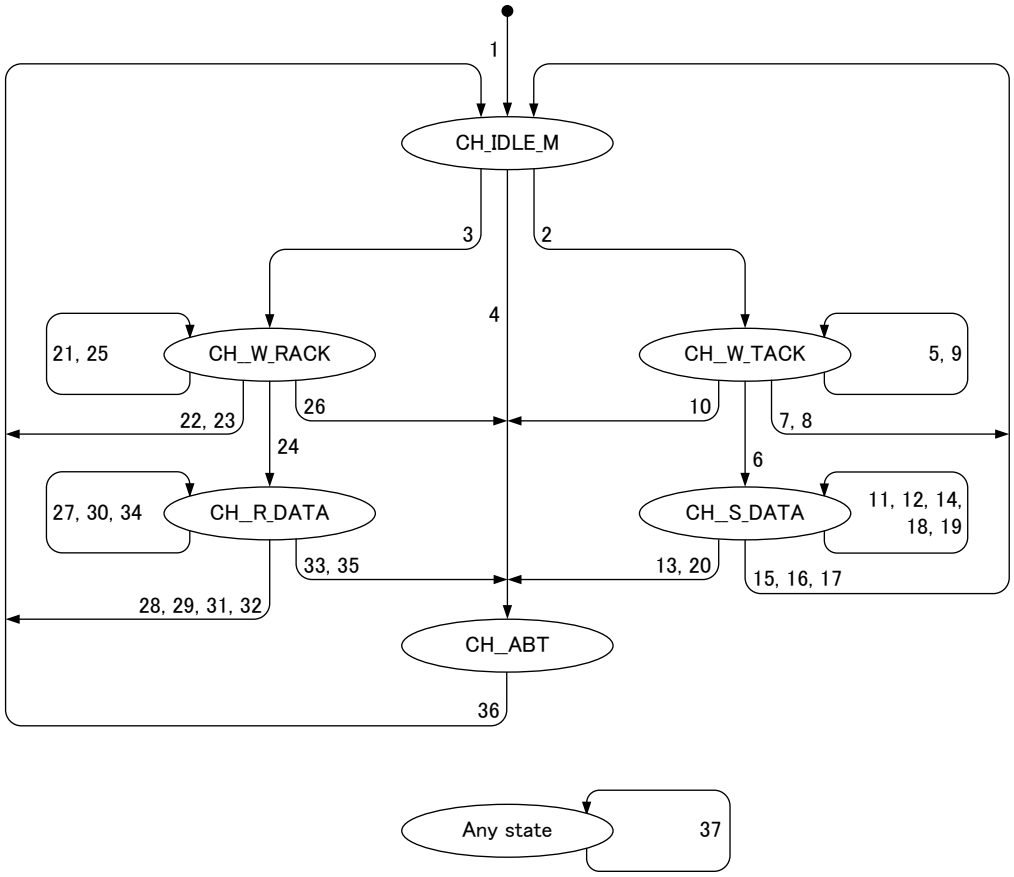


Figure 17 – The state diagram of message initiator for short format

Table 56 – The state table of message initiator for short format

No.	Current state	Event /condition =>action	Next state
1	Any states	Power on or CTC_Reset.req => (none)	CH_IDLE_M
2	CH_IDLE_M	DL-SDA.req{SAP_ID,Node_ID,Length,SndDLSDU} => V(Nmno_n) = GET_SMSG_NUM(Length) STORE_DLSDU(Ec1c2, Node_ID, SndDLSDU) V(Fmsg_sending_n) := True	CH_W_TACK
3	CH_IDLE_M	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) => V(Nmsd_n) := 0, V(Nmend_n) := 1, V(Nms_n) := 0 SndSRCSDU := BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n)) V(Fmsg_sending_n) := False	CH_W_RACK
4	CH_IDLE_M	DL-ABORT-SDA.req{SAP_ID,Node_ID} => V(Fmsg_sending_n) := False	CH_ABT
5	CH_W_TACK	EXEC_MSPM_IS(Fc1c2, SndSRCSDU) => V(Nmsd_n) := 0, V(Nmend_n) := 0, V(Nms_n) := 0 SndSRCSDU := BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_W_TACK
6	CH_W_TACK	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 0 => (none)	CH_S_DATA

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
7	CH_W_TACK	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = ABT_NUM => DL-SDA.cnf{ SND_ABT }	CH_IDLE_M
8	CH_W_TACK	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) <> ABT_NUM => DL-SDA.cnf{ SND_ERR }	CH_IDLE_M
9	CH_W_TACK	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1) => (none)	CH_W_TACK
10	CH_W_TACK	DL-ABORT-SDA.req{SAP_ID,Node_ID} => (none)	CH_ABT
11	CH_S_DATA	EXEC_MSPM_IS(Fc1c2, SndSRCSDU) / (V(Nmno_n)) > 1 => V(Nmsd_n) := 1, V(Nmend_n) := 0 SndSRCSDU := BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_S_DATA
12	CH_S_DATA	EXEC_MSPM_IS(Fc1c2, SndSRCSDU) / (V(Nmno_n)) = 1 => V(Nmsd_n) := 1, V(Nmend_n) := 1 SndSRCSDU := BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_S_DATA
13	CH_S_DATA	EXEC_MSPM_IS(Fc1c2, SndSRCSDU) / (V(Nmno)) < 1 => (none)	CH_ABT
14	CH_S_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 0 => (none)	CH_S_DATA
15	CH_S_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = ABT_NUM => DL-SDA.cnf{ SND_ABT }	CH_IDLE_M
16	CH_S_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) <> ABT_NUM => DL-SDA.cnf{ SND_ERR }	CH_IDLE_M
17	CH_S_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) = V(Nms_n) && V(Nmno_n) = 1	CH_IDLE_M

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		=> DL-SDA.cnf{ SND_OK }	
18	CH_S_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) = V(Nms_n) => V(Nms_n) := V(Nms_n) + 1 V(Nmno_n) := V(Nmno_n) – 1	CH_S_DATA
19	CH_S_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) <> V(Nms_n) => (none)	CH_S_DATA
20	CH_S_DATA	DL-ABORT-SDA.req{SAP_ID,Node_ID} => (none)	CH_ABT
21	CH_W_RACK	EXEC_MSPM_IS(Fc1c2, SndSRCSDU) => V(Nmsd_n) := 0, V(Nmend_n) := 1, V(Nms_n) := 0 SndSRCSDU := BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_W_RACK
22	CH_W_RACK	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 0 => (none)	CH_IDLE_M
23	CH_W_RACK	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) = ABT_NUM => (none)	CH_IDLE_M
24	CH_W_RACK	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) <> ABT_NUM => (none)	CH_R_DATA
25	CH_W_RACK	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1) => (none)	CH_W_RACK
26	CH_W_RACK	DL-ABORT-SDA.req{SAP_ID,Node_ID} => (none)	CH_ABT
27	CH_R_DATA	EXEC_MSPM_IS(Fc1c2, SndSRCSDU) => V(Nmsd_n) := 0, V(Nmend_n) := 1 SndSRCSDU := BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_R_DATA
28	CH_R_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 0 => (none)	CH_IDLE_M
29	CH_R_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK	CH_IDLE_M

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		&& CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) = ABT_NUM => (none)	
30	CH_R_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) <> ABT_NUM => (none)	CH_R_DATA
31	CH_R_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) = V(Nms_n) && (CHK_END(RcvSRCSDU) = 1) => Node_ID := RcvSRCSDU.SA STORE_PDU_MSG(Ec1c2, RcvSRCSDU) DLSDU := GET_DLSDU(Ec1c2)) Length := GET_DLSDU_LEN(DLSDU) DL-SDA.ind{ SAP_ID, Rcv_sts , Node_ID, Length, DLSDU}	CH_IDLE_M
32	CH_R_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) = V(Nms_n) && (CHK_END(RcvSRCSDU) = 0) && CHK_MRBUF() = OK => V(Nms_n) := V(Nms_n) + 1 STORE_PDU_MSG(Ec1c2, RcvSRCSDU)	CH_IDLE_M
33	CH_R_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) = V(Nms_n) && (CHK_END(RcvSRCSDU) = 0) && CHK_MRBUF() <> OK => (none)	CH_ABT
34	CH_R_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) <> V(Nms_n)) => (none)	CH_R_DATA
35	CH_R_DATA	DL-ABORT-SDA.req{SAP_ID,Node_ID} => (none)	CH_ABT

36	CH_ABT	EXEC_MSPM_IS(Fc1c2, SndSRCSDU)
		
		=> V(Nmsd_n) := 0, V(Nmend_n) := 1, V(Nms_n) := ABT_NUM
		SndSRCSDU :=
		BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))
		DL-ABORT-SDA.cnf
		
		if (V(Fmsg_sending_n) = True) then
		DL-SDA.cnf{SND_ABT}
		endif
			CH_IDLE_M

37	Any state	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts <> OK => (none)	Same state

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

6.2.3.3.3.2 Responder for short format DLPDU

Figure 18 and Table 57 show the state diagram and the state table of a responder of the message segmentation machine when DLE adopts short format DLPDU.

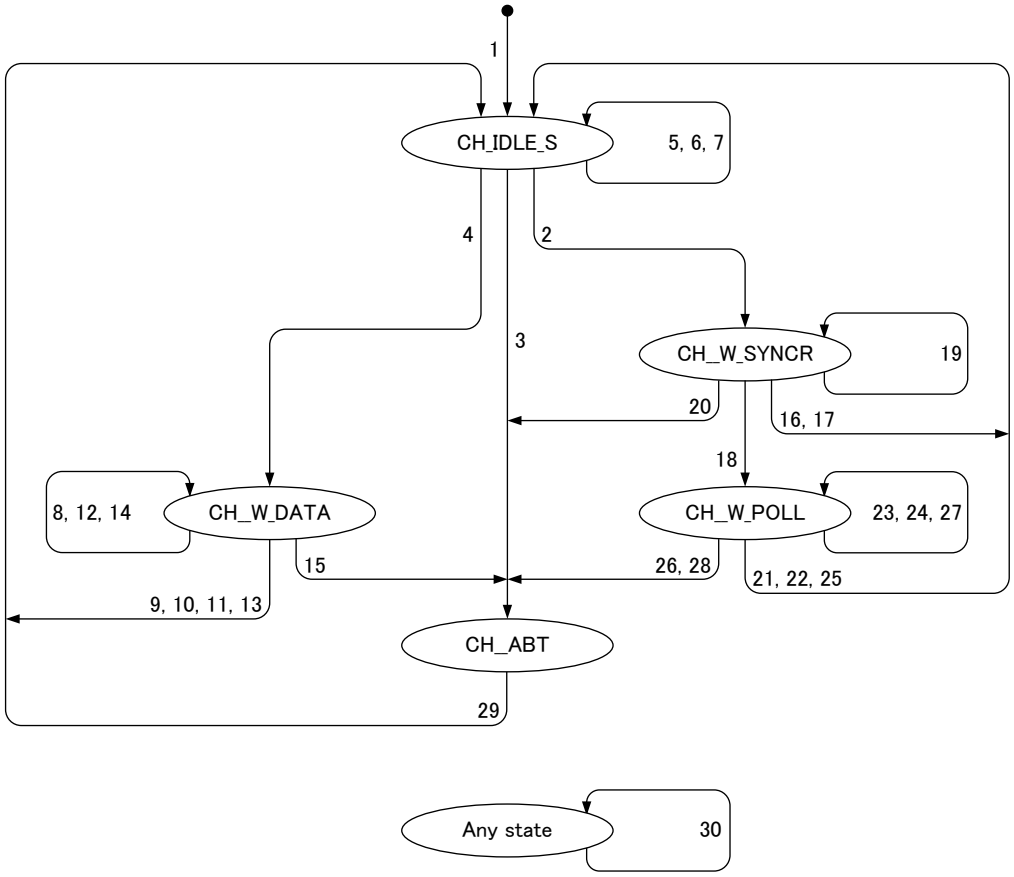


Figure 18 – The state diagram of message responder for short format

Table 57 – The state table of message responder for short format

No.	Current state	Event /condition =>action	Next state
1	Any states	Power on or CTC_Reset.req => (none)	CH_IDLE_S
2	CH_IDLE_S	DL-SDA.req{SAP_ID,Node_ID,Length,SndDLSDU} => V(Nmno) = GET_SMSG_NUM(Length) STORE_DLSDU(Ec1c2, Node_ID, SndDLSDU) V(Fmsg_sending_n) := True	CH_W_SYNCR
3	CH_IDLE_S	DL-ABORT-SDA.req{SAP_ID,Node_ID} => V(Fmsg_sending_n) := False	CH_ABT
4	CH_IDLE_S	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 0 => V(Nmsd_n) := 0, V(Nmend_n) := 0, V(Nms_n) := 0 SndSRCSDU := BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n)) V(Fmsg_sending_n) := False	CH_W_DATA
5	CH_IDLE_S	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU)	CH_IDLE_S

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		/ Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = ABT_NUM => V(Nmsd_n) := 0, V(Nmend_n) := 1, V(Nms_n) := ABT_NUM SndSRCSDU := BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	
6	CH_IDLE_S	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) <> ABT_NUM => V(Nmsd_n) := 0, V(Nmend_n) := 0, V(Nms_n) := 0 SndSRCSDU := BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_IDLE_S
7	CH_IDLE_S	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1 => V(Nmsd_n) := 0, V(Nmend_n) := 0, V(Nms_n) := 0 SndSRCSDU := BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_IDLE_S
8	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 0 => SndSRCSDU := BUILD_PDU_LAST_SMSG()	CH_W_DATA
9	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = ABT_NUM => V(Nmsd_n) := 0, V(Nmend_n) := 1, V(Nms_n) := ABT_NUM SndSRCSDU := BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_IDLE_S
10	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) <> ABT_NUM => SndSRCSDU := BUILD_PDU_LAST_SMSG()	CH_IDLE_S
11	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = V(Nms_n) && CHK_END(RcvSRCSDU) = 1 => V(Nmsd_n) := CHK_SD(RcvSRCSDU) V(Nmend_n) := CHK_END(RcvSRCSDU) V(Nms_n) := CHK_S(RcvSRCSDU) SndSRCSDU := BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n)) STORE_PDU_MSG(Ec1c2, RcvSRCSDU) Node_ID := RcvSRCSDU.SA DLSDU := GET_DLSDU(Ec1c2)) Length := GET_DLSDU_LEN(DLSDU) DL-SDA.ind{ SAP_ID, Rcv_sts , Node_ID, Length, DLSDU}	CH_IDLE_S
12	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = V(Nms_n)	CH_W_DATA

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		&& CHK_END(RcvSRCSDU) = 0 && CHK_MRBUF() = OK => V(Nmsd_n) := CHK_SD(RcvSRCSDU) V(Nmend_n) := CHK_END(RcvSRCSDU) V(Nms_n) := CHK_S(RcvSRCSDU) SndSRCSDU := BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n)) V(Nms_n) := V(Nms_n) + 1 STORE_PDU_MSG(Ec1c2, RcvSRCSDU)	
13	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = V(Nms_n) && CHK_END(RcvSRCSDU) = 0 && CHK_MRBUF() <> OK => V(Nmsd_n) := 0, V(Nmend_n) := 1, V(Nms_n) := ABT_NUM SndSRCSDU := BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_IDLE_S
14	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) <> V(Nms_n) => SndSRCSDU := BUILD_PDU_LAST_SMSG()	CH_W_DATA
15	CH_W_DATA	DL-ABORT-SDA.req{SAP_ID,Node_ID} => (none)	CH_ABT
16	CH_W_SYNCR	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 0 => SndSRCSDU := BUILD_PDU_LAST_SMSG() DL-SDA.cnf{ SND_ERR }	CH_IDLE_S
17	CH_W_SYNCR	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = ABT_NUM => V(Nmsd_n) := 0, V(Nmend_n) := 1, V(Nms_n) := ABT_NUM SndSRCSDU := BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n)) DL-SDA.cnf{ SND_ABT }	CH_IDLE_S
18	CH_W_SYNCR	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) <> ABT_NUM => V(Nmsd_n) := 0, V(Nmend_n) := 1, V(Nms_n) := 0 SndSRCSDU := BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_W_POLL
19	CH_W_SYNCR	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1 => V(Nmsd_n) := 0, V(Nmend_n) := 1, V(Nms_n) := 0 SndSRCSDU := BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_W_SYNCR
20	CH_W_SYNCR	DL-ABORT-SDA.req{SAP_ID,Node_ID} => (none)	CH_ABT
21	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU)	CH_IDLE_S

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		/ Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 0 => SndSRCSDU := BUILD_PDU_LAST_SMSG() DL-SDA.cnf{ SND_ERR }	
22	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && (CHK_S(RcvSRCSDU) = ABT_NUM => V(Nmsd_n) := 0, V(Nmend_n) := 1, V(Nms_n) := ABT_NUM SndSRCSDU := BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n)) DL-SDA.cnf{ SND_ABT }	CH_IDLE_S
23	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && (CHK_S(RcvSRCSDU) <> ABT_NUM => SndSRCSDU := BUILD_PDU_LAST_SMSG()	CH_W_POLL
24	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = V(Nms_n) && V(Nmno_n) > 1 => V(Nmsd_n) := CHK_SD(RcvSRCSDU) V(Nmend_n) := CHK_END(RcvSRCSDU) V(Nms_n) := CHK_S(RcvSRCSDU) BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n)) V(Nms_n) := V(Nms_n) + 1 V(Nmno_n) := V(Nmno_n) – 1	CH_W_POLL
25	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = V(Nms_n) && V(Nmno_n) = 1 => V(Nmsd_n) := CHK_SD(RcvSRCSDU) V(Nmend_n) := 0 V(Nms_n) := CHK_S(RcvSRCSDU) BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n)) DL-SDA.cnf{ SND_OK }	CH_IDLE_S
26	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = V(Nms_n) && V(Nmno_n) < 1 => V(Nmsd_n) := 0, V(Nmend_n) := 1, V(Nms_n) := ABT_NUM SndSRCSDU := BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n)) DL-SDA.cnf{ SND_ERR }	CH_ABT
27	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) <> V(Nms_n) => SndSRCSDU := BUILD_PDU_LAST_SMSG()	CH_W_POLL
28	CH_W_POLL	DL-ABORT-SDA.req{SAP_ID,Node_ID} => (none)	CH_ABT
29	CH_ABT	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU)	CH_IDLE_S

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state

		=> V(Nmsd_n) := 0, V(Nmend_n) := 1, V(Nms_n) := ABT_NUM
		SndSRCSDU := BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))
		DL-ABORT-SDA.cnf
		if (V(Fmsg_sending_n) = True) then
		DL-SDA.cnf{SND_ABT}
		endif	

30	Any state	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts <> OK => (none)	Same state

6.2.3.3.4 Functions used by message segmentation protocol machine

Table 58 shows the list of functions which is used by the message segmentation machine.

Table 58 – List of functions used by the message segmentation machine

Function name	Parameter		Return Value	Operation
	Input	Output		
STORE_DLSDU	Ec1c2, DLS-user_message	none	none	This function stores the DLS-user data of the message into the internal buffer.
GET_DLSDU	Ec1c2	none	DLS-user_message	This function retrieves the DLS-user message received from the remote station.
BUILD_PDU_BMSG	mframe_type, Ns, Nr, flag	none	SRCSDU	This function builds the SRCSDU to be sent. The input parameters are the value that to be contained the MCTL field of the message frame.
STORE_PDU_BMSG	Ec1c2, SRCSDU	none	none	This function stores the received SRCSDU into the internal buffer to assemble DLSPDU.
GET_TYP	SRCSDU	none	frame_type	This function takes out the value of the type field of the specified SRCSDU.
CHK_MC_TYP	SRCSDU	none	mframe_type	This function takes out the message type from the MCTL field of the specified SRCSDU.
GET_MCTL_NR	SRCSDU	none	Nr	This function takes out the value of Nr from the MCTL field of the specified SRCSDU.
GET_MCTL_NS	SRCSDU	none	Ns	This function takes out the value of Ns from the MCTL field of the specified SRCSDU.
GET_MCTL_F	SRCSDU	none	Flag	This function takes out the value of P/F from the MCTL field of the specified SRCSDU.
GET_SMSG_NUM		msg_packet_num		This function calculates

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Function name	Parameter		Return Value	Operation
	Input	Output		
				the number of message segments.
CHK_SD		Result		This function returns the value of S/D flag out of the received message frame.
CHK_END		Result		This function returns the value of END flag out of the received message frame.
CHK_S		s_number		This function returns the value of S(n) field out of the received message frame.
CHK_MRBUF		Result		This function checks whether there is a free space in the receive buffer.
BUILD_PDU_SMSG	SD_flag, END_flag, S_number	SRCSDU		This function builds message frame to be sent from specified data.
BUILD_PDU_LAST_SMSG		SRCSDU		This function builds message frame that is sent at previous cycle.

6.2.3.4 Acyclic transmission protocol machine

6.2.3.4.1 Overview

This subclause describes the protocol machine that works in the acyclic transmission mode. When entering the operational state, CTC starts this protocol machine by the request by DLM instead of the cyclic transmission protocol machine. There is only one kind of protocol machine regardless of the station type of the C1 master, the C2 master, and the slave.

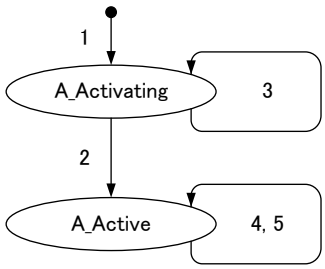


Figure 19 – The state diagram of acyclic transmission protocol machine

Table 59 – The state table of acyclic transmission protocol machine

No.	Current state	Event /condition =>action	Next state
1	Any state	Power on or CTC_Reset.req => (none)	A_Activating
2	A_Activating	CTC_Start.req { } / ((V(Cyc_Sel) = CMode_Ayclic)) && (V(PDUType) = PDUBasic)) => CTC_Start.cnf { OK }	A_Active

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
3	A_Activating	CTC_Start.req / ((V(Cyc_Sel) <> CMode_Ayclic)) || (V(PDUType) <> PDUBasic)) => CTC_Start.cnf { NG }	A_Activating
4	A_Active	DL-SDN.req {SAP_ID, Node_ID, Length, DLSDU } / (V(Cyc_Sel) = CMode_Ayclic)) => SRCSDU := BUILD_PDU_ACYC(Node_ID, Length, DLSDU) Len := GET_LEN(SRCSDU) SRC_Send_Frame.req { Node_ID, Len, SRCSDU } DL-SDN.cnf { OK }	A_Active
5	A_Active	SRC_Recv_Frame.ind {Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) =>Node_ID := GET_DA(RcvSRCSDU) Len := GET_LEN(SRCSDU) DLSDU := GET_DATA(SRCSDU) DL-SDN.ind {SAP_ID, Node_ID, Len, DLSDU }	A_Active

6.2.3.4.2 Functions used by acyclic transmission protocol machine

All the functions used by acyclic transmission protocol machine are summarized in Table 60.

Table 60 – The list of functions used acyclic transmission protocol machine

Function name	Input	Output	Operation
GET_DA	SRCSDU	Node_ID	This function takes out the value of the destinaion address field of the specified SRCSDU.
GET_LEN	SRCSDU	Length	This function takes out the value of the length field of the specified SRCSDU.
BUILD_PDU_ASYNC	Node_ID, Length, DLSDU		This function builds SRCSDU to request SRC from DLS-user data.

6.2.4 CTC-DLM interface

6.2.4.1 Overview

This subclause describes the interface between CTC and DL-management.

The CTC provides the following services for DL-management.

- Reset service (CTC_Reset)
- Start transmission (CTC_Start)
- Error event service (CTC_Err_Event)

6.2.4.2 Detailed definition of the primitives and parameters

Table 61 indicates the primitives and parameters exchanged between CTC and DLM. Details of these parameters will be shown in the following sections.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Table 61 – Primitives and parameters exchanged between CTC and DLM

Primitive	Source	Parameter	Function
CTC_Reset.req	DLM		Reset CTC
CTC_Reset.cnf	CTC	Result	
CTC_Start.req	DLM		Start cyclic or acyclic transmission
CTC_Start.cnf	CTC	Result	
CTC_Err_Event.ind	CTC	ErrEvent_ID	Error event

6.2.4.2.1 CTC_Reset

None

6.2.4.2.2 CTC_Start

None

6.2.4.2.3 CTC_Err_Event

Table 62 indicates the primitives and parameters of event service.

Table 62 – Error event primitive and parameters

CTC_Err_Even	Indication
Parameter name	Output
Err_Event_ID	M

6.3 Send Receive Control

6.3.1 General

SRC provides the following functions.

- Send frame
- Receive frame
- Repeater or loop back of received frame

6.3.1.1 Send frame

SRC executes the transmission of one frame by one request from CTC and DLM. When they request to send a frame, SRC serializes the frame to be sent and outputs to PHY. When two PHY interfaces or more are implemented in DLE, SRC outputs it to all ports.

6.3.1.2 Receive frame

When SRC receives the data from PHY, SRC assembles the frame from received data and notifies CTC or DLM of the receipt..

6.3.1.3 Repeater or loop back of received frame

Repeater and loop back are functions that operate exclusively. Repeater is a default active function. DLM issues the request for SRC to toggle these functions. When DLM activates loop back function, SRC repeater function becomes inactive.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

When two PHY interfaces or more are implemented in the DLE and repeater function is active, SRC repeats the received data from one PHY interface to others. When loop back function is active, SRC return the received data to the PHY interface that input it.

6.3.2 SRC-CTC interface

6.3.2.1 Overview

SRC provides the following services to CTC.

- Send frame service (SRC_Send_Frame)
- Receive frame service (SRC_Recv_Frame)

6.3.2.2 Detailed definitions of the service primitives and parameters

Table 63 indicates the primitives and parameters which exchanged between SRC and CTC. Details of these parameters will be shown in the following sections.

Table 63 – primitives and parameters for SRC-CTC interface

Primitives	Source	Parameters
SRC_Send_Frame.req	CTC	Length, SRCSDU
SRC_Send_Frame.cnf	SRC	Result
SRC_Recv_Frame.ind	SRC	Rcv_Sts, Length, SRCSDU

6.3.2.2.1 SRC_Send_Frame

Table 64 indicates the primitives and parameters of send frame service.

Table 64 – Send frame primitive and parameters

SRC_Send_Frame	Request	Confirm
Parameter name	Input	Output
Length	M	
SRCSDU	M	
Result		M

6.3.2.2.2 SRC_Recv_Frame

Table 65 indicates the primitives and parameters of receive frame service.

Table 65 – Receive frame primitives and parameters

SRC_Recv_Frame	indication
Parameter name	Output
Rcv_Sts	M
Length	M
SRCSDU	M

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

6.3.3 Detailed specification of SRC

6.3.3.1 Overview

There are two kinds of SRC according to the number of PHY interfaces that the SRC entity mounts. Figure 20 shows the internal architecture of the SRC entity when it has one PHY interface, and Figure 21 shows that when it has two or more. When two PHY interfaces or more are mounted, repeater function shall be mounted in SRC entity.

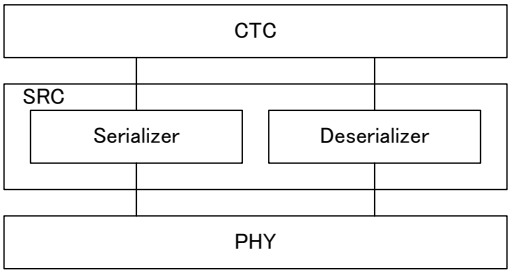


Figure 20 – Internal architecture of one-port SRC

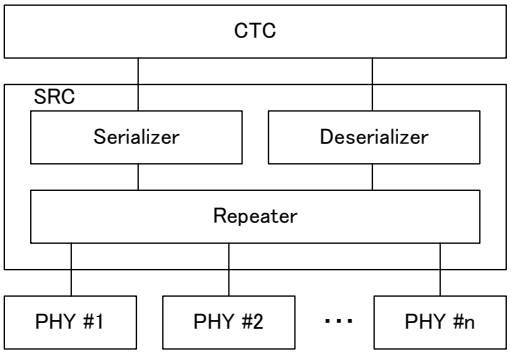


Figure 21 – Internal architecture of multi-port SRC

6.3.3.2 Serializer

6.3.3.2.1 Internal architecture

Figure 22 shows internal structure of the serializer. There are five function blocks in the serializer.

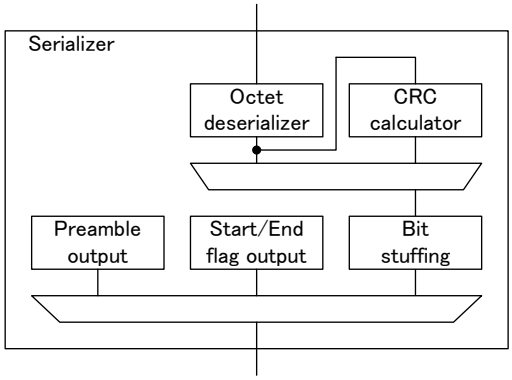


Figure 22 – Internal architecture of serializer

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

6.3.3.2.2 Behaviour

6.3.3.2.2.1 Overview

When the SRC received SRC_Send_Frame request primitive to send a frame, the serializer is activated and transmit it. After the serializer finishes transmitting, the SRC sends SRC_Send_Frame.cnf to notify the requester of the result.

The serializer uses the following procedure to transmit a frame:

1) The preamble output block issues Ph_data request
2) The start/end flag output block issues Ph_data request to output start flag
3) Enable the bit stuffing block and CRC calculator block
4) The octet serializer serializes all octets of SRCSDU.
5) The CRC calculator issues Ph_data request
6) Disable the bit stuffing block and CRC calculator block
7) The start/end flag output block issues Ph_data request to output end flag.

Where, the serializer shall set Ph_frame request to be True when it issues Ph_data request.

The following subclause describes the requirements of behaviour to each function blocks which are included in the serializer. The requirements are different depends on DLPDU format.

6.3.3.2.2.2 Serializer for basic format frame

a) Preamble output

This function block outputs preamble pattern (see 5.2.1.1).

b) Start flag and End flag output

This function block outputs the sequence of bit pattern (see 5.2.1.2). The function outputs nothing when the end flag output is requested.

c) Octet serializer

This function block serializes SRCSDU from LSB to MSB by one octet, and outputs the processed data to the bit stuffing block and CRC calculator block. It notifies CRC calculator of the end of SRCSDU when the serialization for the length specified by the parameter of request is finished. During the processing, the serializer detects that the length of input SRCSDU is not equal to that specified by the parameter of request, it aborts processing and notifies CRC calculator of this failure.

d) CRC calculator

This function block has two states, enable and disable. In the enable state, it executes CRC calculation (see 5.2.1.8) from input bit stream. It outputs the calculation result as a bit stream from MSB to LSB when notified of the end of SRCSDU by the octet serializer. In the disable state, it doesn't work.

e) Bit stuffing

This function block outputs the input bit stream as it is.

6.3.3.2.2.3 Serializer for short format frame

a) Preamble output

This function block outputs preamble pattern (see 5.3.1.1).

b) Start flag and End flag output

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

This function block outputs the sequence of bit pattern (see 5.3.1.2). The function outputs the same bit pattern when the end flag output is requested (see 5.3.1.7).

c) Octet serializer

See 6.3.3.2.2.2.

d) CRC calculator

This function block has two states, enable and disable. In the enable state, it executes CRC calculation (see 5.3.1.6) from input bit stream. It outputs the calculation result as a bit stream from MSB to LSB when notified of the end of SRCSDU by the octet serializer. In the disable state, it doesn't work.

e) Bit stuffing

This function block has two states, enable and disable. In the enable state, its behaviour is identical to that of HDLC serializer. When it detects 5 consecutive "1"s from the input bit stream, it inserts a "0" immediately after them. Otherwise it outputs the input bit stream as it is. In the disable state, it doesn't work.

6.3.3.3 Deserializer

6.3.3.3.1 Internal architecture

Figure 23 shows internal structure of the deserializer. There are two functions in the deserializer.

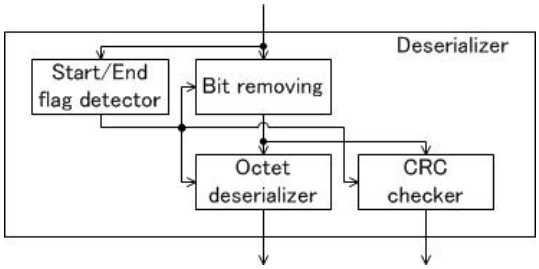


Figure 23 – Internal architecture of deserializer

6.3.3.3.2 Behaviour

6.3.3.3.2.1 Overview

When the SRC received Ph_lock indication primitive from the PHY entity, the deserializer is activated and receiving the input stream. After the deserializer finishes receiving, the SRC issues SRC_Recv_Frame.ind to pass the SRCPDU and the receive status. The receive status is OK only when both CRC check is OK and Ph_status is Normal, otherwise NG.

The deserializer uses the following procedure to receive data:

1) The start/end start flag detector begins to accept Ph_data indication as the input stream when it receives Ph_lock indication.
2) The start/end flag detector enables all other blocks in the deserializer, when it detects the start flag from the input stream.
3) The octet deserializer reassembles a SRCSDU from the input stream while Ph_frame indication is True.
4) The start/end flag detector notifies all other blocks in the deserializer of the end of stream, when it detects the end flag from the input stream or Ph_frame indication become False.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

5) The CRC checker checks CRC field and notifies of the result when it is notified of the end of stream.

The destination of SRC_Recv_Frame.ind is CTC only while the state of DLM is idle. Otherwise, the destination is DLM.

6.3.3.3.2.2 Deserializer for basic format frame

a) Start flag and End flag detector

This function block begins to accept Ph_data indication as the input stream when it receives Ph_lock indication. It enables all other blocks in the deserializer, when it detects the SFD (see 5.2.1.2) from the input stream. It notifies all other blocks in the deserializer of the end of steam, when Ph_frame indication become False.

b) Bit removing

This function block outputs the input bit stream as it is.

c) Octet deserializer

This function block has two states, enable and disable. In the enable state, it assembles the octet row from the input bit stream to build a SRCSDU. At this time, each octet is assembled in order of LSB to MSB. When it is notified of the end of stream by the start/end flag detector, it changes the state into disable. In the disable state, it doesn't work.

d) CRC checker

This function block has two states, enable and disable. In the enable state, it execute CRC calculation (see 5.2.1.8) from input bit stream. When it is notified of the end of stream by the start/end flag detector, it compares the calculated value and the last 32-bits data of the input stream, and notifies of the result.

6.3.3.3.2.3 Deserializer for short format frame

a) Start flag and End flag detector

This function block begins to accept Ph_data indication as the input stream when it receives Ph_lock indication. It enables all other blocks in the deserializer, when it detects the start flag (see 5.3.1.2) from the input stream. It notifies all other blocks in the deserializer of the end of steam when it detects the end flag from the input stream. Or, it notifies of the abort when Ph_frame indication is false before the end flag.

b) Bit removing

This function block has two states, enable and disable. In the enable state, its behaviour is identical to that of HDLC deserializer. When this function block detects the pattern of "0" following 5 consecutive "1" from the input bit stream, it removes "0" from the output. When it is notified of the end of stream by the start/end flag detector, it changes the state into disable.

c) Octet deserializer

See 6.3.3.3.2.2 c).

d) CRC checker

This function block has two states, enable and disable. In the enable state, it executes CRC calculation (see 5.3.1.6) from input bit stream. When it is notified of the end of stream by the start/end flag detector, it compares the calculated value and the last 32-bits data of the input stream, and notifies of the result. When it is notified of the abort by the start/end flag detector, it notifies of failure.

This function block has two states, enable and disable. In the enable state, it executes CRC calculation (see 5.2.1.8) from input bit stream. It outputs the calculation result as a bit stream

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

from MSB to LSB when notified of the end of SRCSDU by the octet serializer. In the disable state, it doesn't work.

During the processing, the serializer detects that the length of input SRCSDU is not equal to that specified by the parameter of request, it aborts processing and notifies CRC calculator of this failure.

6.3.3.4 Repeater

The repeater has two states, normal state and loop back state. Normal state is default state. In normal state, the repeater transmit incoming stream from one PHY interface to all others.

The repeater changes its state into loop back and start repeat-watch timer when it receives SRC_Enable_Loopback request primitive from DL-management. In loop back state, the repeater returns the incoming stream to the PHY interface which inputted it, instead of transmitting to other PHY interfaces. The repeater changes its state into normal, when the number of execution which is specified by the parameter of the request primitive is finished or repeat-watch timer is expired. Then the repeater passes SRC_Enable_Loopback confirm primitive to the DL-management to indicate the success of the corresponding service request.

6.3.4 SRC-DLM interface

6.3.4.1 Overview

This subclause describes the interface between CTC and DL-management.

The CTC provides the following services for DL-management.

- Reset service (SRC_Reset)
- Send frame (SRC_Send_Frame)
- Recv frame (SRC_Recv_Frame)
- Enable loopback (SRC_Enable_Loopback)
- Error Event service (SRC_Err_Event)

6.3.4.2 Detailed definition of the primitive and parameters

Table 66 shows the primitives exchanged between SRC and DLM. The following subclauses describe the primitives and the parameters.

Table 66 – Primitives and parameters exchanged between SRC and DLM

Primitive	Source	parameters
SRC_Reset.req	DLM	<none>
SRC_Reset.cnf	SRC	Result
SRC_Send_Frame.req	DLM	Length, SRCSDU
SRC_Send_Frame.cnf	SRC	Result
SRC_Recv_Frame.ind	SRC	Rcv_Sts, Length, SRCSDU
SRC_Enable_Loopback.req	DLM	LB_Cnt
SRC_Enable_Loopback.cnf	SRC	Result
SRC_Err_Event.ind	SRC	Err_Event_ID

6.3.4.2.1 SRC_Reset

None

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

6.3.4.2.2 SRC_Send_Frame

See 6.3.2.2.1.

6.3.4.2.3 SRC_Recv_Frame

See 6.3.2.2.2.

6.3.4.2.4 SRC_Enable_Loopback

Table 67 indicates the primitives and parameters of event service.

Table 67 – Get value primitive and parameters

SRC_Enable_Loopback	Request	Confirm
Parameter name	Input	Output
LB_Cnt	M	
Result		M

6.3.4.2.5 SRC_Err_Event

Table 68 indicates the primitives and parameters of event service.

Table 68 – Error event primitive and parameters

SRC_Err_Event	Indication
Parameter name	Output
Err_Event_ID	M

7 DL-management layer (DLM)

7.1 Overview

The interface protocol between the DLM and the DLMS-user is described in this subclause. This subclause of the DL-management protocol provides the DL-management services specified in IEC 61158-3-24, Clause 5 by making use of the services available by the DLMS-user. DLM provide the services for DLMS-user to initialize stations.

In this subclause, fully implementation matters and local matters are intentionally excluded.

7.2 Primitive definitions

7.2.1 Primitives exchanged between DLMS-user and DLM

The primitives exchanged with the DLMS-user between DLM are shown below. Table 69 is for the case that issued by DLMS-user and Table 70 is for the case that issued by DLM.

Table 69 – The list of primitives and parameters (DLMS-user source)

Primitive	Source	Parameters
DLM-RESET.req	DLMS-user	
DLM-SET-VALUE.req	DLMS-user	Var_ID, Val
DLM-GET-VALUE.req	DLMS-user	Var_ID
DLM-MEAS-DELAY.req	DLMS-user	

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Primitive	Source	Parameters
DLM-SET-COMMOD.req	DLMS-user	
DLM-START.req	DLMS-user	
DLM-CLR-ERR.req	DLMS-user	Err_Status

Table 70 – The list of primitives and parameters (DLM source)

Primitive	Source	Parameters
DLM-RESET.cnf	DLM	Result
DLM-SET-VALUE.cnf	DLM	Result
DLM-GET-VALUE.cnf	DLM	Result, Val
DLM-MEAS-DELAY.cnf	DLM	Result
DLM-MEAS-DELAY.ind	DLM	Delay_time
DLM-SET-COMMOD.cnf	DLM	Result
DLM-START.cnf	DLM	Result
DLM-START.ind	DLM	Com_Mode, Cycle_time, C2_stime, Max_Delay, TM_unit
DLM-CLR-ERR.cnf	DLM	Result
DLM-EVENT.ind	DLM	Event_ID

7.2.2 Parameters used with DLM primitives

See IEC 61158-3-24, 5.3 for parameters of service primitive.

7.3 DLM protocol machine

7.3.1 C1 master

The DLM state transition diagrams for C1 master are shown in Figure 24 and Table 71.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

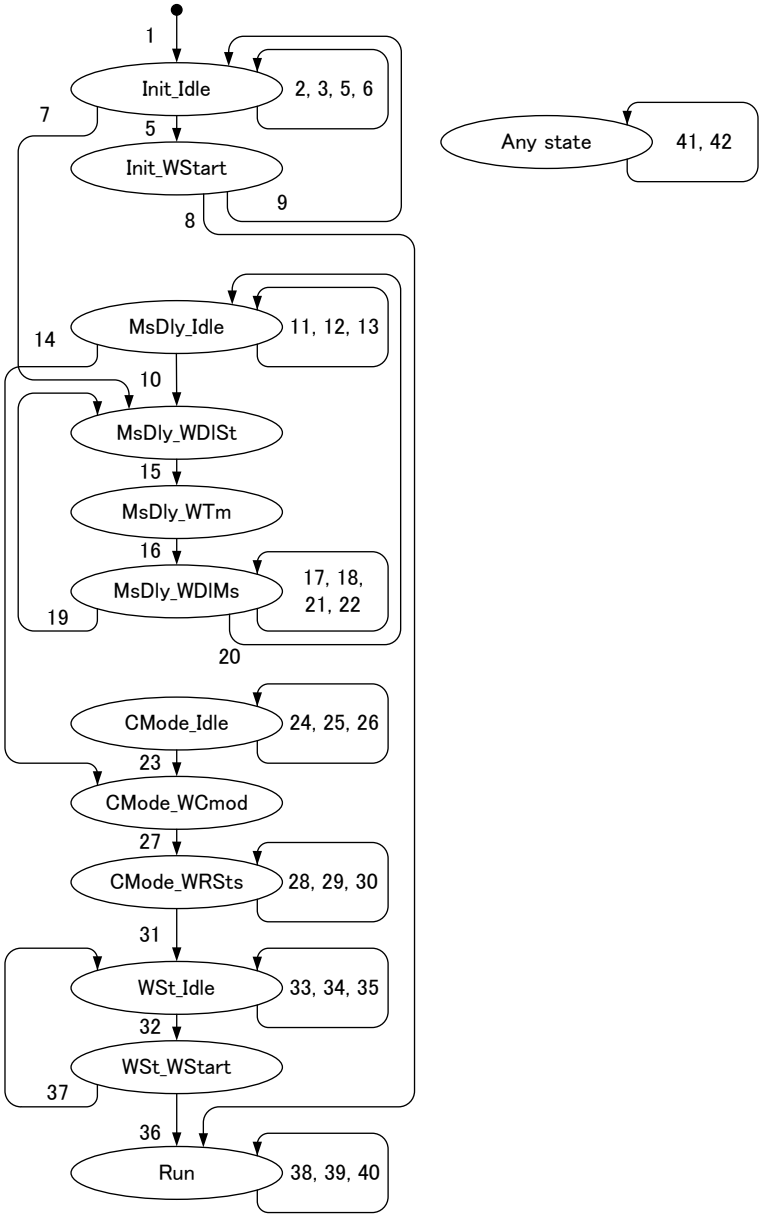


Figure 24 – State diagram of C1 master DLM

Table 71 – State table of C1-Master DLM

No.	Current state	Event /condition =>action	Next state
1	Any state	Power on or DLM-RESET.req => DLM-RESET.cnf CTC_Reset.req { } SRC_Reset.req { }	Init_Idle
2	Init_Idle	DLM-SET-VALUE.req { Var_ID, Val } => Result := SET_VALUE(Var_ID, Val) DLM-SET-VALUE.cnf { Result }	Init_Idle
3	Init_Idle	DLM-GET-VALUE.req { Par_ID } => Result := GET_VALUE(Var_ID, CurVal) DLM-GET-VALUE.cnf { Result, CurVal }	Init_Idle

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
4	Init_Idle	DLM-CLR-ERR.req { Err_ID } => CLR_ERR_STS (Err_ID) DLM-CLR-ERR.cnf { OK }	Init_Idle
5	Init_Idle	DLM-START.req { } / (((V(Cyc_Sel) = CMode_Cyclic) && (V(SlotType) = TSFixed)) || (V(Cyc_Sel) = CMode_Ayclic)) => CTC_Start.req { }	Init_WStart
6	Init_Idle	DLM-START.req { } / (Result = OK) && (V(Cyc_Sel) = CMode_Cyclic) && (V(SlotType) = TSConfig) => DLM-START.cnf { NG }	Init_Idle
7	Init_Idle	DLM-MEAS-DELAY.req { } => V(Nslave) := 0 SRCSDU := BUILD_PDU_DLST(V(Nslave), V(Nmax_dl_cnt)) SRC_Send_Frame.req { SRCSDU} START_TIMER_FR( )	MsDly_WDlSt
8	Init_WStart	CTC_Start.cnf { Result } / (Result = OK) => DLM-START.cnf { Result }	Run
9	Init_WStart	CTC_Start.cnf { Result } / (Result <> OK) => DLM-START.cnf { Result }	Init_Idle
10	MsDly_Idle	DLM-MEAS-DELAY.req { } => V(Nslave) := 0 SRCSDU := BUILD_PDU_DLST(V(Nslave), V(Nmax_dl_cnt)) SRC_Send_Frame.req { SRCSDU} START_TIMER_FR( )	MsDly_WDlSt
11	MsDly_Idle	DLM-SET-VALUE.req { Var_ID, Val } => Result := SET_VALUE(Var_ID, Val) DLM-SET-VALUE.cnf { Result }	MsDly_Idle
12	MsDly_Idle	DLM-GET-VALUE.req { Par_ID } => Result := GET_VALUE(Var_ID, CurVal) DLM-GET-VALUE.cnf { Result, CurVal }	MsDly_Idle
13	MsDly_Idle	DLM-CLR-ERR.req { Err_ID } => CLR_ERR_STS (Err_ID) DLM-CLR-ERR.cnf { OK }	MsDly_Idle
14	MsDly_Idle	DLM-SET-COMMOD.req { } => SRCSDU := BUILD_PDU_COMMOD(V(Cyc_Sel), V(Tcycle), V(Tc2_dly), V(Tidly), V(Tunit)) SRC_Send_Frame.req { SRCSDU}	CMode_WCmod
15	MsDly_WDlSt	SRC_Send_Frame.cnf { } => START_TIMER(T(Tdl_dlms), V(Tdl_dlms))	MsDly_WTm
16	MsDly_WTm	EXPIRED_TIMER (T(Tdl_dlms)) = True => V(Ndly_cnt) = 0 V(Ten) := LATCH_TSTAMP( )	MsDly_WDlMs

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		SRCSDU := BUILD_PDU_DLMS(V(Nslave), 0) SRC_Send_Frame.req { SRCSDU} V(Tst) := LATCH_TSTAMP( )	
17	MsDly_WDlMs	SRC_Send_Frame.cnf { } => (None)	MsDly_WDlMs
18	MsDly_WDlMs	SRC_Rev_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (ST_NO(V(Nslave) = RcvSRCSDU.SA) && (V(Ndly_cnt) <= V(Nmax_dly_cnt))) => V(Ten) := LATCH_TSTAMP( ) V(Tdly) := COMP_DELAY(V(Tst), V(Ten)) SRCSDU := BUILD_PDU_MDLY(V(Nslave), V(Tdly)) SRC_Send_Frame.req { SRCSDU} V(Tst) := LATCH_TSTAMP( ) V(Ndly_cnt) = V(Ndly_cnt) + 1	MsDly_WDlMs
19	MsDly_WDlMs	SRC_Rev_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (ST_NO(V(Nslave) = RcvSRCSDU.SA) && (V(Ndly_cnt) > V(Nmax_dly_cnt)) && (V(Nslave) <= V(Nmax_slave))) => V(Nslave) := V(Nslave) + 1 SRCSDU := BUILD_PDU_DLST(V(Nslave), V(Nmax_dl_cnt)) SRC_Send_Frame.req { SRCSDU}	MsDly_WDlSt
20	MsDly_WDlMs	SRC_Rev_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (ST_NO(V(Nslave) = RcvSRCSDU.SA) && (V(Ndly_cnt) > V(Nmax_dly_cnt)) && (V(Nslave) > V(Nmax_slave))) => STOP_TIMER_FR( ) DLM-MEAS-DELAY.cnf { }	MsDly_Idle
21	MsDly_WDlMs	SRC_Rev_Frame.ind { Rcv_sts, Length, RcvSRCSDU } /((Rcv_sts = OK) && (ST_NO(V(Nslave) <> RcvSRCSDU.SA)) => SRCSDU := BUILD_PDU_MDLY(V(Nslave), V(Tdly)) SRC_Send_Frame.req { SRCSDU} V(Tst) := LATCH_TSTAMP( )	MsDly_WDlMs
22	MsDly_WDlMs	SRC_Rev_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / (Rcv_sts <> OK) => SRCSDU := BUILD_PDU_MDLY(V(Nslave), V(Tdly)) SRC_Send_Frame.req { SRCSDU} V(Tst) := LATCH_TSTAMP( )	MsDly_WDlMs
23	CMode_Idle	DLM-SET-COMMOD.req { } => SRCSDU := BUILD_PDU_COMMOD(V(Cyc_Sel), V(Tcycle), V(Tc2_dly), V(Tidly), V(Tunit)) SRC_Send_Frame.req { SRCSDU}	CMode_WCmod
24	CMode_Idle	DLM-SET-VALUE.req { Var_ID, Val } => Result := SET_VALUE(Var_ID, Val) DLM-SET-VALUE.cnf { Result }	CMode_Idle

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
25	CMode_Idle	DLM-GET-VALUE.req { Par_ID } => Result := GET_VALUE(Var_ID, CurVal) DLM-GET-VALUE.cnf { Result, CurVal }	CMode_Idle
26	CMode_Idle	DLM-CLR-ERR.req { Err_ID } => CLR_ERR_STS (Err_ID) DLM-CLR-ERR.cnf { OK }	CMode_Idle
27	CMode_WCmod	SRC_Send_Frame.cnf { } => V(Nslave) := 0 SRCSDU := BUILD_PDU_STS(V(Nslave)) SRC_Send_Frame.req { SRCSDU}	CMode_WRSts
28	CMode_WRSts	SRC_Send_Frame.cnf { } => (none)	CMode_WRSts
29	CMode_WRSts	SRC_Recv_Frame.ind { Rcv_sts, RcvSRCSDU} / (Rcv_sts = OK) && (V(Nslave) <= V(Nmax_slave)) => UPDATE_STI(V(Nslave), RcvSRCSDU.ST_NO) V(Nslave) := V(Nslave) + 1 SRCSDU := BUILD_PDU_STS(V(Nslave)) SRC_Send_Frame.req { SRCSDU}	CMode_WRSts
30	CMode_WRSts	SRC_Recv_Frame.ind { Rcv_sts, RcvSRCSDU} / (Rcv_sts = OK) && (V(Nslave) > V(Nmax_slave)) => UPDATE_STI(V(Nslave), RcvSRCSDU.ST_NO) DLM-SET-COMMOD.ind { }	WSt_Idle
31	CMode_WRSts	SRC_Recv_Frame.ind { Rcv_sts, RcvSRCSDU} / (Rcv_sts <> OK) => SRCSDU := BUILD_PDU_STS(V(Nslave)) SRC_Send_Frame.req { SRCSDU}	CMode_WRSts
32	WSt_Idle	DLM-START.req { } => CTC_Start.req	WSt_WStart
33	WSt_Idle	DLM-SET-VALUE.req { Var_ID, Val } => Result := SET_VALUE(Var_ID, Val) DLM-SET-VALUE.cnf { Result }	WSt_Idle
34	WSt_Idle	DLM-GET-VALUE.req { Par_ID } => Result := GET_VALUE(Var_ID, CurVal) DLM-GET-VALUE.cnf { Result, CurVal }	WSt_Idle
35	WSt_Idle	DLM-CLR-ERR.req { Err_ID } => CLR_ERR_STS (Err_ID) DLM-CLR-ERR.cnf { OK }	WSt_Idle
36	WSt_WStart	CTC_Start.cnf { Result } / (Result = OK) => DLM-START.cnf { Result }	Run
37	WSt_WStart	CTC_Start.cnf { Result } / (Result <> OK) => DLM-START.cnf { Result }	WSt_Idle
38	Run	DLM-SET-VALUE.req { Var_ID, Val }	Run

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current state	Event /condition =>action	Next state
		=> Result := SET_VALUE(Var_ID, Val) DLM-SET-VALUE.cnf { Result }	
39	Run	DLM-GET-VALUE.req { Par_ID } => Result := GET_VALUE(Var_ID, CurVal) DLM-GET-VALUE.cnf { Result, CurVal }	Run
40	Run	DLM-CLR-ERR.req { Err_ID } => CLR_ERR_STS (Err_ID) DLM-CLR-ERR.cnf { OK }	Run

41	Any state	CTC_Err_Event.ind
		=> FsendEvent := UPDATE_ERR_STS (Err_Event_ID)
		If (FsendEvent = True) then
		DLM-EVENT.ind { Err_Event_ID }
		Endif
			Same state
			(No transition)

42	Any state	SRC_Err_Event.ind
		=> FsendEvent := UPDATE_ERR_STS (Err_Event_ID)
		If (FsendEvent = True) then
		DLM-EVENT.ind { Err_Event_ID }
		Endif
			Same state
			(No transition)

7.3.2 Slave and C2 master

The DLM state transition diagrams for Slave and C2 master are shown in Figure 25 and Table 72.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

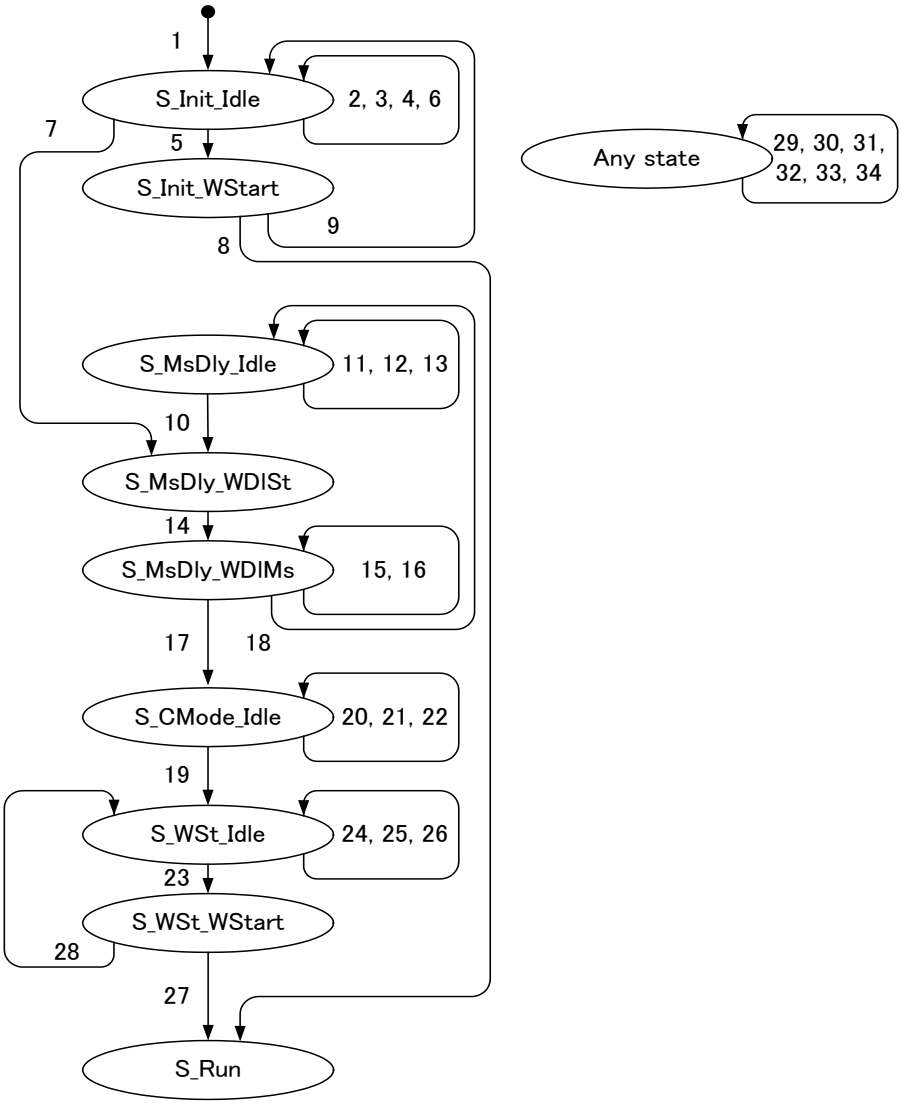


Figure 25 – State diagram of Slave and C2 master DLM

Table 72 – State table of Slave and C2 master DLM

No.	Current stat	Event /condition =>action	Next state
1	Any state	Power on or DLM-RESET.req => DLM-RESET.cnf CTC_Reset.req { } SRC_Reset.req { }	S_Init_Idle
2	S_Init_Idle	DLM-SET-VALUE.req { Var_ID, Val } => Result := SET_VALUE(Var_ID, Val) DLM-SET-VALUE.cnf { Result }	S_Init_Idle
3	S_Init_Idle	DLM-GET-VALUE.req { Par_ID } => Result := GET_VALUE(Var_ID, CurVal) DLM-GET-VALUE.cnf { Result, CurVal }	S_Init_Idle
4	S_Init_Idle	DLM-CLR-ERR.req { Err_ID } => CLR_ERR_STS (Err_ID) DLM-CLR-ERR.cnf { OK }	S_Init_Idle

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current stat	Event /condition =>action	Next state
5	S_Init_Idle	DLM-START.req { } / (((V(Cyc_Sel) = CMode_Cyclic) && (V(SlotType) = TSFixed)) || (V(Cyc_Sel) = CMode_Ayclic)) => CTC_Start.req { }	S_Init_WStart
6	S_Init_Idle	DLM-START.req { } / ((V(Cyc_Sel) = CMode_Cyclic) && (V(SlotType) = TSConfig)) => DLM-START.cnf { NG }	S_Init_Idle
7	S_Init_Idle	DLM-MEAS-DELAY.req { } => (none)	S_MsDly_WDlSt
8	S_Init_WStart	CTC_Start.cnf { Result } / (Result = OK) => DLM-START.cnf { Result }	S_Run
9	S_Init_WStart	CTC_Start.cnf { Result } / (Result <> OK) => DLM-START.cnf { Result }	S_Init_Idle
10	S_MsDly_Idle	DLM-MEAS-DELAY.req { } => (None)	S_MsDly_WDlSt
11	S_MsDly_Idle	DLM-SET-VALUE.req { Var_ID, Val } => Result := SET_VALUE(Var_ID, Val) DLM-SET-VALUE.cnf { Result }	S_MsDly_Idle
12	S_MsDly_Idle	DLM-GET-VALUE.req { Par_ID } => Result := GET_VALUE(Var_ID, CurVal) DLM-GET-VALUE.cnf { Result, CurVal }	S_MsDly_Idle
13	S_MsDly_Idle	DLM-CLR-ERR.req { Err_ID } => CLR_ERR_STS (Err_ID) DLM-CLR-ERR.cnf { OK }	S_MsDly_Idle
14	S_MsDly_WDlSt	SRC_Recv_Frame.ind { Rcv_sts, RcvSRCSDU} / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_TYPE(RcvSRCSDU) = DLST)) => V(Ndly_cnt) := 0 V(Nmax_dly_cnt) := GET_DLY_CNT(RcvSRCSDU) CTC_Enable_LB.req { True, V(Nmax_dly_cnt) } START_TIMER(T(Twrpt), V(Twrpt))	S_MsDly_WDlMs
15	S_MsDly_WDlMs	CTC_Enable_LB.cnf { } => (none)	S_MsDly_WDlMs
16	S_MsDly_WDlMs	SRC_Recv_Frame.ind { Rcv_sts, RcvSRCSDU} / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_TYPE(RcvSRCSDU) = DLMS) && (V(Ndly_cnt) <= V(Nmax_dly_cnt)) => SAVE_DELAY(RcvSRCSDU) V(Ndly_cnt) := V(Ndly_cnt) + 1	S_MsDly_WDlMs
17	S_MsDly_WDlMs	SRC_Recv_Frame.ind { Rcv_sts, RcvSRCSDU} / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_TYPE(RcvSRCSDU) = DLMS) && (V(Ndly_cnt) > V(Nmax_dly_cnt)) => STOP_TIMER(T(Twrpt), V(Twrpt))	S_CMode_WCMod

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current stat	Event /condition =>action	Next state
		DLM-MEAS-DELAY.cnf { OK }	
18	S_MsDly_WDlMs	EXPIRED_TIMER(T(Twrpt)) => STOP_TIMER(T(Twrpt), V(Twrpt)) CTC_Enable_LB.req { False, 0 } DLM-MEAS-DELAY.cnf { NG }	S_MsDly_Idle
19	S_CMode_Idle	DLM-SET-COMMOD.ind { } => V(Cyc_Sel) := P(Cyc_Sel) V(Tcycle) := P(Tcycle) V(Tc2_dly) := P(Tc2_dly) V(Tidly) := P(Tc2_dly) V(Tunit) := P(Tunit)	S_WSt_Idle
20	S_CMode_Idle	DLM-SET-VALUE.req { Var_ID, Val } => Result := SET_VALUE(Var_ID, Val) DLM-SET-VALUE.cnf { Result }	S_CMode_Idle
21	S_CMode_Idle	DLM-GET-VALUE.req { Par_ID } => Result := GET_VALUE(Var_ID, CurVal) DLM-GET-VALUE.cnf { Result, CurVal }	S_CMode_Idle
22	S_CMode_Idle	DLM-CLR-ERR.req { Err_ID } => CLR_ERR_STS (Err_ID) DLM-CLR-ERR.cnf { OK }	S_CMode_Idle
23	S_WSt_Idle	DLM-START.req { } => CTC_Start.req	S_WSt_WCTC
24	S_WSt_Idle	DLM-SET-VALUE.req { Var_ID, Val } => Result := SET_VALUE(Var_ID, Val) DLM-SET-VALUE.cnf { Result }	S_WSt_Idle
25	S_WSt_Idle	DLM-GET-VALUE.req { Par_ID } => Result := GET_VALUE(Var_ID, CurVal) DLM-GET-VALUE.cnf { Result, CurVal }	S_WSt_Idle
26	S_WSt_Idle	DLM-CLR-ERR.req { Err_ID } => CLR_ERR_STS (Err_ID) DLM-CLR-ERR.cnf { OK }	S_WSt_Idle
27	S_WSt_WCTC	CTC_Start.cnf { Result } / (Result = OK) => DLM-START.cnf { Result }	S_Run
28	S_WSt_WCTC	CTC_Start.cnf { Result } / (Result = OK) => DLM-START.cnf { Result }	S_WSt_Idle

29	Any state	SRC_Err_Event.ind
		=> FsendEvent := UPDATE_ERR_STS (Err_Event_ID)
		If (FsendEvent = True) then
		DLM-EVENT.ind { Err_Event_ID }
		Endif
			Same state
			(No transition)

30	Any state	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) &&	Same state

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

No.	Current stat	Event /condition =>action	Next state
		(GET_TYP(RcvSRCSDU) = STS)) => SndSRCSDU := BUILD_PDU_STS( ) SRC_Send_Frame.req { Node_ID_C1, Len_Sts, SndSRCSDU }	
31	Any state (See NOTE)	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (RcvSRCSDU.TYPE <> STS)) => (none)	Same state
32	Any state	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && ((GET_DA(RcvSRCSDU) <> V(MA)) => (none)	Same state
33	Any state	SRC_Send_Frame.cnf { } => (none)	Same state
34	Any state	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts <> OK) => (none)	Same state

7.4 Functions

The list of the functions which used by DLM protocol machine is shown in Table 73.

Table 73 – The list of the functions used by DLM protocol machine

Function name	Input	Output	Operation
GET_VALUE	Var_ID	CurVal	This function reads the current value of the specified variable.
SET_VALUE	Var_ID, Val		This function stores the specified value into the specified variable.
BUILD_PDU_DLST	Node_ID, Nmax_dl_cnt	SRCSDU	This function builds SRCSDU to request SRC to send delay measurement start frame.
BUILD_PDU_DLMS	Node_ID, Tdly	SRCSDU	This function builds SRCSDU to request SRC to send delay measurement frame.
BUILD_PDU_COMMOD	Cyc_Sel, Tcycle, Tc2_dly, Tidly, Tunit	SRCSDU	This function builds SRCSDU to request SRC to send cyclic information frame.
BUILD_PDU_STS	Node_ID	SRCSDU	This function builds SRCSDU to request SRC to send status frame.
START_TIMER	Tim_ID, Val	None	(See Table 53)
STOP_TIMER	Tim_ID		(See Table 53)
START_TIMER_FR	None	None	This function starts free-run timer.
STOP_TIMER_FR	None	None	This function stops free-run timer.
LATCH_TSTAMP		Timestamp	This function returns the current value of free-run timer.
UPDATE_STI	Node_ID, Sts		This function updates the connection status of the specified station in the data specified with Sts.
UPDATA_ERR_STS	Err_Event_ID	FSendEvent	This function saves the error factors that notified by CTC and SRC. It returns True when the notified error is a new error and returns False when detected error by the

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Function name	Input	Output	Operation
			return value.
GET_DA	SRCPDU		This function gets the error factors that notified by CTC and SRC. It returns True when the notified error is a new error and returns False when detected error by the return value.
GET_TYPE	SRCSDU		This function takes out the value of the type field of the specified SRCSDU.
GET_DLY_CNT	SRCSDU		This function takes out the value of the measured delay field of the specified SRCSDU.
SAVE_DLY	SRCSDU		This function updates V(Tdly) with the value of the measured delay field of the specified SRCSDU.
CLR_ERR_STS	Err_ID		This function clears the error factor specified with Err_ID. Even if this function is executed, the error for which power on or reset is necessary cannot be cleared.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Bibliography

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

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*

_____

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

SOMMAIRE

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

_____

RÉSEAUX DE COMMUNICATION INDUSTRIELS – SPÉCIFICATIONS DES BUS DE TERRAIN –

Partie 4-24: Spécification du protocole de la couche liaison de données – Éléments de type 24

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.

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 protocole sont spécifiées dans la CEI 61784-1 et la CEI 61784-2.

La Norme internationale CEI 61158-4-24 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.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

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. A cette date, la publication sera

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

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

INTRODUCTION

La présente partie de la CEI 61158 est l'une d'une série produite pour faciliter l'interconnexion de composants d'un système d'automation. Elle est liée à d'autres normes de la série telle que définie par le modèle de référence des 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 que doivent réaliser des entités de liaison de données (DLE) homologues 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 prioritaires dans le modèle OSI.

La présente norme traite, en particulier, de la communication et de l'interaction des capteurs, effecteurs et autres appareils d'automatisation. Grâce à l'utilisation conjointe de la présente norme avec d'autres normes entrant dans les modèles de référence de l'OSI ou des bus de terrain, des systèmes auparavant incompatibles peuvent fonctionner ensemble dans toute combinaison.

NOTE L'attention est attirée sur le fait que l'utilisation de certains types de protocole associés est limitée par leurs détenteurs de droit à la propriété intellectuelle. Dans tous les cas, l'engagement pris par les détenteurs quant à une diffusion limitée desdits droits de propriété intellectuelle permet d'utiliser un type particulier de protocole de couche liaison de données avec des protocoles de couche physique et de couche application dans les combinaisons de types explicitement spécifiées dans les séries de profils. L'utilisation des divers types de protocoles dans d'autres combinaisons peut nécessiter une autorisation de la part de leurs détenteurs de droits à la propriété intellectuelle respectifs.

La Commission Electrotechnique Internationale (CEI) attire l'attention sur le fait qu'il est déclaré que la conformité avec les dispositions du présent document peut impliquer l'utilisation de brevets comme décrit ci-dessous, la notation [xx] indiquant le détenteur des droits de brevet:

US 8223804 JP 4760978 CN 200880002225.3 EPC 08738862.5 KR 10-2009-7011514 TW 97111183	[YE]	APPAREIL DE COMMUNICATION, SYSTÈME SYNCHRONISÉ ET PROCÉDÉ DE COMMUNICATION SYNCHRONISÉ
US 7769935 JP 4683346 US 8046512 EPC 07850686.2 TW 96150287	[YE]	SYSTÈME DE COMMUNICATION MAÎTRE ESCLAVE ET PROCÉDÉ DE COMMUNICATION MAÎTRE ESCLAVE
JP 4356698	[YE]	APPAREIL DE COMMUNICATION, SYSTÈME DE COMMUNICATION SYNCHRONISÉ ET PROCÉDÉ DE COMMUNICATION SYNCHRONISÉ

La CEI ne prend pas position quant à la preuve, à la validité et à la portée de ces droits de propriété.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Les détenteurs de ces droits de propriété ont donné l'assurance à l'ISO [et/ou] à la CEI qu'ils consentent à négocier des licences avec des demandeurs du monde entier, soit sans frais soit à des termes et conditions raisonnables et non discriminatoires. À ce propos, la déclaration du détenteur des droits de propriété est enregistrée à la CEI. Des informations peuvent être demandées à:

[YE] YASKAWA ELECTRIC CORPORATION
2-1 Kurosakishiroishi, Yahatanishi-ku, Kitakyushu 806-0004, Japon
À l'attention du Service des droits de propriété intellectuelle.

L'attention est d'autre part attirée sur le fait que certains des éléments du présent document peuvent faire l'objet de droits de propriété autres que ceux qui ont été mentionnés ci-dessus. La CEI ne saurait être tenue pour responsable de l'identification de ces droits de propriété en tout ou partie.

L'ISO (www.iso.org/patents) et la CEI (http://patents.iec.ch) maintiennent des bases de données, consultables en ligne, des droits de propriété pertinents à leurs normes. Les utilisateurs sont encouragés à consulter ces bases de données pour obtenir l'information la plus récente concernant les droits de propriété.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

RÉSEAUX DE COMMUNICATION INDUSTRIELS – SPÉCIFICATIONS DES BUS DE TERRAIN –

Partie 4-24: Spécification du protocole de la couche liaison de données – Éléments de type 24

1 Domaine d'application

1.1 Généralités

La Couche liaison de données assure les communications de messagerie élémentaires prioritaires entre les appareils d'un environnement d'automatisation.

Ce protocole offre à toutes les entités de liaison de données participantes

a) des opportunités de communication cyclique à démarrage synchrone, selon un ordre préétabli, ou

b) de manière acyclique, de la façon requise par chacune de ces entités de liaison de données.

Ainsi, ce protocole peut être qualifié de protocole offrant un accès cyclique et acyclique asynchrone, mais avec un redémarrage synchrone de chaque cycle.

1.2 Spécifications

La présente norme spécifie:

a) les procédures de transfert en temps opportun des données et des informations de commande entre une entité utilisateur de liaison de données et une entité utilisateur homologue et, parmi les entités de liaison de données formant le fournisseur de services distribués de liaison de données;

b) les procédures pour donner des opportunités de communication à toutes les entités DL participantes, séquentiellement et de manière cyclique pour le transfert déterministe et synchronisé à intervalles cycliques pouvant aller jusqu'à 64 ms;

c) les procédures pour donner les opportunités de communication disponibles pour la transmission de données prioritaires ainsi que pour la transmission de données non prioritaires, sans nuire à la transmission de données prioritaire;

d) les procédures pour donner des opportunités de communication cyclique et acyclique pour la transmission de données prioritaires avec accès hiérarchisés;

e) les procédures pour donner des opportunités de communication selon le contrôle d'accès au support physique normalisé par l'ISO/CEI 8802-3, avec des dispositions pour les nœuds à ajouter ou à retirer lors du 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) homologues par l'échange de DLPDU de bus de terrain;

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

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

c) d'interactions entre un fournisseur DLS et un fournisseur de service Ph au sein du même système par l'échange de primitives de service 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 à différents besoins de communications prioritaires.

1.5 Conformité

La présente norme spécifie également les exigences relatives aux systèmes mettant mise en œuvre ces procédures. La présente norme ne fournit pas d'essais destinés 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).

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 61158-2, *Réseaux de communication industriels – Spécifications des bus de terrain – Partie 2: Spécifications et définition des services de la couche physique*

CEI 61158-3-24:2014, *Réseaux de communication industriels – Spécifications des bus de terrain – Partie 3-24: Définition des services de la couche liaison de données – Eléments de type 24*

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/IEC 9899, *Information technology – Programming languages – C* (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*

ISO/IEC 13239:2002, *Information technology – Telecommunications and information exchange between systems – High-level data link control (HDLC) procedures* (disponible en anglais seulement)

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

ISO/IEC 19501:2005, *Information technology – Open Distributed Processing – Unified Modeling Language (UML) Version 1.4.2* (disponible en anglais seulement)

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 du modèle de référence

La présente norme repose en partie sur les concepts développés dans l'ISO/CEI 7498-1 et l'ISO/CEI 7498-3, et utilise les termes suivants.

3.1.1 **acquittement** [ISO/CEI 7498-1]

3.1.2 **entités (N) correspondantes** [ISO/CEI 7498-1]

entités DL correspondantes (N=2)

entités Ph correspondantes (N=1)

3.1.3 **adresse DL** [ISO/CEI 7498-3]

3.1.4 **protocole DL** [ISO/CEI 7498-1]

3.1.5 **unité de données de protocole DL** [ISO/CEI 7498-1]

3.1.6 **unité de données de service DL** [ISO/CEI 7498-1]

3.1.7 **utilisateur de DLS** [ISO/CEI 7498-1]

3.1.8 **données d'utilisateur de DLS** [ISO/CEI 7498-1]

3.1.9 **Événement** [ISO/CEI 19501]

3.1.10 **gestion de couche** [ISO/CEI 7498-1]

3.1.11 **nom de primitive** [ISO/CEI 7498-1]

3.1.12 **Reset** [ISO/CEI 7498-1]

3.1.13 **Segmentation** [ISO/CEI 7498-1]

3.1.14 **État** [ISO/CEI 19501]

3.1.15 **diagramme d'états** [ISO/CEI 19501]

3.1.16 **gestion-système** [ISO/CEI 7498-1]

3.1.17 **Transition** [ISO/CEI 19501]

3.1.18 **entité (N)** [ISO/CEI 7498-1]

entité DL (N=2)

entité Ph (N=1)

3.1.19 **couche (N)** [ISO/CEI 7498-1]

couche DL (N=2)

couche Ph (N=1)

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

service (N) [ISO/CEI 7498-1]

service DL (N=2)

service Ph (N=1)

point d'accès au service (N) [ISO/CEI 7498-1]

point d'accès au service DL (N=2)

point d'accès au service Ph (N=1)

3.2 Termes, définitions et conventions des services

La présente norme utilise également les termes suivants définis dans l'ISO/CEI 10731 tels qu'ils s'appliquent à la Couche liaison de données:

3.2.1 (primitive de) confirmation
3.2.2 primitive de service DL;
3.2.3 fournisseur de service DL
3.2.4 utilisateur de service DL
3.2.5 (primitive d') indication
3.2.6 (primitive de) demande
3.2.7 demandeur
3.2.8 (primitive de) réponse

3.3 Termes et définitions communs

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

3.3.1
transmission acyclique
échange non périodique de télégrammes

3.3.2
maître C1
un des types de station qui lance et contrôle la transmission cyclique

3.3.3
message C1
communication de message dont le maître C1 est à l'origine pour échanger des messages avec l'esclave ou le maître C2

3.3.4
maître C2
un des types de station ayant pour fonction de surveiller toutes les données de processus transmises sur le réseau et qui peut générer une communication de message

3.3.5
message C2
communication de message dont le maître C2 est à l'origine pour échanger des messages avec l'esclave ou le maître C1

3.3.6
transmission cyclique
échange périodique de télégrammes

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

3.3.7
données
terme générique utilisé pour désigner toute information transmise via un bus de terrain

3.3.8
appareil
entité physique connectée au bus de terrain, constituée d'au moins un élément de communication (l'élément réseau) et qui peut être un élément de commande et/ou un élément final (transducteur, actionneur, etc.)

3.3.9
mode événementiel
mode de transmission du protocole de couche d'application du type de communication 24 dans lequel une transaction commande-réponse-échange se déroule au fur et à mesure des demandes de l'utilisateur

3.3.10
trame
synonyme de DLPDU

3.3.11
initiateur
station à l'origine de l'échange des données de processus ou du message

3.3.12
interface
frontière commune entre deux unités fonctionnelles, définie par des caractéristiques fonctionnelles, des caractéristiques de signal ou d'autres caractéristiques appropriées

3.3.13
données d'entrée
données de processus envoyées par l'esclave et reçues par le maître C1

3.3.14
message
série ordonnée d'octets destinée à communiquer des informations

Note 1 à l'article: En principe, utilisé pour acheminer des informations entre des homologues au niveau de la couche d'application.

3.3.15
esclave de surveillance
esclave chargé de la surveillance de toutes les données de processus transmises sur le réseau

3.3.16
réseau
ensemble de nœuds reliés par un certain type de support de communication, avec d'éventuels répéteurs, ponts, routeurs et passerelles de couche inférieure intermédiaires

3.3.17
nœud
a) entité DL simple telle qu'elle apparaît sur une liaison locale

b) point d'extrémité d'une liaison dans un réseau ou point de rencontre d'au moins deux liaisons

[SOURCE: CEI 61158-2]

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

3.3.18
données de sortie
données de processus envoyées par le maître C1 et reçues par les esclaves

3.3.19
protocole
convention à l'égard des formats de données, des suites chronologiques et de la correction d'erreurs dans le cadre de l'échange de données des systèmes de communication

3.3.20
communication en temps réel
transfert de données en temps réel

3.3.21
utilisateur de DLS destinataire
utilisateur de service DL auquel sont destinées les données utilisateur DL

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

3.3.22
répondeur
station qui envoie des données de processus ou des messages après avoir été amorcée par l'initiateur

3.3.23
transmission de données avec acquittement
service de transfert de données avec acquittement de la part de la DLE correspondante

3.3.24
transmission de données sans acquittement
service de transfert de données sans acquittement de la part de la DLE correspondante

3.3.25
esclave
un des types de station qui accède au support uniquement après avoir été initialisé par le maître C1 ou le maître C2

3.3.26
utilisateur DLS expéditeur
utilisateur du service DL à la source des données utilisateur DL

3.3.27
station
nœud

3.3.28
topologie
architecture de réseau physique relative à la connexion entre les stations du système de communication

3.3.29
cycle de transmission
période fixe de transmission cyclique

3.3.30
intervalle de temps
période réservée pour que l'initiateur et le répondeur puissent échanger une trame respectivement

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

3.4 Symboles et abréviations

3.4.1	**DA**	Destination address (Adresse de destination)
3.4.2	**DL-**	Data-link layer (Couche liaison de données) (préfixe)
3.4.3	**DLE**	DL-entity (Entité DL) (instance active locale de la couche liaison de données)
3.4.4	**DLL**	DL-layer (Couche Liaison de données)
3.4.5	**DLM**	DL-management (Gestion DL)
3.4.6	**DLME**	DL-management entity (Entité de gestion de DL) (instance active locale de la gestion de DL)
3.4.7	**DLMS**	DL-management service (Service de gestion de DL)
3.4.8	**DLPDU**	DL-protocol-data-unit (Unité de données de protocole DL)
3.4.9	**DLS**	DL-service (Service DL)
3.4.10	**DLSAP**	DL-service-access-point (Point d'accès au service DL)
3.4.11	**DLSDU**	DL-service-data-unit (Unité de données de service DL)
3.4.12	**FIFO**	First-in first-out (premier entré, premier sorti) (méthode de mise en file d'attente)
3.4.13	**ID**	Identifiant
3.4.14	**OSI**	Open Systems Interconnection (Interconnexion de systèmes ouverts)
3.4.15	**PDU**	Protocol Data Unit (Unité de données de protocole)
3.4.16	**Ph-**	Physical layer (Couche physique) (préfixe)
3.4.17	**PhE**	Ph-entity (Entité Ph) (instance locale active de la couche physique)
3.4.18	**PhL**	Ph-layer (Couche Ph)
3.4.19	**PHY**	Physical layer device (Appareil de couche physique) (spécifié dans l'ISO/CEI 8802-3)
3.4.20	**QoS**	Quality of service (Qualité de service)
3.4.21	**RT**	Real-time (Temps réel)
3.4.22	**SAP**	Service access point (Point d'accès au service)
3.4.23	**SDU**	Service data unit (Unité de données de service)

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

3.5 Symboles et abréviations supplémentaires de type 24

3.5.1	**ACK**	Acknowledge (Acquittement)
3.5.2	**C1MSG**	C1 message (message C1)
3.5.3	**C2MSG**	C2 message (message C2)
3.5.4	**E/S**	Entrée et/ou sortie
3.5.5	**MSG**	Message
3.5.6	**Rx**	Réception
3.5.7	**SDA**	Send data with acknowledge (Transmission de données avec acquittement)
3.5.8	**SDN**	Send data with no-acknowledge (Transmission de données sans acquittement)
3.5.9	**SM**	State machine (Diagramme d'états)
3.5.10	**Tcycle**	Transmission cycle (Cycle de transmission)
3.5.11	**Tslot**	Time slot (Intervalle de temps)
3.5.12	**Tx**	Transmission

3.6 Conventions générales

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 temps-séquence utilisés sont des descriptions totalement abstraites; ils ne constituent pas une spécification pour une mise en œuvre.

Les primitives de service sont utilisées pour représenter les interactions entre utilisateur de service et fournisseur de service (ISO/CEI 10731). Elles acheminent des paramètres qui indiquent des 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, avec une colonne pour ces primitives et pour les directions de transfert de paramètres utilisés par le DLS:

- les paramètres d'entrée de la primitive de demande;
- les paramètres de sortie de la primitive d'indication;
- les paramètres d'entrée de la primitive de réponse;
- les paramètres de sortie de la primitive de confirmation.

NOTE Les primitives de demande, d'indication, de réponse et de confirmation sont également connues respectivement comme les primitives requestor.submit, acceptor.deliver, acceptor.submit, et requestor.deliver (voir ISO/CEI 10731).

Un paramètre (ou un composant 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 spécifiée dans la colonne:

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

M Le paramètre est obligatoire pour la primitive.

U Le paramètre est une option de l'utilisateur et peut ou peut ne pas être fourni, cela dépendant de l'usage dynamique de l'utilisateur 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 DLS.

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

Certaines entrées sont en plus qualifiées par des éléments entre parenthèses. Ceux-ci peuvent être une contrainte spécifique au paramètre:

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

Dans une interface particulière, il n'est pas indispensable d'énoncer tous les paramètres de façon explicite. Certains paramètres peuvent être implicitement associés à la primitive.

Dans les diagrammes qui illustrent ces interfaces, des traits discontinus indiquent les relations cause-effet ou temps-séquence, alors que des lignes ondulées indiquent que les événements sont approximativement contemporains.

3.7 Conventions de type 24 supplémentaires

3.7.1 Conventions pour les primitives

La notation suivante, forme raccourcie des classes de primitive définies en 3.2, est utilisée dans les figures.

req primitive de demande

ind primitive d'indication

cnf primitive de confirmation

3.7.2 Conventions pour les diagrammes d'états

Les séquences de protocole sont décrites au moyen de diagrammes d'états.

Dans les diagrammes d'état, les états sont représentés par des cases et les transitions d'état par des flèches.

Les noms des états et des transitions du diagramme d'état correspondent aux noms indiqués dans la table d'états. Cette liste de transitions d'états est structurée comme indiqué dans le Tableau 1.

Tableau 1 – Descriptions des transitions d'état

N°	État courant	Événement /condition =>action	État suivant

La description des éléments d'un diagramme d'états est donnée dans le Tableau 2.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Tableau 2 – Description des éléments d'un diagramme d'états

Élément de description	Signification
N°	Numéro de la transition.
État courant, État suivant	Noms de l'état de départ et de l'état de transition cible.
Événement	Nom ou description de l'événement déclencheur qui provoque la transition.
/ conditions	Expression booléenne, qui doit être vraie pour que la transition ait lieu.
=>action	Liste des affectations et des appels de service ou de fonction. Il convient que l'action soit atomique. Le signe "=>" qui précède ne fait pas partie de l'action.
NOTE "/ conditions" peut être omis.	

Les conventions utilisées dans les diagrammes d'états sont montrées dans le Tableau 3.

Tableau 3 – Conventions utilisées dans les diagrammes d'états

Convention	Signification
+ - * /	Opérateurs arithmétiques
:=	La valeur d'une entité de gauche est remplacée par celle d'une entité de droite. Si une entité de droite est un paramètre, celui-ci provient de la primitive identifiée comme un événement d'entrée.
=	Condition logique indiquant qu'une entité de gauche est égale à une entité de droite.
<	Condition logique indiquant qu'une entité de gauche est inférieure à une entité de droite.
>	Condition logique indiquant qu'une entité de gauche est supérieure à une entité de droite.
<=	Condition logique indiquant qu'une entité de gauche est inférieure ou égale à une entité de droite.
>=	Condition logique indiquant qu'une entité de gauche est supérieure ou égale à une entité de droite.
<>	Condition logique indiquant qu'une entité de gauche n'est pas égale à une entité de droite.
&&	"ET" logique
||	"OU" logique

4 Présentation du protocole DL

4.1 Fonctionnalité caractéristique du protocole DL

Le Tableau 4 montre les fonctionnalités caractéristiques du protocole DL de Type 24.

Tableau 4 – Fonctionnalités caractéristiques du protocole de liaison de données de bus de terrain

Profils	Description
Type de station et nombre maximal de stations	- maître C1 (station active avec commande d'accès au bus, 1 station (obligatoire)) - maître C2 (station active avec commande d'accès restreint au bus, max. 1 station (facultative)) - esclave (stations passives sans commande d'accès au bus, max. 62)
Adressage des stations	1 à 255 (255 = adresses globales pour les messages de diffusion), extension d'adresse de largeur 8 bits pour appareil intégré
Cycle de transmission	31,25 µs à 64 ms
Taille DLSDU	8 à 64 octets
Caractéristique de transmission	- Échange de données cycliques et événement cyclique, synchronisé avec

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Profils	Description
	une durée de cycle précise (gigue inférieure à 1 μs) - Nombre maximal de nouvelles tentatives: 62 (n fois / 1 station), à l'intérieur de la durée de cycle - Transmission de messages acyclique

Il existe trois types de stations: maître C1, maître C2 et esclave. L'échange de données est exécuté entre une station maître (maître C1 ou maître C2) et N stations esclaves. Ce protocole supporte deux modes de communication, la transmission cyclique et la transmission acyclique.

En mode de transmission cyclique, la transmission est exécutée de manière cyclique avec une période précise. Le cycle de transmission prend une valeur fixée par le maître C1, située dans la plage allant de 31,25 [μs] à 64 [ms]. Étant donné que la valeur fixée pour le cycle de transmission est spécifique à la ligne de transmission, tous les esclaves connectés doivent supporter cette valeur. Il n'est pas permis de fixer différentes valeurs de cycles de transmission pour les esclaves connectés au même réseau.

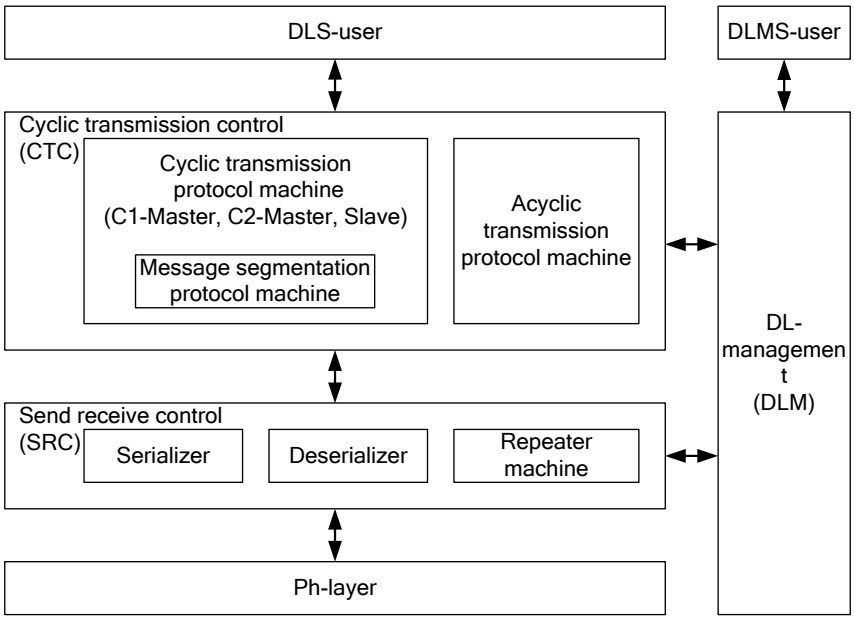
Le cycle de transmission dispose d'une bande d'échange de données d'E/S pour émettre les données de procédé et d'une bande de communication de messages pour l'transmission des messages. La machine protocolaire du maître C1 commande la séquence de transmission du mode de transmission cyclique. La période de temps utilisée par une station maître pour les échanges avec une station esclave s'appelle un intervalle de temps. Il existe deux types de séquences de communication; l'un est le "type à intervalle de temps de largeur fixe", dont l'intervalle de temps est de même largeur pour toutes les stations, l'autre est le "type à intervalle de temps configurable", dont l'intervalle de temps peut être défini pour chaque station. Toutes les stations doivent utiliser des trames de même longueur de données lorsqu'elles utilisent le type à intervalle de temps de largeur fixe. La largeur de l'intervalle de temps est statique dans les deux types, et la largeur est définie par la gestion de DL durant l'initialisation. Une fois que la communication cyclique a commencé, elle ne doit pas être modifiée.

Le mode de transmission acyclique est utilisé par l'utilisateur de DLS qui fonctionne en mode commandé par événement. En mode de transmission acyclique, les transmissions sont exécutées sporadiquement. La même séquence de transmission et la même communication de messages peuvent être exécutées en transmission acyclique, comme en mode de transmission cyclique mais sans que l'on fixe le cycle de transmission.

4.2 Composant de couche DL

La couche DL se compose de trois sous-couches: CTC (Cyclic transmission control - commande de transmission cyclique), SRC (Send Receive Control - commande d'envoi et réception) et DLM (Data-link management - gestion de liaison de données). La SRC se positionne sur la couche inférieure de la CTC; la DLM couvre à la fois la sous-couche CTC et la sous-couche SRC. Le composant de la couche liaison de données est présenté à la Figure 1.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe



Légende

Anglais	Français
DLS-user	Utilisateur de DLS
DLMS-user	Utilisateur de DLMS
Cyclic transmission control (CTC)	Commande de transmission cyclique (CTC)
Cyclic transmission protocol machine (C1-Master, C2-Master, Slave)	Machine protocolaire de transmission cyclique (maître C1, maître C2, esclave)
Message segmentation protocol machine	Machine protocolaire de segmentation de message
Acyclic transmission protocol machine	Machine protocolaire de transmission acyclique
DL-Management (DLM)	Gestion DL (DLM)
Send receive control (SRC)	Commande d'envoi et réception (SRC)
Serializer	Sérialiseur (convertisseur parallèle-série)
Deserializer	Désérialiseur (convertisseur série-parallèle)
Repeater machine	Machine de répétition
Ph-layer	Couche Ph

Figure 1 – Composant de la couche liaison de données

4.2.1 Commande de transmission cyclique (CTC)

Il s'agit de la sous-couche qui construit la DLPDU et exécute une machine protocolaire. Elle possède deux modes de communication: le mode de transmission cyclique et le mode de transmission acyclique. La CTC exécute l'un ou l'autre, selon la demande émise par l'utilisateur du DLMS.

4.2.2 Commande d'envoi et réception (SRC)

La SRC envoie ou reçoit des trames sur demande de la sous-couche CTC. Elle est sérialisée ou désérialisée en fonction du PHY correspondant. Lorsque la SRC met en œuvre deux ports PHY ou plus, elle fournit une fonction de répétition de trame entre les ports PHY mis en œuvre.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

4.2.3 Gestion de DL

Il s'agit d'une sous-couche qui configure le fonctionnement des DLE en positionnant des variables internes et gère les erreurs détectées par chaque sous-couche.

4.3 Séquence chronologique

4.3.1 Présentation

Il existe deux types de mode de transmission, le mode de transmission cyclique et le mode de transmission acyclique. La transmission cyclique est de deux types; l'un est le "type à intervalle de temps de largeur fixe", dont l'intervalle de temps est de même largeur pour toutes les stations, l'autre est le "type à intervalle de temps configurable", dont l'intervalle de temps peut être défini pour chaque station.

La largeur de l'intervalle de temps est statique dans les deux types, et la largeur est définie par la gestion de DL durant l'initialisation. Une fois que la communication cyclique a commencé, elle ne doit pas être modifiée.

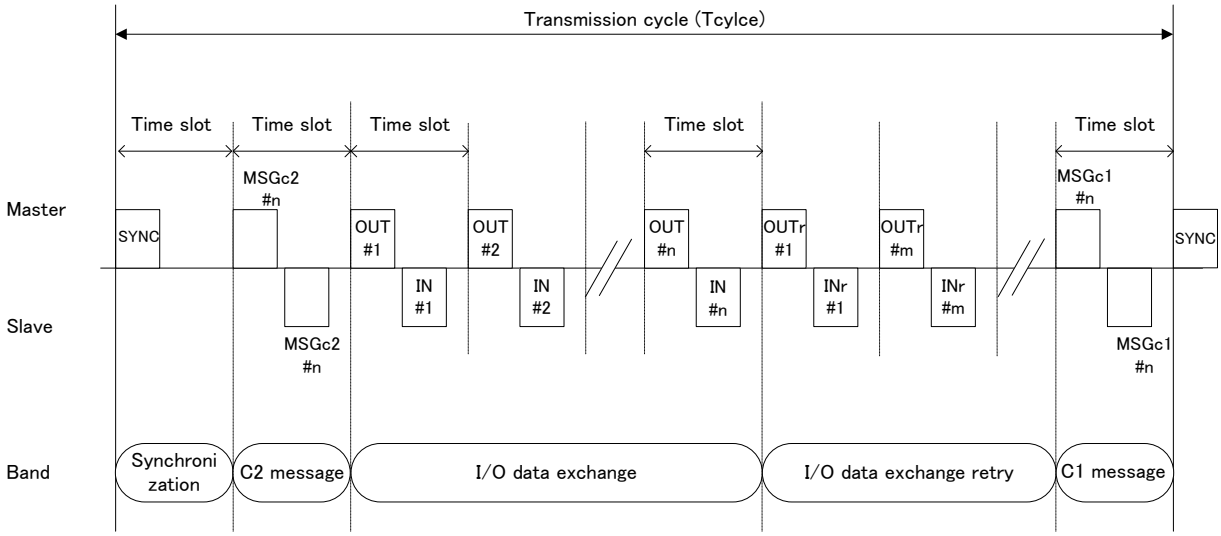
4.3.2 Mode de transmission cyclique

4.3.2.1 Type à intervalle de temps de largeur fixe

4.3.2.1.1 Présentation

La Figure 2 montre la séquence de transmission du type à intervalle de temps de largeur fixe. Dans ce type, la largeur des intervalles de temps qui sont alloués pour l'exécution des échanges de données de procédé, l'un après l'autre, entre le maître et l'esclave, est identique pour tous les esclaves. Un ou plusieurs intervalles de temps sont alloués à chacune des largeurs de bande représentées sur le chronogramme.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe



SYNC: Synchronous frame
OUT #n: Output data to slave #n
IN #n: Input data from slave #n
OUTr #m: Output data retry to slave #m
INr #m: Input data retry from slave #m
MSGc1 #n: C1 message to slave #n
MSGc2 #n: C2 message to slave #n

Légende

Anglais	Français
Transmission cycle (Tcycle)	Cycle de transmission (Tcycle)
Master	Maître
Slave	Esclave
Band	Bande
Time slot	Intervalle de temps
Synchronization	Synchronisation
C2 message	Message C2
I/O data exchange	Échange de données d'E/S
I/O data exchange retry	Nouvelle tentative d'échange de données d'E/S
C1 message	Message C1
Synchronous frame	Trame synchrone
Output data to slave #n	Données de sortie vers l'esclave n° n
Input data from slave #n	Données d'entrée provenant de l'esclave n° n
Output data retry to slave #m	Nouvelle tentative de données de sortie vers l'esclave n° m
Input data retry from slave #m	Nouvelle tentative de données d'entrée provenant de l'esclave n° m
C1 message to slave #n	Message C1 vers l'esclave n° n
C2 message to slave #n	Message C2 vers l'esclave n° n

Figure 2 – Chronogramme de la communication cyclique de type à intervalle de temps de largeur fixe

4.3.2.1.2 Description détaillée de la bande de communication

4.3.2.1.2.1 Synchronisation

Il s'agit d'une largeur de bande au moyen de laquelle le maître C1 diffuse une trame synchrone vers l'esclave et le maître C2. Un intervalle de temps est alloué à cette largeur de

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

bande. À l'intérieur de cette largeur de bande, seule la transition de synchronisation de la trame émise par le maître C1 est admise; ni l'esclave ni le maître C2 n'ont le droit d'émettre de trame.

4.3.2.1.2.2 Message C2

Il s'agit d'une largeur de bande pour la transmission de message (transmission de message C2), le maître C2 étant le client (station principale), le maître C1 ou l'esclave étant le serveur (station secondaire). Un intervalle de temps est alloué à cette largeur de bande, et la demande et la réponse sont émises une fois, respectivement.

4.3.2.1.2.3 Échange de données d'E/S

Il s'agit d'une largeur de bande au moyen de laquelle le maître C1 échange des données d'E/S avec tous les esclaves qui sont connectés au réseau. Les intervalles de temps du nombre d'esclaves configuré sont affectés à cette largeur de bande. Le maître C1 et une station esclave exécutent une fois l'échange de données d'E/S dans cet intervalle de temps.

Le maître C1 inscrit l'esclave avec lequel il n'a pas réussi à échanger des données d'E/S au moyen de cette largeur de bande sur la liste des nouvelles tentatives, comme cible pour une retransmission au moyen d'une autre largeur de bande utilisée pour une nouvelle tentative d'échange de données d'E/S.

4.3.2.1.2.4 Nouvelle tentative d'échange de données d'E/S

Il s'agit d'une largeur de bande au moyen de laquelle le maître C1 réessaye d'effectuer un échange des données d'E/S qui n'a pas pu s'effectuer au moyen de la largeur de bande d'échange de données d'E/S. Le maître C1 réexécute l'échange d'E/S avec l'esclave cible qui a été inscrit dans la largeur de bande d'échange de données d'E/S en vue d'une nouvelle tentative. Les intervalles de temps du nombre qui a été défini dans le maître C1 avant le début de la transmission cyclique sont alloués à cette largeur de bande. Le maître C1 effectue une nouvelle tentative en fonction de l'ordre d'inscription dans la liste de nouvelles tentatives, en allant jusqu'au nombre d'intervalles de temps alloués. La DLE cesse toute nouvelle tentative lorsqu'il utilise tous les intervalles de temps alloués, même si un esclave en attente d'une nouvelle tentative est inscrit dans la liste de nouvelles tentatives.

Le maître C1 réessaye d'effectuer l'échange de données d'E/S une fois pour chaque esclave inscrit. Si la nouvelle tentative échoue, le maître C1 n'effectue pas de nouvelle tentative pour ce même esclave.

4.3.2.1.2.5 Message C1

Il s'agit d'une largeur de bande pour la transmission de message (transmission de message C1), le maître C1 étant le client (initiateur), le maître C2 ou l'esclave étant le serveur (répondeur). Un intervalle de temps au maximum est alloué dans la largeur de bande qui est allouée à la nouvelle tentative d'échange de données d'E/S mentionnée ci-dessus, et la demande et la réponse sont émises une fois, respectivement. Du fait que l'une des largeurs de bande utilisées pour une nouvelle tentative d'échange de données d'E/S est allouée comme largeur de bande à cet effet, lorsque tous les intervalles de temps alloués pour nouvelle tentative sont utilisés, le message C1 ne peut pas être exécuté à l'intérieur du cycle de transmission.

4.3.2.1.3 Estimation de la durée de cycle

Le cycle de transmission du type à intervalle de largeur fixe T_{cycle} calculé est la somme des largeurs de bande décrites en 4.3.2.2, c'est-à-dire:

$$T_{cycle} = T_{sync} + T_{C2msg} + T_{io} + T_{retry} + T_{C1msg}$$

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

où

T_{sync} est la Bande de synchronisation;
T_{C2msg} es la Bande de message C2;
T_{io} est la Bande d'échange de données d'E/S;
T_{retry} est la Bande de nouvelle tentative d'échange de données d'E/S;
T_{C1msg} est la Bande de message C1.

Un multiple entier du nombre d'intervalles de temps est alloué aux largeurs de bande. On peut transformer la formule représentée ci-dessus en indiquant l'intervalle de temps par T_{slot}, le nombre de stations esclaves connectées au réseau par n, et le nombre de nouvelles tentatives par nr.

$$T_{cycle} = T_{\text{slot}} + T_{\text{slot}} + n \times T_{\text{slot}} + n_r \times T_{\text{slot}} + T_{\text{slot}}$$

$$= (n + n_r + 3) \times T_{\text{slot}}$$

On peut calculer l'intervalle de temps T_{slot} indiqué dans la formule suivante en indiquant la durée de transmission de trame (identique à la trame d'instruction, la trame de réplique, la trame de demande et la trame de réponse) par T_{tr} et l'espacement entre les trames par T_{gap}:

$$T_{slot} = \max(T_{tr\_c}(n) + T_{dly}(n) + T_{gap} + T_{tr\_r}(n) + T_{dly}(n) + T_{tr\_r} + T_{gap})$$
$$= 2 \times \max\left(T_{tr\_c}(n) + T_{dly}(n) + T_{gap}\right)$$

T_{tr_c} est la durée de transmission d'instruction du maître C1 à l'esclave n;

T_{tr_r} est la durée de transmission de réponse de l'esclave n au maître C1;

T_{gap} est l'espacement entre les trames;

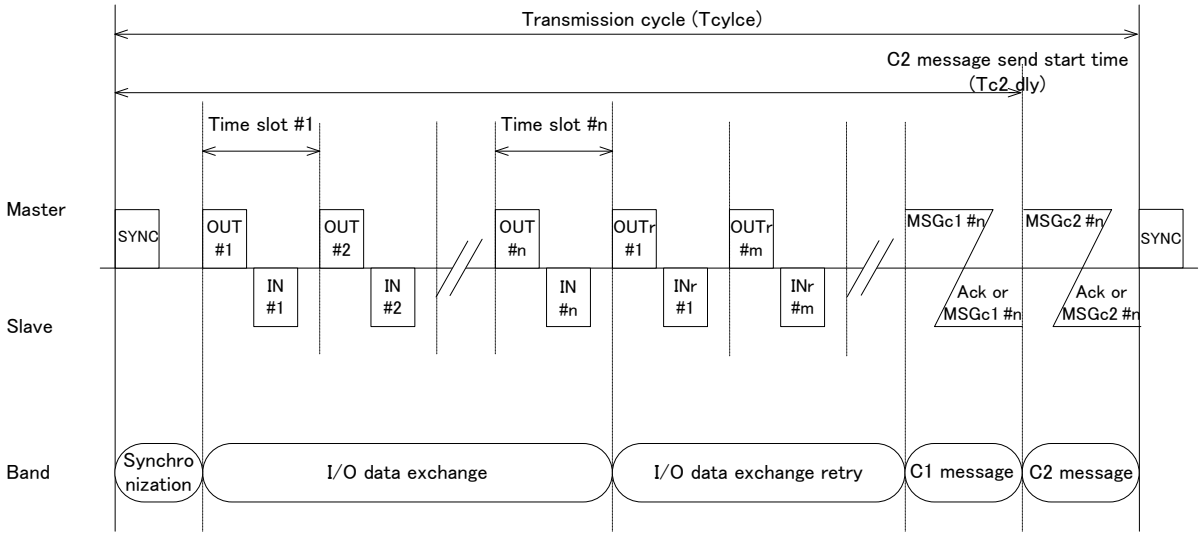
T_{dly} est le délai de transmission des trames entre le maître C1 et l'esclave n.

4.3.2.2 Type à intervalle de temps configurable

4.3.2.2.1 Présentation

La Figure 3 montre la séquence de transmission du type à intervalle de temps configurable. Dans ce type, la longueur des intervalles de temps qui sont alloués pour l'exécution des échanges de commande et de réponse, l'un après l'autre, entre le maître et l'esclave est différent pour chaque esclave. La DLE gère le temps résiduel du cycle de transmission en utilisant les intervalles de temps configurés pour chaque esclave par l'utilisateur du DLMS. Le détail de chacune des largeurs de bande de transmission est décrit dans les paragraphes suivants.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe



SYNC: Syncronous frame
OUT #n: Output data to slave #n
IN #n: Input data from slave #n
OUTr #m: Output data retry to slave #m
INr #m: Input data retry from slave #m
MSGc1 #n: C1 message to slave #n
MSGc2 #n: C2 message to slave #n

Légende

Anglais	Français
Transmission cycle (Tcycle)	Cycle de transmission (Tcycle)
C2 message send start time (Tc2-dly)	Instant de début d'envoi de message C2 (Tc2-dly)
Time slot #1	Intervalle de temps n° 1
Time slot #n	Intervalle de temps n° n
Ack or	Acc. ou
Master	Maître
Slave	Esclave
Band	Bande
Synchronization	Synchronisation
I/O data exchange	Échange de données d'E/S
I/O data exchange retry	Nouvelle tentative d'échange de données d'E/S
C1 message	Message C1
C2 message	Message C2
Synchronous frame	Trame synchrone
Output data to slave #n	Données de sortie vers l'esclave n° n
Input data from slave #n	Données d'entrée provenant de l'esclave n° n
Output data retry to slave #m	Nouvelle tentative de données de sortie vers l'esclave n° m
Input data retry from slave #m	Nouvelle tentative de données d'entrée provenant de l'esclave n° m
C1 message to slave #n	Message C1 vers l'esclave n° n
C2 message to slave #n	Message C2 vers l'esclave n° n

Figure 3 – Chronogramme de la communication cyclique de type à intervalle de temps configurable

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

4.3.2.2.2 Description détaillée de la phase de communication

4.3.2.2.2.1 Synchronisation

Voir 4.3.2.1.2.1.

4.3.2.2.2.2 Échange de données d'E/S

Il s'agit d'une largeur de bande au moyen de laquelle le maître C1 exécute l'échange de données d'E/S avec tous les esclaves connectés au réseau. La durée entre la fin de la largeur de bande de synchronisation en tête du cycle de transmission et le début du message C2 est allouée pour la largeur de bande qui regroupe cette largeur de bande, la largeur de bande suivante de nouvelle tentative d'échange de données d'E/S, et la largeur de bande de message C1. Le maître C1 et une station esclave exécutent une fois l'échange de données d'E/S dans cet intervalle de temps.

Le maître C1 inscrit l'esclave qui ne parvient pas à effectuer l'échange de données d'E/S dans cette largeur de bande sur la liste des nouvelles tentatives, comme cible pour la retransmission au moyen de la largeur de bande suivante utilisée pour une nouvelle tentative d'échange de données d'E/S.

4.3.2.2.2.3 Nouvelle tentative d'échange de données d'E/S

Cette bande est essentiellement la même que dans le cas de l'intervalle de temps de largeur fixe (voir 4.3.2.1.2.4). Le Paragraphe 4.3.2.2.2.3 décrit les différences.

La durée entre la fin de la largeur de bande de synchronisation en tête du cycle de transmission et le début du message C2 est allouée pour la largeur de bande qui regroupe la largeur de bande d'échange de données d'E/S, cette largeur de bande, et la largeur de bande de message C1. Le maître C1 exécute la nouvelle tentative pour l'esclave inscrit dans la liste des nouvelles tentatives dans l'ordre d'inscription, et efface l'inscription si la nouvelle tentative réussit. Au même moment, le maître C1 compare le temps nécessaire pour effectuer l'échange de données d'E/S avec l'esclave et le temps restant jusqu'à la fin de la largeur de bande (jusqu'au début du message C2), puis exécute la nouvelle tentative si le temps restant est supérieur. Si le temps restant est inférieur au temps nécessaire, le maître C1 met fin à cette largeur de bande.

Le maître C1 exécute la nouvelle tentative pour l'esclave inscrit dans la liste des nouvelles tentatives dans l'ordre d'inscription, et efface l'inscription si la nouvelle tentative réussit. Lorsque la largeur de bande se termine avant que les nouvelles tentatives n'aient été effectuées pour toutes les cibles à réessayer, le maître C1 cesse d'effectuer de nouvelles tentatives.

Si la nouvelle tentative exécutée pour un esclave inscrit ne réussit pas, le maître C1 réinscrit l'esclave à la fin de la liste des nouvelles tentatives. Le maître C1 continue d'effectuer de nouvelles tentatives pour le même esclave dans le temps restant de la largeur de bande. Lorsque les nouvelles tentatives de toutes les cibles de la liste des nouvelles tentatives ont été exécutées, le maître C1 récupère l'esclave dans la liste des nouvelles tentatives puis exécute à nouveau la nouvelle tentative. Le maître C1 met fin à la largeur de bande lorsque tous les esclaves ont été effacés de la liste des nouvelles tentatives.

4.3.2.2.2.4 Message C1

Il s'agit d'une largeur de bande pour la transmission de message (transmission de message C1), le maître C1 étant le client (station principale), le maître C2 ou l'esclave étant le serveur (station secondaire). Le temps restant entre la fin de la largeur de bande de nouvelle tentative d'échange de données d'E/S et le début de la transmission de message C2 est alloué à cette largeur de bande. Le maître C1 exécute la transmission de message C1 à l'intérieur de cette largeur de bande allouée. Le maître C1 peut répéter la transmission, constituée d'une demande et d'une réponse formant une paire, dans cette largeur de bande. Cependant, le

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

maître C1 ne peut pas exécuter la transmission de message C1 s'il ne reste pas suffisamment de temps pour une transmission lorsque la largeur de bande de nouvelle tentative d'échange de données d'E/S se termine.

4.3.2.2.2.5 Message C2

Il s'agit d'une largeur de bande pour la transmission de message (transmission de message C2), le maître C2 étant le client (station principale), le maître C1 ou l'esclave étant le serveur (station secondaire). Le temps entre le début du message C2 et la fin du cycle de transmission est alloué à cette largeur de bande. Le maître C2 exécute la transmission de message C2 à l'intérieur de cette largeur de bande allouée. Le maître C2 peut répéter la transmission, constituée d'une demande et d'une réponse formant une paire, dans la largeur de bande.

4.3.2.2.3 Estimation de la durée de cycle

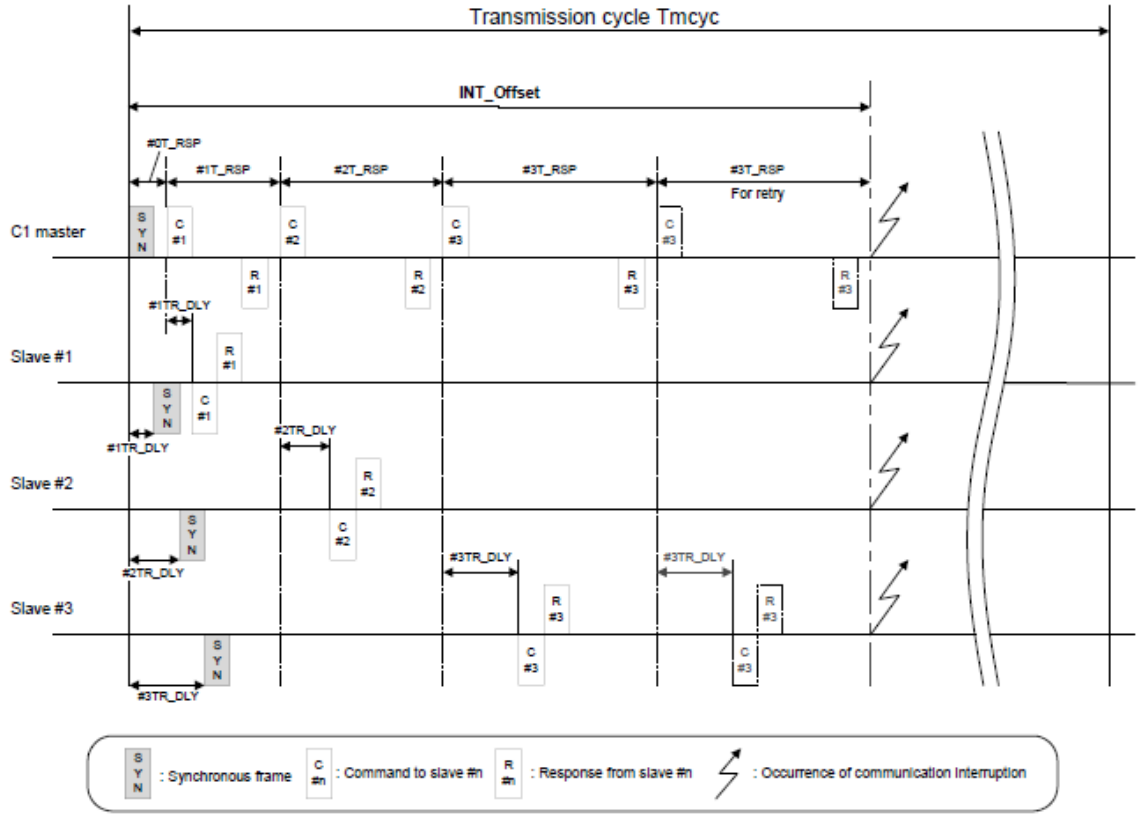
Le fonctionnement des esclaves multiples est synchronisé par la commande envoyée par le maître C1. Pour s'en assurer, on mesure le temps de retard de transmission de chaque esclave durant l'initialisation. Le temps de retard mesuré est conservé par le maître C1 et chaque esclave.

Selon le temps de retard de transmission mesuré, le maître C1 calcule le temps de surveillance de réponse et le temps de retard d'interruption pour chaque esclave, pour l'adapter à la chronologie des interruptions de communication du système.

Le temps de retard d'interruption est fourni avec la trame synchrone (trame SYN de la figure). Le maître C1 et chaque esclave génèrent une chronologie d'interruption de communication d'après le temps de retard d'interruption et le temps de retard de transmission conservé dans la station locale. Il en résulte que l'interruption de communication se produit au même moment sur l'ensemble du système.

Le système, en exécutant le traitement de réception de données simultanément au niveau de tous les esclaves en suivant la chronologie des événements cycliques, peut fonctionner de manière synchrone.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe



Légende

Anglais	Français
Transmission cycle Tmcyc	Cycle de transmission Tmcyc)
C1 master	Maître C1
Slave #1	Esclave n° 1
Slave #2	Esclave n° 2
Slave #3	Esclave n° 3
For retry	Pour nouvelle tentative
Synchronous frame	Trame synchrone
Command to slave #n	Commande vers l'esclave n° n
Response from slave #n	Réponse de l'esclave n° n
Occurrence of communication interruption	Occurrence des interruptions de communication.

Figure 4 – Schéma de l'occurrence des interruptions de communication

Le cycle de transmission du type à intervalle de temps variable T_{cycle} calculé est le regroupement des largeurs de bande décrites en 4.3.2.3.2, indiqué dans la formule suivante:

$$T_{cycle} = T_{sync} + T_{io} + T_{retry} + T_{C1msg} + T_{C2msg}$$

où

T_{sync} est la Bande de synchronisation;

T_{io} est la Bande d'échange de données d'E/S;

T_{retry} est la Bande de nouvelle tentative d'échange de données d'E/S;

T_{C1msg} est la Bande de message C1;

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

T_{C2msg} est la Bande de message C2.

Le procédé de calcul des largeurs de bande mentionnées ci-dessus est indiqué dans ce qui suit.

a) Bande de synchronisation

La largeur de bande de synchronisation T_{sync} est calculée de la façon suivante: où T_{tr_s} est le temps de transmission de la trame synchrone, et T_{gap} est l'espacement entre les trames:

$$T_{\text{sync}} = T_{\text{tr\_s}} + T_{\text{gap}}$$

b) Bande d'échange de données d'E/S

La largeur de bande d'échange de données d'E/S Tio est calculée de la façon suivante: où N est le nombre de connexions esclaves, $T_{tr\_c}(n)$ est le temps de transmission d'instruction du maître C1 vers l'esclave de numéro n, $T_{dly}(n)$ est le temps de retard de transmission entre le maître C1 et l'esclave de numéro n, et $Tgap$ est l'espacement entre les trames:

$$T_{io} = \sum_{n} \left\{ T_{tr\_c}(n) + T_{dly}(n) + T_{gap} + T_{tr\_r}(n) + T_{dly}(n) + T_{tr\_r} \right\}$$

$$= \sum_{n} \left\{ T_{tr\_c}(n) + T_{tr\_r}(n) + 2 \times T_{dly}(n) \right\} + 2 \times N \times T_{gap}$$

c) Bande de nouvelle tentative d'échange de données d'E/S

La largeur de bande d'échange de données d'E/S Tretry est calculée de la façon suivante: où N_r est le numéro de la nouvelle tentative, $T_{tr\_c}(r)$ le temps de transmission d'instruction du maître C1 vers l'esclave de numéro r, $T_{dly}(r)$ est le temps de retard de transmission de trame entre le maître C1 et l'esclave de numéro r, et Tgap est l'espacement entre les trames:

$$T_{retry} = \sum_{ir} \left\{ T_{tr\_c}(r) + T_{dly}(r) + T_{gap} + T_{tr\_r}(r) + T_{dly}(r) + T_{gap} \right\}$$

$$= \sum_{r} \left\{ T_{tr\_c}(r) + T_{tr\_r}(r) + 2 \times T_{dly}(r) \right\} + 2 \times N_r \times T_{gap}$$

d) Bande de message C1

La largeur de bande de message C1 T_{c1msg} est calculée de la façon suivante: $T_{tr\_c1c}(m_1)$ est le temps de transmission de demande entre la station principale (maître C1) et la station secondaire m_1 (m_1 est le numéro de station de l'esclave ou du maître C2 qui devient la station secondaire), $T_{tr\_c1r}(m_1)$ est le temps de transmission de la réponse de la station secondaire à la station principale, $T_{dly}(m_1)$ est le retard de transmission entre la station principale et la station secondaire, N_{c1msg} est le nombre de messages C1, et T_{gap} est l'espacement entre les trames:

$$T_{C1msg} = N_{c1msg} \times \left\{ T_{tr\_c1c}(m_1) + T_{dly}(m_1) + T_{gap} + T_{tr\_c1r}(m_1) + T_{dly}(m_1) + T_{gap} \right\}$$

$$= N_{c1msg} \times \left\{ T_{tr\_c1c}(m_1) + T_{tr\_c1r}(m_1) + 2 \times T_{dly}(m_1) + 2 \times T_{gap} \right\}$$

e) Bande de message C2

La largeur de bande de message C2 T_{c2msg} est calculée de la façon suivante: $T_{tr\_c2c}(m_2)$ est le temps de transmission de demande entre la station principale (maître C2) et la station secondaire m_2 (m_2 est le numéro de station de l'esclave ou du maître C1 qui devient la station secondaire), $T_{tr\_c2r}(m2)$ est le temps de transmission de la réponse de la station

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

secondaire à la station principale, $T_{dly}(m_2)$ est le retard de transmission entre la station principale et la station secondaire, N_{c2msg} est le nombre de messages C2, et T_{gap} est l'espacement entre les trames:

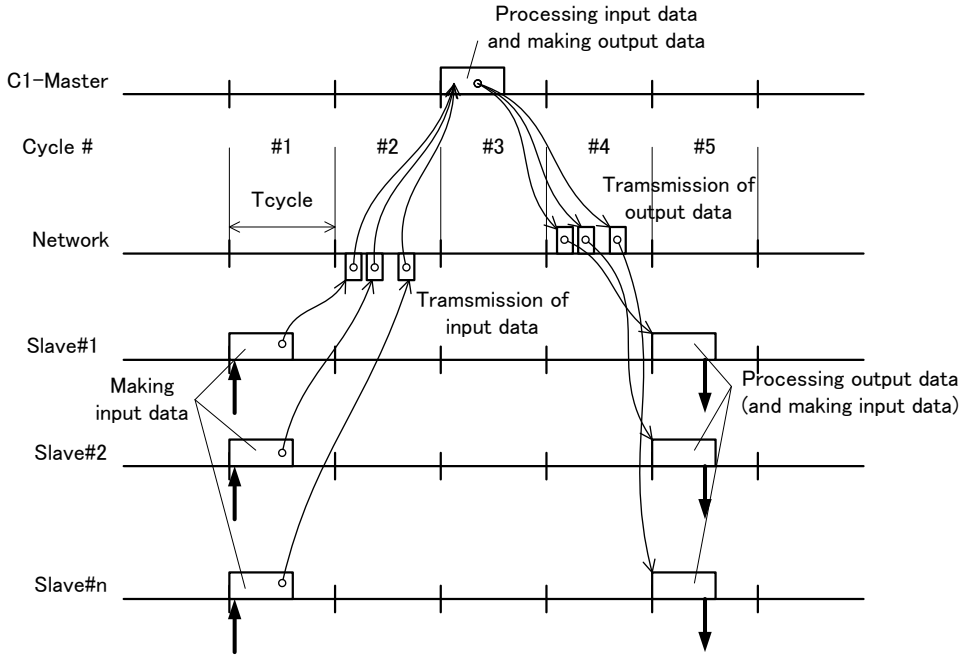
$$T_{c2msg} = N_{c2msg} \times \{T_{tr\_c2c}(m_2) + T_{dly}(m_2) + T_{gap} + T_{tr\_c2r}(m_2) + T_{dly}(m_2) + T_{gap}\}$$

$$= N_{c2msg} \times \{T_{tr\_c2c}(m_2) + T_{tr\_c2r}(m_2) + 2 \times T_{dly}(m_2) + 2 \times T_{gap}\}$$

4.3.2.3 Relation chronologique entre la transmission cyclique et le traitement des données

Le présent paragraphe explique la relation chronologique entre la transmission cyclique et le traitement des données à l'aide de la Figure 5. Dans le cycle n° 1, les esclaves verrouillent l'entrée et réalisent les données d'entrée à émettre. Les données d'entrée réalisées par chaque esclave sont envoyées au maître C1 dans le cycle n° 2. Elles sont reçues par le maître C1, mais ne sont pas traitées par lui à ce moment-là. Le maître C1 commence à les traiter à la fin du cycle n° 3. Le retard entre le verrouillage de l'entrée par l'esclave et l'instant prévu pour le traitement par le maître est donc de deux cycles de transmission.

De même, les données de sortie réalisées par le maître C1 au cycle n° 3 sont envoyées à tous les esclaves, l'un après l'autre, dans le cycle n° 4. Elles sont reçues par chacun des esclaves au cours du cycle n° 4, mais ne sont pas traitées par eux à ce moment-là. Les esclaves commencent à les traiter dans leur ensemble à la fin du cycle n° 5. Le retard entre la réalisation des données de sortie par le maître et l'instant prévu pour le traitement par les esclaves est donc de deux cycles de transmission, ce qui est la même chose que pour les données d'entrée.



NOTE Les données de sortie et les données d'entrée sont émises dans chaque cycle, mais ces schémas ont été omis dans cette figure pour plus de clarté. Le traitement des données effectué par le maître C1 et les esclaves à chaque cycle est également omis.

Légende

Anglais	Français
Processing input data and making output data	Traitement des données d'entrée et réalisation des données de sortie

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

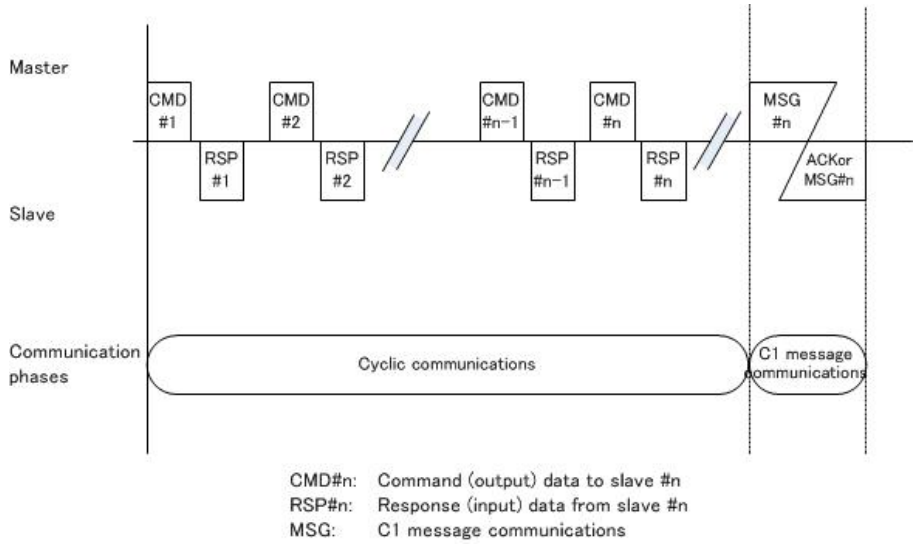
Anglais	Français
C1-Master	Maître C1
Cycle #	Cycle n°
Network	Réseau
Slave#1	Esclave n° 1
Slave#2	Esclave n° 2
Slave#n	Esclave n° n
Tcycle	Tcycle
Transmission of output data	Transmission de données de sortie
Transmission of input data	Transmission de données d'entrée
Making input data	Réalisation des données d'entrée
Processing output data (and making input data)	Traitement des données de sortie (et réalisation des données d'entrée)

Figure 5 – Relation chronologique entre la transmission cyclique et le traitement des données

4.3.3 Mode transmission acyclique

La transmission acyclique peut être utilisée dans un système qui ne nécessite pas de communication en temps réel ou d'échange de données cyclique. En mode de transmission acyclique, on peut exécuter la même séquence de transmission qu'en mode de transmission cyclique, mais sans que l'on fixe le cycle de transmission. Bien que la communication de messages C2 soit également possible, l'utilisateur DLS doit exécuter l'arbitrage de la chronologie de transmission. En mode de transmission acyclique, la longueur des données est fixée à 64 octets.

Étant donné que la transmission acyclique peut ne pas utiliser la trame synchrone, les esclaves exécutent le traitement des données de sortie envoyées par le maître et le traitement des données pour envoyer les données d'entrée à l'instant prévu pour elles. (Les esclaves ne fonctionnent pas simultanément.)



Légende

Anglais	Français
Master	Maître
Slave	Esclave
Communication phases	Phases de communication

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Anglais	Français
ACK or	ACC. ou
Command/response communication	Communication commande/réponse
C1 message communication	Communication message C1
Command (output) data to slave #n	Données de commande (sortie) vers l'esclave n° n
Response (input) data from slave #n	Données de réponse (entrée) provenant de l'esclave n° n

Figure 6 – Exemple de chronogramme de communication acyclique

4.4 Service pris en charge à partir de la PhL

4.4.1 Exigences générales

Il existe deux types de service Ph exigés par la DLE. L'un est défini dans la CEI 61158-2, Type 24, l'autre est défini à l'Article 6 de l'ISO/CEI 8802-3.

Les éléments d'interface suivants sont nécessaires aux deux PHY:

- Machine de transmission
- Machine de réception
- Sérialiseur
- Désérialiseur

4.4.2 Symboles DL

Les unités de données de l'interface PhL, présentes sur l'interface DLL-PhL, doivent être des symboles DL. Le symbole DL doit prendre l'une des valeurs suivantes:

a) ZÉRO, qui correspond à un "0" binaire;

b) UN, qui correspond à un "1" binaire.

4.4.3 Primitives du PhS prises en charge

Le PhS est censé fournir les deux catégories suivantes de primitives au protocole DL de type 24.

a) Primitives de service pour la transmission et la réception de trames en direction/provenance d'autres DLE homologues.

b) Primitives de service fournissant les informations dont la DLE locale a besoin pour exécuter les fonctions d'accès aux supports.

Les primitives attribuées au PhS sont groupées dans les deux catégories suivantes:

c) Transfert de données vers la DLE correspondante

   1) Demande PLS_DATA

   2) Indication PLS_DATA

d) Indication de la PhL locale

   3) Indication PLS_CARRIER

   4) Indication PLS_SIGNAL

   5) Indication PLS_DATA_VALID

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

4.5 Paramètres locaux, variables, compteurs, temporisateurs

4.5.1 Présentation

La présente spécification utilise les paramètres de demande d'utilisateur de DLS P(...) et les variables locales V(...) comme moyen pour clarifier l'effet de certaines actions et les conditions sous lesquelles ces actions sont valides, les temporisateurs locaux T(...) comme moyen de surveiller les actions du fournisseur de DLS distribué et de garantir une réponse DLE locale en cas d'absence de ces actions. Elle utilise également les files d'attente locales Q(...) comme moyen pour ordonner certaines activités, clarifier les effets de certaines actions et clarifier les conditions sous lesquelles ces activités sont valides.

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 spécifiée;

b) tous les temporisateurs doivent être initialisés à l'état inactif;

c) toutes les files d'attente doivent être initialisées à l'état vide.

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

4.5.2 Variables, paramètres, compteurs et temporisateurs supportant la fonction DLE

4.5.2.1 V(MA), P(MA)

Cette variable est utilisée par la CTC pour enregistrer l'adresse de station de cette station. Cette valeur doit être mise en œuvre par toutes les stations. Elle prend des valeurs comprises entre 1 et 2^{16}-1. Lorsque la station adopte des DLPDU de format court, les bits utiles de cette variable sont les 8 bits de poids faible.

4.5.2.2 V(Tcycle), P(Tcycle)

Cette variable est utilisée par la CTC pour enregistrer la période de transmission de la communication cyclique. Cette valeur doit être mise en œuvre par toutes les stations. Le temps indiqué par une valeur définie dépend de V(Tunit). Elle prend des valeurs comprises entre 1 000 et 64 000.

4.5.2.3 V(Nmax_slaves), P(Nmax_slaves)

Cette variable est utilisée par la CTC pour enregistrer le nombre de stations esclaves connectables. Elle prend des valeurs comprises entre 1 et 62.

4.5.2.4 V(Cyc_sel), P(Cyc_sel)

Cette variable est utilisée par la DLM pour enregistrer la sélection du mode de transmission, qui est cyclique ou acyclique. La liste des valeurs est indiquée dans le Tableau 5.

Tableau 5 – Liste des valeurs de la variable Cyc_sel

Valeur	Symbole	Description
0	CMode_Cyclic	Mode de transmission cyclique
1	CMode_Acyclic	Mode transmission acyclique

4.5.2.5 V(Nmax_dly_cnt), P(Nmax_dly_cnt)

Cette variable est utilisée par la DLM pour enregistrer le comptage de mesure maximal, qui limite le nombre d'exécutions de mesures de retard. Cette variable n'est mise en œuvre que par le maître C1 adoptant les intervalles de temps configurables. Sa valeur par défaut est de 2. Elle prend des valeurs comprises entre 2 et 31.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

4.5.2.6 V(IO_sz), P(IO_sz)

Cette variable est utilisée par la CTC pour conserver et désigner la taille des données des trames d'envoi et de réception, qui sont transférées dans la bande d'échange de données d'E/S. Elle prend des valeurs comprises entre 8 et 64. Lorsque la CTC adopte des intervalles de temps de largeur fixe, sa valeur doit être identique pour toutes les stations.

4.5.2.7 V(Pkt_sz), P(Pkt_sz)

Cette variable est utilisée par la machine de segmentation de messages pour enregistrer la taille des paquets de message de la communication cyclique. Elle prend des valeurs comprises entre 4 et 500. Lorsque la CTC adopte des intervalles de temps de largeur fixe, sa valeur doit être égale à V(IO_sz).

4.5.2.8 V(Nmax_retry), P(Nmax_retry)

Cette variable est utilisée par la CTC pour enregistrer le nombre maximal de nouvelles tentatives, qui limite le nombre de nouvelles tentatives dans la bande des nouvelles tentatives d'échange de données d'E/S. Cette variable n'est mise en œuvre que par le maître C1. Sa valeur par défaut est 0, qui signifie qu'aucune nouvelle tentative n'est autorisée. Elle prend des valeurs comprises entre 0 et 62.

4.5.2.9 V(Tslot), P(Tslot)

Cette variable est utilisée par la CTC pour conserver et désigner la valeur de la temporisation pour l'échange de données d'E/S avec chacune des stations. Cette valeur doit être mise en œuvre par le maître C1 et le maître C2. Elle prend des valeurs comprises entre 0 et une valeur ne dépassant pas V(Tcycle). Le temps indiqué par une valeur définie dépend de V(Tunit). Lorsque la CTC adopte des intervalles de temps de largeur fixe, sa valeur doit être identique pour toutes les stations.

4.5.2.10 V(Tunit), P(Tunit)

Cette variable est utilisée par la CTC pour enregistrer l'unité d'une valeur définie pour une variable liée au temps, lorsque la CTC adopte des intervalles de temps configurables. Le Tableau 6 indique les valeurs qu'elle prend. Cette variable ne doit être mise en œuvre que par une CTC qui adopte des intervalles de temps configurables. Une CTC qui adopte des intervalles de temps de largeur fixe ne doit pas utiliser cette variable; elle doit utiliser 250 ns comme unité de temps pour toutes les variables concernant le temps.

Tableau 6 – Liste des valeurs de la variable Tunit

Valeur	Définition	Tcycle
0	10 ns	31,25 µs à 500 µs (voir NOTE)
1	100 ns	De plus de 500 µs à 4 ms
2	1 µs	De plus de 4 ms à 64 ms
NOTE Valeur 0 par défaut.		

4.5.2.11 V(Tidly), P(Tidly)

Cette variable est utilisée par la CTC pour enregistrer la chronologie de l'événement, qui est émise périodiquement pour synchroniser l'utilisateur de DLS. Cette variable doit être mise en œuvre par toutes les stations qui adoptent des intervalles de temps configurables. La valeur est le temps de retard à partir du début de la transmission cyclique. Elle prend des valeurs comprises entre 0 et une valeur ne dépassant pas V(Tcycle). Le temps indiqué par une valeur définie dépend de V(Tunit).

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

4.5.2.12 V(Tc2_dly), P(Tc2_dly)

Cette variable est utilisée par la CTC pour enregistrer l'instant de la communication du message C2. Cette variable doit être mise en œuvre par un maître C1 et un maître C2 qui adoptent des intervalles de temps configurables. La valeur est le temps de retard à partir de la fin de la transmission cyclique. Elle prend des valeurs comprises entre 0 et une valeur ne dépassant pas V(Tcycle). Le temps indiqué par une valeur définie dépend de V(Tunit).

4.5.2.13 V(Tmsg), P(Tmsg)

Cette variable est utilisée par la CTC pour enregistrer la période de temps pour la communication de messages. Cette valeur doit être mise en œuvre par le maître C1 et le maître C2. Elle prend des valeurs comprises entre 0 et une valeur ne dépassant pas V(Tcycle). Le temps indiqué par une valeur définie dépend de V(Tunit). Lorsque la CTC adopte des intervalles de temps de largeur fixe, sa valeur doit être identique pour toutes les stations.

4.5.2.14 V(Twrpt)

Cette variable est utilisée par la SRC pour enregistrer la période de temps pour la mesure de retard de transmission. Cette valeur doit être mise en œuvre par le maître C2 et l'esclave. Elle prend des valeurs comprises entre V(Tslot) x 3 et une valeur ne dépassant pas 64 ms. La valeur par défaut est de 500 μs. Le temps indiqué par une valeur définie dépend de V(Tunit).

4.5.2.15 V(IO_MAP), P(IO_MAP)

Cette variable est utilisée par la DLM et la CTC pour enregistrer les informations des esclaves et du maître C2 qui doivent être connectés au réseau. Voir 5.3.2.2.13 de la CEI 61158-3-24 pour les détails.

4.5.2.16 V(Sts_STI), P(Sts_STI)

Cette variable est utilisée par la DLM pour conserver le statut de connexion des stations qui doivent être connectées au réseau. Voir 5.3.3.2.2.1 de la CEI 61158-3-24 pour les détails.

4.5.2.17 V(Sts_Err), P(Sts_Err)

Cette variable est utilisée par la DLM pour conserver le facteur d'erreur qui s'est produit dans la DLE.

4.5.2.18 V(Fc2msg)

Cette variable est utilisée par la CTC pour conserver la présence de la largeur de bande de communication de messages C2 dans le mode de transmission cyclique. Cette valeur doit être mise en œuvre par le maître C1 et le maître C2. Elle a pour valeur 0 ou 1. La valeur doit être fixée à 1 lorsque la largeur de bande de communication de messages C1 est affectée, à 0 sinon.

4.5.2.19 V(Nslave)

Cette variable est utilisée par la CTC pour conserver et désigner le numéro de l'esclave en train d'être traité dans l'intervalle de temps courant. Cette valeur doit être mise en œuvre par le maître C1 et le maître C2. Elle prend des valeurs comprises entre 0 et V(Nmax_slaves). La valeur est remise à zéro au début de chaque cycle de transmission cyclique. Elle est incrémentée lorsque l'échange de données d'E/S vers un esclave est exécuté.

4.5.2.20 V(Nretry)

Cette variable est utilisée par la CTC pour conserver et désigner l'élément de la liste des nouvelles tentatives en train d'être traitée dans l'intervalle de temps courant. La valeur est

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

remise à zéro au début de chaque cycle de transmission cyclique. Elle est incrémentée lorsque l'échange de données d'E/S vers un esclave est en défaut, et décrémentée lorsque la nouvelle tentative d'échange de données d'E/S est exécutée.

4.5.2.21 V(Nrest_slot)

Cette variable est utilisée par la CTC pour conserver et désigner le nombre d'intervalles de temps restants jusqu'à la fin de ce cycle. Elle prend des valeurs comprises entre 0 et V(Nmax_slave). Cette variable doit être mise en œuvre par le maître C1, qui adopte des intervalles de temps de largeur fixe. La valeur est décrémentée dans la plage allant de V(Nmax_retry) à 0.

4.5.2.22 V(Ndly_cnt)

Cette variable est utilisée par la DLM pour conserver et désigner le nombre de mesures de retard exécutées. Cette variable doit être mise en œuvre par le maître C1, qui adopte des intervalles de temps configurables. La valeur est remise à zéro au début de chaque cycle de transmission cyclique, et incrémentée lorsque la mesure de retard est exécutée. Elle prend des valeurs comprises entre 0 et V(Nmax__dly_cnt).

4.5.2.23 V(PDUType)

Cette variable est utilisée par la CTC et la DLM pour conserver et désigner la sélection de type de DLPDU, à savoir la DLPDU au format de base ou la DLPDU au format court. Lorsque la DLE adopte les deux types, elle peut ne pas mettre en œuvre cette variable. La liste des valeurs est indiquée dans le Tableau 7.

Tableau 7 – Liste des valeurs de la variable PDUType

Valeur	Symbole	Description
0	PDUBasic	DLPDU au format de base
1	PDUShort	DLPDU de format court

4.5.2.24 V(SlotType)

Cette variable est utilisée par la CTC et la DLM pour conserver et désigner la sélection de type d'intervalle de temps, à savoir le type de largeur fixe ou le type configurable. Lorsque la DLE adopte les deux types, elle peut ne pas mettre en œuvre cette variable. La liste des valeurs est indiquée dans le Tableau 8.

Tableau 8 – Liste des valeurs de la variable SlotType

Valeur	Symbole	Description
0	TSFixed	Intervalle de temps de largeur fixe
1	TSConfig	Intervalle de temps configurable

4.5.2.25 V(Nms_1), V(Nms_2)

Cette variable est utilisée par la CTC pour conserver le nombre de segmentations à envoyer dans la communication de messages utilisant la DPLDU au format de base. Les suffixes "_1" et "_2" montrent la largeur de bande de la communication de messages et désignent respectivement la communication de messages C1 et la communication de messages C2. Cette valeur doit être mise en œuvre par toutes les stations qui exécutent la communication de messages. La valeur est remise à zéro au premier segment de chaque message. Elle est incrémentée d'une unité chaque fois que la CTC reçoit un accusé de réception d'un segment, normalement de la part d'une station homologue. Elle prend des valeurs comprises entre 0 et 127.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

4.5.2.26 V(Nmr_1), V(Nmr_2)

Cette variable est utilisée par la CTC pour conserver le nombre de segments à recevoir dans la communication de messages utilisant la DPLDU au format de base. Les suffixes "_1" et "_2" montrent la largeur de bande de la communication de messages et désignent respectivement la communication de messages C1 et la communication de messages C2. Cette valeur doit être mise en œuvre par toutes les stations qui exécutent la communication de messages. La valeur est remise à zéro au premier segment de chaque message. Elle est incrémentée lorsqu'un segment est reçu de la part d'une station homologue. Elle prend des valeurs comprises entre 0 et 127.

4.5.2.27 V(Fmp_1), V(Fmp_2)

Cette variable est utilisée par la CTC pour conserver et désigner l'indicateur de demande de sondage dans la communication de messages utilisant la DPLDU au format de base. Les suffixes "_1" et "_2" montrent la largeur de bande de la communication de messages et désignent respectivement la communication de messages C1 et la communication de messages C2. Cette valeur doit être mise en œuvre par toutes les stations qui exécutent la communication de messages. Elle prend la valeur 0 ou 1. La valeur 1 correspond à la demande de sondage, la valeur 0 (par défaut) à l'autre.

4.5.2.28 V(Fmf_1), V(Fmf_2)

Cette variable est utilisée par la CTC pour conserver et désigner l'indicateur du dernier segment dans la communication de messages utilisant la DPLDU au format de base. Les suffixes "_1" et "_2" montrent la largeur de bande de la communication de messages et désignent respectivement la communication de messages C1 et la communication de messages C2. Cette valeur doit être mise en œuvre par toutes les stations qui exécutent la communication de messages. Elle prend la valeur 0 ou 1. La valeur est remise à zéro au premier segment de chaque message et mise à un lorsque le dernier segment du message est envoyé ou reçu.

4.5.2.29 V(Ten)

Cette variable est utilisée par la DLM pour conserver l'horodatage de la fin de la mesure de retard lorsque la DLE adopte les intervalles de temps configurables. La DLE ne doit mettre en œuvre cette variable que si la DLE adopte des intervalles de temps configurables.

4.5.2.30 V(Tdly)

Cette variable est utilisée par la DLM pour conserver le retard de transmission mesuré dans l'état CompDly de la DLM lorsque la DLE adopte les intervalles de temps configurables. La DLE ne doit mettre en œuvre cette variable que si la DLE adopte des intervalles de temps configurables.

4.5.2.31 V(Tmax_dly)

Cette variable est utilisée par la DLM pour conserver le retard de transmission mesuré dans l'état CompDly de la DLM lorsque la DLE adopte les intervalles de temps configurables. La DLE ne doit mettre en œuvre cette variable que si la DLE adopte des intervalles de temps configurables.

4.5.2.32 V(Tst)

Cette variable est utilisée par la DLM pour conserver l'horodatage du début de mesure de retard lorsque la DLE adopte les intervalles de temps configurables. La DLE ne doit mettre en œuvre cette variable que si la DLE adopte des intervalles de temps configurables.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

4.5.2.33 T(Tcycle)

T(Tcycle) est utilisée par la CTC pour mesurer la période de transmission cyclique. La valeur est décrémentée dans la plage allant de V(Tcycle) à 0.

4.5.2.34 T(Tslot)

T(Tslot) est utilisée par la CTC pour mesurer le temps écoulé depuis le dernier envoi d'une trame. La valeur est décrémentée dans la plage allant de V(Tslot) à 0.

4.5.2.35 T(Tmsg)

T(Tmsg) est utilisée par la CTC pour mesurer la période de temps de la bande de communication de messages. La valeur est décrémentée dans la plage allant de V(Tmsg) à 0.

4.5.2.36 T(Twrpt)

T(Twrpt) est utilisée par la SRC pour surveiller la fonction de répétition. La valeur est décrémentée dans la plage allant de V(Twrpt) à 0.

4.5.2.37 Q(MSGc1s), Q(MSGc2s)

Le tampon de file d'attente est mis en œuvre au niveau de l'interface entre la CTC et la MSM pour transférer le message à envoyer. La CTC met en file d'attente le message à envoyer, puis la MSM l'extrait de la file d'attente pour construire la DLPDU et l'envoyer.

4.5.2.38 Q(MSGc1r), Q(MSGc2r)

Le tampon de file d'attente est mis en œuvre au niveau de l'interface entre la CTC et la machine protocolaire de segmentation de messages pour transférer le message reçu. La MSM construit le message à partir de la DLPDU reçue et le met en file d'attente, puis la CTC l'extrait de la file d'attente pour le transférer vers l'utilisateur de DLS.

5 Structure de la DLPDU

5.1 Présentation

5.1.1 Syntaxe de transfert des séquences de bits

Pour la transmission sur des couches DL de type 24, une séquence de bits est réordonnée en une séquence d'octets. La notation hexadécimale est utilisée pour les octets de la façon spécifiée dans l'ISO/CEI 9899. Soit b = b_0...b_{n-1}, une séquence de bits. Soit k un entier naturel tel que 8(k - 1) <= n < 8 k. La séquence b est convertie en k octets assemblés comme indiqué au Tableau 9. Les bits b_i, i > n, de l'octet portant le numéro le plus élevé doivent être ignorés.

Tableau 9 – Syntaxe de transfert des séquences de bits

Numéro d'octet	1	2	k
	b_7...b_0	b_{15}..b_8	b_{8k-1}..b_{8k-8}

Lorsque la DLE est mise en œuvre sur la couche PHY définie dans la CEI 61158-2, Type 24, l'octet 1 est émis en premier, l'octet k en dernier. Par conséquent, la séquence de bits est transférée comme suit sur le réseau:

b_0, b_1, ..., b_7, b_8, ..., b_{15}, ...

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

5.1.2 Codages des types de données

Les données du type de données de base Unsigned*n* sont des valeurs entières naturelles. La plage de valeurs est comprise entre 0 et 2^n-1. Les données sont représentées sous forme de séquences de bits de longueur n. La séquence de bits

$b = b_0 ... b_{n-1}$

prend la valeur

$$\text{Unsigned}(b) = b_{n-1} \times 2^{n-1} + ... + b_1 \times 2^1 + b_0 \times 2^0$$

La séquence de bits commence à gauche par l'octet de poids le plus faible.

EXEMPLE La valeur 266 = 0x10A de type de données Unsigned16 est transférée sous la forme de deux octets: d'abord 0x0A, puis 0x01.

Tableau 10 – Ordre des bits

Numéro d'octet	1	2	3	4	5	6	7	8
Unsigned8	$b_7..b_0$							
Unsigned16	$b_7..b_0$	$b_{15}..b_8$						
Unsigned32	$b_7..b_0$	$b_{15}..b_8$	$b_{23}..b_{16}$	$b_{31}..b_{24}$				
Unsigned64	$b_7..b_0$	$b_{15}..b_8$	$b_{23}..b_{16}$	$b_{31}..b_{24}$	$b_{39}..b_{32}$	$b_{47}..b_{40}$	$b_{55}..b_{48}$	$b_{63}..b_{56}$

Le transfert des types de données Unsigned*n* s'effectue comme indiqué dans le Tableau 10. Les types de données Unsigned tels que Unsigned1 à Unsigned7 et Unsigned 9 à Unsigned15 sont également utilisés. Dans ce cas, l'élément suivant commence à la première position de bit libre indiquée en 5.1.1.

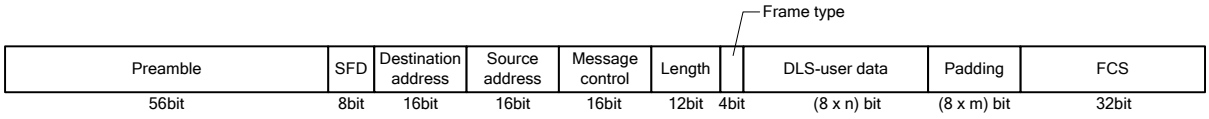
5.1.3 Format de trame

Il existe deux types de format de trame. L'un est le format de base, l'autre le format court défini par l'ISO/CEI 13239:2002 (HDLC).

5.2 Structure de la DLPDU au format de base

5.2.1 Généralités

La Figure 7 montre la structure du format de base des trames.



Légende

Anglais	Français
Preamble	Préambule
SFD	SFD
Destination address	Adresse destination
Source address	Adresse source

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Anglais	Français
Message control	Contrôle de message
Length	Longueur
Frame type	Type de trame
DLS-user data	Données d'utilisateur de DLS
Padding	Remplissage
FCS	FCS
56bit	56 bits
8bit	8 bits
16bit	16 bits
12bit	12 bits
4bit	4 bits
(8 x n) bit	(8 x n) bits
(8 x m) bit	(8 x m) bits
32bit	32 bits

Figure 7 – Structure de la DLPDU au format de base

5.2.1.1 Préambule

Le champ de préambule est identique à celui de l'Article 3 de l'ISO/CEI 8802-3:2000. Le champ de préambule est un champ de 56 bits utilisé pour permettre aux circuits de la partie de signalisation physique d'atteindre leur synchronisation de régime établi avec la chronologie des trames de réception.

Le motif du préambule est:

“10101010 10101010 10101010 10101010 10101010 10101010 10101010.“

5.2.1.2 Délimiteur de début de trame (SFD)

Le délimiteur de début de trame (SFD) est identique à celui de l'Article 3 de l'ISO/CEI 8802-3:2000. Le champ SFD est la séquence de motif binaire "10101011". Il suit immédiatement le motif du préambule et indique le début de la trame.

5.2.1.3 Adresse de destination (DA)

L'adresse de destination doit contenir l'adresse de nœud de la DLE de destination.

Les 8 bits de poids fort d'une adresse de 16 bits sont utilisés comme adresse étendue, les 8 bits de poids faible comme adresse de station (voir Tableau 11).

Pour les adresses de station comme pour l'adresse d'extension, les adresses utilisables sont spécifiées dans les Tableau 12 et Tableau 13.

Tableau 11 – Format d'adresse de source et de destination

Numéro d'octet	Taille	Contenu
1	Unsigned8	Adresse de station
2	Unsigned8	Adresse étendue

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Tableau 12 – Adresse de station

Adresse de station	Contenu
0x00	Réservé
0x01	Maître C1
0x02	Maître C2
0x03 à 0xEF	Esclave, passerelle
0xF0 à 0xFE	Réservé
0xFF	Adresse de diffusion

Tableau 13 – Adresse étendue

Adresse étendue	Contenu
0x00 à 0xFEh	Adresse d'appareil pour le nœud qui possède plusieurs appareils, ou adresse distante pour le nœud de passerelle
0xFF	Adresse de diffusion[a]
[a] Seules les trames synchrones peuvent utiliser une adresse de diffusion.	

5.2.1.4 Adresse source (SA)

L'adresse source doit contenir l'adresse de nœud de la DLE source. Voir en 5.2.1.3 son format et les valeurs qu'elle peut prendre.

5.2.1.5 Contrôle de message

Le champ Contrôle de message sert à envoyer des données avec le service d'accusé de réception. Ce champ n'a d'effet que si le type de trame qui suit ce champ est Trame de message. Ce champ doit être à zéro sauf dans le cas de la Trame de message.

Ce champ possède deux formats. L'un est le format transfert d'informations, l'autre le format superviseur (voir le Tableau 14 et le Tableau 15). Chaque champ doit avoir la même fonction que le champ de même nom de l'ISO/CEI 13239:2000 (HDLC).

NOTE Dans la HDLC, l'ordre de transmission des bits est l'inverse de celui de la présente spécification.

Tableau 14 – Format du champ Contrôle de message (Format de transfert d'informations)

Numéro d'octet	Taille	Symbole	Contenu
1	Unsigned7	N(R)	Transmission du numéro de séquence de réception
	Unsigned1	P/F	Bit de sondage – transmissions principales Bit final – transmission secondaire
2	Unsigned7	N(S)	Transmission du numéro de séquence d'envoi
	Unsigned1	-	Réservé (0)[a]
[a] Ce bit doit être à zéro.			

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Tableau 15 – Format du champ Contrôle de message (Format superviseur)

Numéro d'octet	Taille	Symbole	Contenu
1	Unsigned7	N(R)	Transmission du numéro de séquence de réception
	Unsigned1	-	Réservé (1)[a]
2	Unsigned4	-	Réservé (0)[b]
	Unsigned2	S	Bits de fonction de superviseur
	Unsigned1	-	Réservé (0)[b]
	Unsigned1	-	Réservé (1)[a]
a Ce bit doit être à un. b Ces bits doivent être à zéro.			

Tableau 16 – Liste des bits de fonction de superviseur

Valeur	Symbole	Description	Note
0	RR	Commande Réception prête (RR) ou réponse RR	
1	REJ	Commande Rejet (REJ) ou réponse REJ	
2	RNR	Réponse Réception non prête (RNR)	
3	–	Réservé	

5.2.1.6 Champ Type et longueur

Le champ Type de trame et longueur de données comprend une zone de 4 bits de poids fort servant à définir le type de trame et une zone de 12 bits de poids faible servant à définir la longueur des données (voir Tableau 17).

Le type de trame sert à identifier le contenu d'une trame ainsi qu'à identifier le type du protocole utilisé pour la communication de messages. Le Tableau 18 donne la définition des types de trame. La longueur de données indique la taille des données stockées dans la trame en ce qui concerne le nombre d'octets.

Tableau 17 – Format du champ Type de trame et longueur de données

Numéro d'octet	Taille	Contenu
1	Unsigned12	Longueur de données (8 bits de poids faible)
2		Longueur de données (4 bits de poids fort)
	Unsigned4	Type de trame

Tableau 18 – Liste des types de trame

Valeur	Symbole	Description
0	–	Réservé
1	FT_SYNC	Trame synchrone
2	FT_IO	Trame de données de sortie et de données d'entrée
3	FT_DLST	Trame de début de mesure de retard
4	FT_DLMS	Trame de mesure de retard
5	FT_MTKN	Trame de jeton de message
6	FT_STS	Trame de statut

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Valeur	Symbole	Description
7	FT_CINF	Trame d'information cyclique
8 à 11	–	Réservé
12	FT_MSG	Trame de message
13 à 15	–	Réservé

5.2.1.7 Champ Données d'utilisateur de DLS

Le format des données dépend du type de trame. Avec les trames de message, si les données d'application ne peuvent pas tenir dans une seule trame, on peut diviser les données et les envoyer en plusieurs trames au moyen du contrôle de message.

5.2.1.8 Champ Séquence de vérification de champ (FCS)

La séquence de vérification de trame (FCS) utilise le CRC-CCITT 32 bits. La FCS est calculée dans la plage comprise entre l'adresse de destination et la fin des données, à l'exception de la FCS elle-même.

5.2.2 Trame synchrone

Le maître C1 utilise cette trame pour synchroniser les esclaves et le maître C2. Seul le maître C1 peut envoyer cette trame. Le maître C1 doit définir l'adresse de station en tant qu'adresse de diffusion (0xFF), et les esclaves et le maître C2 doivent recevoir cette trame.

Lorsque l'esclave et le maître C2 reçoivent cette trame, ils doivent rafraîchir l'horloge locale avec le temps calculé par addition du retard de transmission mesuré à l'avance (indiqué au moyen de la trame de mesure de retard) et du temps courant stocké dans cette trame.

Le Tableau 19 et le Tableau 20 donnent le détail du format des données de cette trame.

Tableau 19 – Format des données de la trame synchrone

Numéro d'octet	Taille	Contenu
1 à 4	Unsigned32	Horodatage
5 à 6	Unsigned16	Temps de retard d'événement cyclique
7 à 8	Unsigned16	Réservé

Tableau 20 – Liste des champs de la trame synchrone

Champ	Contenu	Valeurs maximale et minimale	Valeur recommandée	Note
Horodatage	L'horodatage du maître C1 dans la trame est envoyé	0 à 2^{32}-1	–	L'unité est définie par le paramétrage de la variable V(Tunit).
Temps de retard d'événement cyclique	Temps de retard entre le temps courant stocké dans cette trame et l'instant où l'événement cyclique est indiqué	0 à 2^{16}-1	0	L'unité est définie par le paramétrage de la variable V(Tunit).

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

5.2.3 Trame de données de sortie ou de données d'entrée

Le maître C1 utilise cette trame pour les données de sortie à envoyer aux esclaves, et les esclaves utilisent cette trame pour les données d'entrée à envoyer au maître C1, dans la largeur de bande d'échange de données d'E/S du mode de transmission cyclique. Le maître C2 ne peut que recevoir cette trame; il ne doit pas l'envoyer.

Au moins l'adresse de destination ou l'adresse de source de cette trame doit être le maître C1, car cette trame est échangée entre le maître C1 et l'esclave. La longueur des données ne doit pas être modifiée durant le fonctionnement normal.

Le Tableau 21 et le Tableau 22 donnent le format des données de cette trame.

Tableau 21 – Format des données de la trame de données de sortie ou de données d'entrée

Numéro d'octet	Taille	Contenu
1 à n	Unsigned8n	Données de sortie ou données d'entrée
(n+1) à (n+m)	Unsigned8m	Remplissage

Tableau 22 – Liste des champs de la trame de données de sortie ou de données d'entrée

Champ	Contenu	Valeurs maximale et minimale	Valeur recommandée	Note
Données de sortie ou données d'entrée	Données de sortie transitant du maître C1 vers l'esclave ou données d'entrée transitant de l'esclave vers le maître C1	De 0 à la valeur maximale qui peut être représentée avec la longueur d'octet de la longueur de données spécifiée	–	
Remplissage	Zone ajustée pour faire en sorte que la longueur de données soit un multiple de 32 bits	Indifférentes	0	

5.2.4 Trame de début de mesure de retard

Le maître C1 utilise cette trame pour spécifier la station cible de la mesure de retard, afin de mesurer le retard de transmission entre le maître C1 et chaque esclave ou le maître C2. Seul le maître C1 émet cette trame. Lorsque la station reçoit cette trame, elle renvoie la trame de réception le nombre de fois spécifié dans cette trame.

Le Tableau 23 et le Tableau 24 donnent le détail du format des données de cette trame.

Tableau 23 – Format des données de la trame de début de mesure de retard

Numéro d'octet	Taille	Contenu
1 à 2	Unsigned16	Numéro de mesure
3 à 4	Unsigned16	Réservé

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Tableau 24 – Liste des champs de la trame de début de mesure de retard

Champ	Contenu	Valeurs maximale et minimale	Valeur recommandée	Note
Numéro de mesure	Nombre de fois pour l'envoi de la trame de mesure de retard de transmission	1 à 32	1	

5.2.5 Trame de mesure de retard

Le maître C1 utilise cette trame pour mesurer le retard de transmission et informer les esclaves et le maître C2 du résultat. Seul le maître C1 émet cette trame. Le maître C1 doit envoyer cette trame à la station à laquelle la trame de début de mesure de retard est envoyée, puis envoyer cette trame le nombre de fois spécifié dans la trame de début de mesure de retard. Le maître C1 doit informer du résultat de la mesure en utilisant cette trame.

L'esclave et le maître C2 qui ont reçu la trame de début de mesure de retard doivent recevoir cette trame et la renvoyer. Ils doivent cesser de renvoyer la trame de réception lorsqu'ils ont reçu cette trame le nombre de fois spécifié dans la trame de début de mesure de retard.

Le Tableau 25 et le Tableau 26 donnent le détail du format des données de cette trame.

Tableau 25 – Format des données de la trame de mesure de retard

Numéro d'octet	Taille	Contenu
1 à 4	Unsigned32	Horodatage
5 à 6	Unsigned16	Retard de transmission
7 à 8	Unsigned16	Réservé

Tableau 26 – Liste des champs de la trame de mesure de retard

Champ	Contenu	Valeurs maximale et minimale	Valeur recommandée	Note
Retard de transmission	Temps écoulé entre l'envoi d'une trame par le maître C1 et sa réception par l'esclave ou le maître C2.	0 à 2^{16}-1	–	L'unité est définie par le paramétrage de l'unité de temps. L'unité par défaut est de 10 ns.

5.2.6 Trame de jeton de message

Le maître C1 envoie cette trame pour informer le maître C2 qu'il dispose de la permission de lancer la communication de messages C2 avant l'instant de début de la communication de messages C2. Le maître C1 ne doit envoyer cette trame que lorsqu'il est en train d'échanger les données de commande/données de réponse avec tous les esclaves et que la communication de messages C1 se termine avant l'instant de début de la communication de messages C2. Cependant, le maître C1 ne doit pas envoyer cette trame si le temps séparant l'heure courante donnée par l'horloge locale et l'instant de début de la communication de messages C2 est inférieur au retard de transmission entre le maître C1 et le maître C2.

Le maître C2 qui reçoit cette trame peut lancer la communication de messages C2, sauf si l'instant de début de la communication de messages C2 est échu.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Seul le maître C1 peut envoyer cette trame. L'adresse de destination et l'adresse de source sont fixes car seul le maître C2 peut recevoir cette trame. Cette trame ne contient pas de données.

5.2.7 Trame de statut

Le maître C1 utilise cette trame pour demander le statut des esclaves et du maître C2. La station qui reçoit cette trame du maître C1 doit envoyer cette trame pour indiquer le statut courant. Les esclaves et le maître C2 doivent également renvoyer cette trame lorsqu'ils reçoivent une trame d'information de cycle dont l'adresse de destination est leur propre adresse. L'esclave et le maître C2 peuvent renvoyer cette trame pour demander au maître C1 d'exécuter la mesure de retard de transmission, même lorsqu'ils reçoivent du maître C1 une trame autre que la trame synchrone.

La destination spécifiée pour cette trame ne doit être ni une adresse de diffusion, ni une adresse de multidiffusion.

Les Tableau 27 à Tableau 30 donnent le détail du format des données de cette trame.

Tableau 27 – Format des données de la trame de statut

Numéro d'octet	Taille	Contenu
1 à 2	Unsigned16	Statut
3 à 4	Unsigned16	Statut de répétition

Tableau 28 – Liste des champs de la trame de statut

Champ	Contenu	Valeurs maximale et minimale	Valeur recommandée	Note
Statut	Valeur courante du statut de la DLE	(Voir Tableau 29)	–	Rafraîchi par l'esclave ou le maître C2
Statut de répétition	Valeur courante du statut de répétition	(Voir Tableau 30)	–	Rafraîchi par l'esclave ou le maître C2

Tableau 29 – Liste des statuts de DLE

Code de statut	Description
0x0000	Station inexistante
0x0023	Adresse de station dupliquée
0x0024	Attente de la demande de mesure de retard de la part de l'utilisateur de DLS
0x0025	Attente de l'indication de début de mesure de retard de la part du maître C1
0x0026	Mesure du retard de transmission
0x0027	Attente de la trame d'information cyclique de la part du maître C1
0x0050	Fonctionnement en mode de transmission cyclique
0x0060	Fonctionnement en mode de transmission acyclique

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Tableau 30 – Liste des statuts de répétition

Statut d'envoi			
bit	**Symbole**	**Description**	**Initial**
0	–	Réservé	0
1	MII_RXTXE	Demande d'envoi détectée pendant la réception de données de la PhL	0
2	MII_TXRXE	Réception de données sur la PhL, détectée pendant l'envoi de données sur la PhL	0
3	MII_RXRXE	Réception de données sur la PhL, détectée pendant la réception de données de la PhL	0
4 à 15	–	Réservé	0

5.2.8 Trame d'information de cycle

Le maître C1 utilise cette trame pour informer l'esclave et le maître C2 du mode de transmission. Seul le maître C1 peut envoyer cette trame.

Le maître C1 peut diffuser cette trame ou l'envoyer aux esclaves ou au maître C2 individuellement. Les esclaves et le maître C2 doivent renvoyer une trame de statut lorsqu'ils reçoivent cette trame et que celle-ci contient une adresse de destination qui est leur propre adresse.

Le Tableau 31 et le Tableau 32 donnent le détail du format des données de cette trame.

Tableau 31 – Format des données de la trame de mesure de retard

Numéro d'octet	Taille	Contenu
1 à 2	Unsigned16	Cycle de transmission
3 à 4	Unsigned16	Retard de message C2
5 à 6	Unsigned16	Retard maximal
7	Unsigned8	Mode de communication
8	Unsigned8	Unité de temps

Tableau 32 – Liste des champs de la trame d'information de cycle

Champ	Contenu	Valeurs maximale et minimale	Note
Cycle de transmission	Cycle de transmission du mode de transmission cyclique	3 125 à 64 000	a)
Retard de message C2	Temps de retard entre l'heure courante stockée dans la trame synchrone et l'instant de début de la communication de messages C2	0 à 2^{16}-1	
Retard maximal	Valeur maximale des retards de transmission mesurés par le maître C1.	0 à 2^{16}-1	a)
Mode de communication	Sélection du mode de transmission à exécuter après l'initialisation	0: Cyclique 1: Acyclique	
Unité de temps	Code de sélection de	0: 10 ns	b)

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Champ	Contenu	Valeurs maximale et minimale	Note
	l'unité de temps	1: 100 ns 2: 1 µs	
a) L'unité est définie par la valeur du champ d'unité de temps de cette trame. b) Si le cycle de transmission Tcycle est 31,25 µs <= T_MCYC <= 500 µs, régler sur 10 ns. Si 500 µs < T_MCYC <= 4 ms, régler sur 100 ns. Si 4 ms < T_MCYC <= 64 ms, régler sur 1 µs.			

5.2.9 Trame de message

Cette trame est utilisée pour la transmission de message. Toutes les stations peuvent envoyer cette trame. Si la taille de la DLSDU à envoyer dépasse ce que peut contenir une trame, la DLSDU doit être divisée au moyen du champ Contrôle de message de cette trame.

Le Tableau 33 et le Tableau 34 donnent le détail du format des données de cette trame.

Tableau 33 – Format des données de la trame de message

Numéro d'octet	Taille	Contenu
1 à n	Unsigned8n	Données de message
(n+1) à (n+m)	Unsigned8m	Remplissage

Tableau 34 – Liste des champs de la trame de message

Champ	Contenu	Valeurs maximale et minimale	Valeur recommandée	Note
Données de message	Données arbitraires, sauf les données de sortie ou les données d'entrée	De 0 à la valeur maximale qui peut être représentée avec la longueur d'octet de la longueur de données spécifiée	–	
Remplissage	Zone ajustée pour faire en sorte que la longueur de données soit un multiple de 32 bits	Indifférentes	0	

5.3 Structure de la DLPDU de format court

5.3.1 Généralités

La DLPDU de format court est présentée sur la Figure 8. La séquence de transmission doit aller de gauche à droite, comme indiqué sur la Figure 8, c'est-à-dire d'abord l'adresse de la station, puis les données d'utilisateur de DLS, puis le CRC.

Preamble	Start flag	Station address	Control	DLS-user data	CRC	End flag
16 bit	8 bit	8 bit	8 bit	(8 x n) bit	16 bit	8 bit

Légende

Anglais	Français
Preamble	Préambule
Start flag	Indicateur de début

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Anglais	Français
Station address	Adresse station
Control	Contrôle
DLS-user data	Données d'utilisateur de DLS
CRC	CRC
End flag	Indicateur de fin
16 bit	16 bits
8 bit	8 bits
(8 x n) bit	(8 x n) bits

Figure 8 – Structure de la DLPDU de format court

5.3.1.1 Préambule

Le champ de préambule est identique à celui de l'ISO/CEI 13239:2002 (HDLC). Le champ de préambule est un champ de 16 bits utilisé pour permettre aux circuits de la partie de signalisation physique d'atteindre leur synchronisation de régime établi avec la chronologie des trames de réception.

Le motif du préambule est:

“10101010 10101010.”

5.3.1.2 Indicateur de début

L'indicateur de début est identique à celui de l'ISO/CEI 13239:2002 (HDLC). Ce champ est la séquence du motif binaire "10101011". Il suit immédiatement le motif du préambule et indique le début de la trame.

En raison de l'insertion de bits de remplissage, on interdit la présence de cette séquence binaire dans la séquence de la DLSDU. La DLE ne doit accepter comme DLPDU une salve de signaux reçue qu'après avoir vérifié cette séquence, et doit retirer cette séquence avant de transférer la séquence de la DLSDU à l'utilisateur de DLS.

5.3.1.3 Champ Adresse de station

Le Tableau 35 indique la plage de valeurs du champ d'adresse de station.

Tableau 35 – Plage de valeurs du champ Adresse de station

Adresse de station	Contenu
0x00	Réservé
0x01	Maître C1
0x02	Maître C2
0x03 à 0xDE	Esclaves
0xDF	Non connecté (par défaut)
0xE0 à 0xFE	Réservé
0xFF	Adresse de diffusion

5.3.1.4 Champ Commande (CTL)

Le champ Commande sert à identifier le type de trame. Ce champ possède deux formats. L'un est le format d'échange d'E/S, l'autre est le format de message. Le Tableau 36 au Tableau 38 donnent le détail de ce champ.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Tableau 36 – Format du champ Commande (Format d'échange de données d'E/S)

Numéro d'octet	Taille	Symbole	Contenu
1	Unsigned4	CMD	Indicateur de données d'entrée ou données de sortie; 1: Données d'entrée (réponse) 3: Données de sortie (commande)
	Unsigned4	-	Réservé (0)[a]
a Ce bit doit être à zéro.			

Tableau 37 – Format du champ Commande (Format du message)

Numéro d'octet	Taille	Symbole	Contenu
1	Unsigned4	S(n)	Numéro de séquence
	Unsigned1	-	Réservé (1)[a]
	Unsigned1	C1/C2	Indicateur de nœud principal
	Unsigned1	END	Indicateur de fin
	Unsigned1	S/D	Indicateur Sync / données
a Ce bit doit être à un.			

Tableau 38 – Liste des champs du format de message

Symbole	Description
S(n)	Transmission du numéro de séquence d'envoi ou de réception
C1/C2	0: maître C1 1: maître C2
END	Si S/D est à 0, 0: Transfert de données du nœud principal vers le nœud secondaire 1: Transfert de données du nœud secondaire vers le nœud principal Si S/D est à 1, 0: Il existe d'autres données 1: Ces données sont les dernières
S/D	0: Trame d'établissement de liaison 1: Trame de données

5.3.1.5 Champ Données d'utilisateur de DLS

La longueur du champ de données d'utilisateur de DLS est comprise entre 8 octets et 64 octets.

5.3.1.6 Champ de CRC

On utilise une séquence de vérification de trame (FCS) utilisant le CRC-CCITT 16 bits. Le CRC est calculé d'après la formule suivante.

$$P(x) = X^{16} + X^{12} + X^{5} + 1$$

5.3.1.7 Indicateur de fin

La séquence de symboles de ce champ est identique à celle de l'indicateur de début (voir 5.3.1.2).

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

5.3.2 Trame synchrone

Le maître C1 utilise cette trame pour synchroniser les esclaves et le maître C2. Seul le maître C1 peut envoyer cette trame. Le maître C1 doit définir l'adresse de station en tant qu'adresse de diffusion (0xFF), et les esclaves et le maître C2 doivent recevoir cette trame. Lorsqu'ils reçoivent cette trame, ils doivent vérifier le champ de CRC contenu dans cette trame, puis émettre un événement cyclique si le résultat de la vérification ne contient pas d'erreur.

Le Tableau 39 et le Tableau 40 donnent le détail du format des données de cette trame.

Tableau 39 – Format des données de la trame synchrone

Numéro d'octet	Taille	Contenu
1 à 2	Unsigned16	Cycle de transmission
3 à 4	Unsigned16	Largeur d'intervalle de temps
5 à 16 ou 31	-	Réservé

Tableau 40 – Liste des champs de la trame de synchronisation

Champ	Contenu	Valeurs maximale et minimale	Note
Cycle de transmission	Cycle de transmission indiqué au maître C2	0 à 2^{16}-1	L'unité est de 0,25 µs
Largeur d'intervalle de temps	Intervalle de temps	0 à 2^{16}-1	L'unité est de 0,25 µs

5.3.3 Trame de données de sortie ou de données d'entrée

5.3.3.1 Trame de données de sortie

Le maître C1 utilise cette trame pour les données de sortie à envoyer à l'esclave dans la largeur de bande d'échange de données d'E/S du mode de transmission cyclique. Seul le maître C2 peut recevoir cette trame. La longueur des données de sortie doit être de 16 octets ou de 31 octets.

Le Tableau 41 et le Tableau 42 donnent le détail du format des données de cette trame.

Tableau 41 – Format des données de la trame de données de sortie

Numéro d'octet	Taille	Contenu
1 à 16 ou 31	Unsigned8n	Données de sortie

Tableau 42 – Liste des champs de la trame de données de sortie

Champ	Contenu	Valeurs maximale et minimale	Valeur recommandée	Note
Données de sortie	Données de sortie transitant du maître C1 vers l'esclave	De 0 à la valeur maximale correspondant à la longueur des données	–	

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

5.3.3.2 Trame de données d'entrée

Les esclaves utilisent cette trame pour les données d'entrée à envoyer au maître C1 dans la largeur de bande d'échange de données d'E/S du mode de transmission cyclique. Seul le maître C2 peut recevoir cette trame. La longueur des données d'entrée doit être de 16 octets ou de 31 octets.

Le Tableau 43 et le Tableau 44 donnent le détail du format des données de cette trame.

Tableau 43 – Format des données de la trame de données d'entrée

Numéro d'octet	Taille	Contenu
1 à 16 ou 31	Unsigned8n	Données d'entrée

Tableau 44 – Liste des champs de la trame de données d'entrée

Champ	Contenu	Valeurs maximale et minimale	Valeur recommandée	Note
Données d'entrée	Données d'entrée transitant de l'esclave vers le maître C1	De 0 à la valeur maximale correspondant à la longueur des données	–	

5.3.4 Trame de message

Cette trame est utilisée pour la transmission de message. Toutes les stations peuvent envoyer cette trame. Si la taille de la DLSDU à envoyer dépasse ce que peut contenir une trame, la DLSDU doit être divisée au moyen du champ Contrôle de message de cette trame.

6 Procédure d'élément DLE

6.1 Présentation

Le paragraphe suivant décrit le fonctionnement de la CTC et de la SRC. Bien que la gestion de DL fasse partie des DLE, elle est décrite dans le prochain article.

6.2 Sous-couche de commande de transmission cyclique

6.2.1 Généralités

La CTC comporte les fonctions suivantes.

- Commande de la séquence de communication et de la bande de communication (maître C1)
- Surveillance de toutes les données d'E/S échangées entre le maître C1 et les esclaves (maître C2)
- Envoi de réponse en tant que serveur (esclave)
- Communication de messages en communication cyclique
- Échange de données sporadique

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

6.2.2 Interface d'utilisateur de DLS

6.2.2.1 Définitions des primitives

Les primitives échangées entre l'utilisateur de DLS et la CTC sont indiquées ci-dessous. Le Tableau 45 concerne celles émises par l'utilisateur de DLS, le Tableau 46 celles émises par la CTC.

Tableau 45 – Primitives et paramètres de l'interface d'utilisateur de DLS émis par l'utilisateur de DLS

Primitive	Source	Paramètre	Fonction
DL-WRITE-DATA.req	utilisateur de DLS	SAP_ID, données d'utilisateur de DLS	Envoi de demande d'écriture de données
DL-READ-DATA.req	utilisateur de DLS	SAP_ID	Réception de demande de lecture de données
DL-SDA.req	utilisateur de DLS	SAP_ID, Node_ID, longueur, données d'utilisateur de DLS	Envoi de demande de données avec confirmation de transmission
DL-SDN.req	utilisateur de DLS	SAP_ID, Node_ID, longueur, données d'utilisateur de DLS	Envoi de demande de données sans confirmation de transmission
DL_GET_STATUS.req	utilisateur de DLS	Sts_ID	Demande faisant référence à un paramètre DL

Tableau 46 – Primitives et paramètres de l'interface d'utilisateur de DLS émis par la CTC

Primitive	Source	Paramètre
DL-WRITE-DATA.cnf	CTC	Résultat
DL-READ-DATA.cnf	CTC	Résultat, données d'utilisateur de DLS
DL-SDA.ind	CTC	Node_ID, longueur, données d'utilisateur de DLS
DL-SDA.cnf	CTC	Résultat
DL-SDN.ind	CTC	Node_ID, longueur, données d'utilisateur de DLS
DL-SDN.cnf	CTC	Résultat
DL_GET_STATUS.cnf	CTC	Résultat, Cur_Status
DL_EVENT.ind	CTC	Event_ID

6.2.2.2 Aperçu des interactions

Voir 4.3 de la CEI 61158-3-24.

6.2.2.3 Définitions détaillées des primitives et des paramètres

Voir 4.4 de la CEI 61158-3-24.

6.2.3 Machines protocolaires de la CTC

6.2.3.1 Présentation

Il existe trois machines protocolaires dans la CTC: la machine protocolaire de transmission cyclique, la machine protocolaire de segmentation de messages et la machine protocolaire de transmission acyclique.

La machine protocolaire de transmission cyclique comprend deux sortes de machines protocolaires correspondant au format de trame adopté. L'une est le "type à intervalle de temps de largeur fixe", dont la séquence de transmission est composée d'intervalles de temps

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

de largeur identique, l'autre est le "type à intervalle de temps configurable", dont la séquence de transmission peut comporter des intervalles de temps différents d'une station à l'autre. De plus, chacune de ces deux machines protocolaires, le "type à intervalle de temps de largeur fixe" et le "type à intervalle de temps configurable", possède trois sortes de machines protocolaires correspondant au type de station (le maître C1, le maître C2 et l'esclave).

La machine protocolaire de communication cyclique possède deux machines protocolaires: l'émetteur et le récepteur.

La machine protocolaire de communication de messages ne fonctionne que lorsque la machine protocolaire de communication cyclique est active.

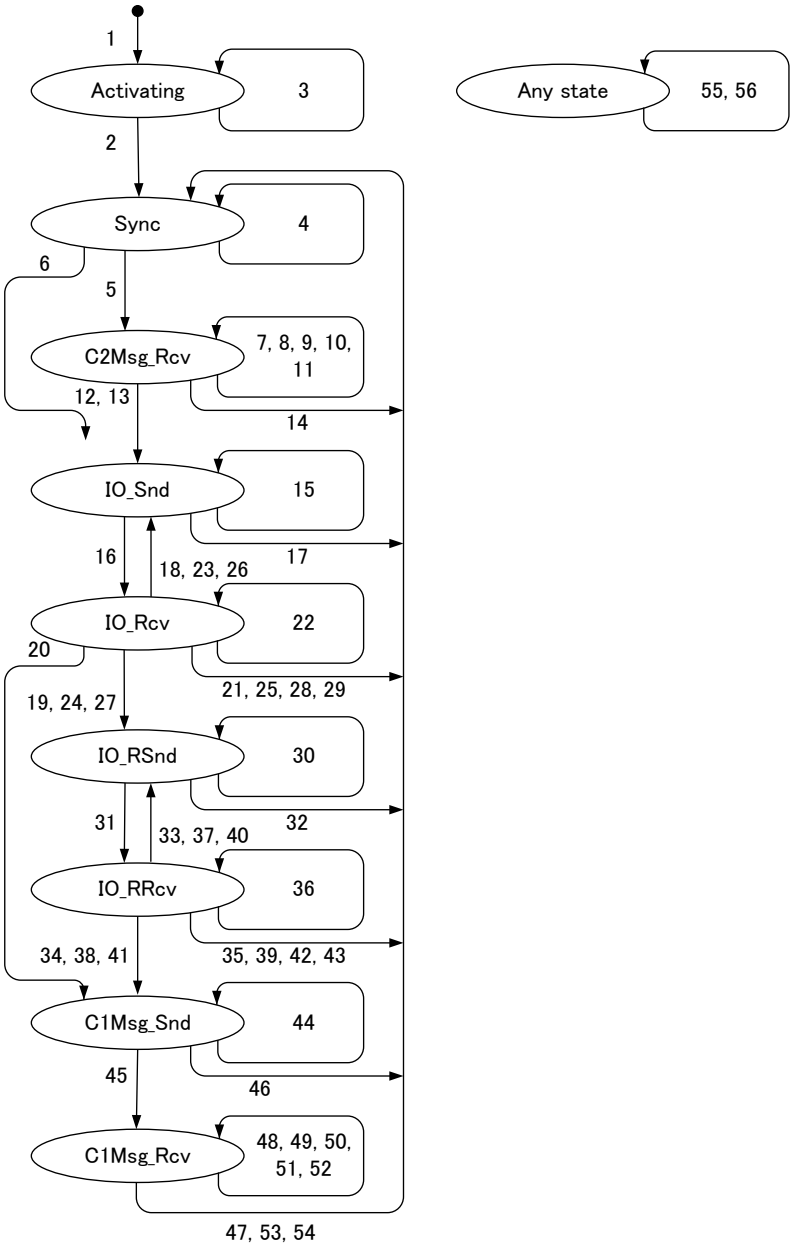
6.2.3.2 Machine protocolaire de transmission cyclique

6.2.3.2.1 La machine protocolaire utilisée pour la communication cyclique comprend des intervalles de temps de largeur fixe

6.2.3.2.1.1 Maître C1

La Figure 9 et le Tableau 47 montrent le schéma d'états et la table d'états du maître C1 adoptant des intervalles de temps de largeur fixe.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe



Légende

Anglais	Français
Activating	Activation en cours
Any state	N'importe quel état
Sync	Sync

Figure 9 – Schéma d'états du maître C1 utilisant des intervalles de temps de largeur fixe

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Tableau 47 – Table d'états du maître C1 utilisant des intervalles de temps de largeur fixe

N°	État courant	Événement /condition => action	État suivant
1	N'importe quel état	Power on or CTC_Reset.req => (none)	Activation en cours
2	Activation en cours	CTC_Start.req / CHECK_VARS() = True => START_TIMER(T(Tcycle), V(Tcycle)) CTC_Start.cnf { OK }	Sync
3	Activation en cours	CTC_Start.req / CHECK_VARS() <> True => CTC_Start.cnf { NG }	Activation en cours
4	Sync	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } SRCSDU:= BUILD_PDU_SYNC() Node_ID:= GET_DA(SRCSDU) Length:= GET_LEN(SRCSDU) SRC_Send_Frame.req { Node_ID, Length, SRCSDU} START_TIMER(T(Tslot), V(Tslot)) V(Nslave):= 1 V(Nretry):= 0 V(Nrest_slot):= V(Nmax_retry)	Sync
5	Sync	SRC_Send_Frame.cnf { Result } / V(Fc2msg) = “True” => (none)	C2Msg_Rcv
6	Sync	SRC_Send_Frame.cnf { Result } / V(Fc2msg) = “Flase” => (none)	IO_Snd
7	C2Msg_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && GET_MC(RcvSRCSDU) = Ec2msg)) => EXEC_MSPM_R(Ec2msg, Rcv_sts, RcvSRCSDU, SndSRCSDU) Node_ID:= GET_DA(SndSRCSDU) Length:= GET_LEN(SndSRCSDU) SRC_Send_Frame.req { Node_ID, Length, SRCSDU}	C2Msg_Rcv
8	C2Msg_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && GET_MC(RcvSRCSDU) = Ec1msg)) => (none)	C2Msg_Rcv
9	C2Msg_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) <> FT_MSG)) => (none)	C2Msg_Rcv
10	C2Msg_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU }	C2Msg_Rcv

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		/ ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) <> V(MA))) => (none)	
11	C2Msg_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / (Rcv_sts <> OK) => (none)	C2Msg_Rcv
12	C2Msg_Rcv	SRC_Send_Frame.cnf { Result } => (none)	IO_Snd
13	C2Msg_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" => SRCSDU:= BUILD_PDU_OUT(V(Nslave)) Node_ID:= GET_DA(SndSRCSDU) Length:= GET_LEN(SndSRCSDU) SRC_Send_Frame.req { Node_ID, Length, SRCSDU}	IO_Snd
14	C2Msg_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot))	Sync
15	IO_Snd	EXPIRED_TIMER (T(Tslot)) = "True" => SRCSDU:= BUILD_PDU_OUT(V(Nslave)) Node_ID:= GET_DA(SndSRCSDU) Length:= GET_LEN(SndSRCSDU) SRC_Send_Frame.req { Node_ID, Length, SRCSDU}	IO_Snd
16	IO_Snd	SRC_Send_Frame.cnf { } => (none)	IO_Rcv
17	IO_Snd	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot)	Sync
18	IO_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) = GET_NODE_ID(V(Nslave)) && (V(Nslave) < V(Nmax_slave)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts) V(Nslave):= V(Nslave) + 1	IO_Snd
19	IO_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) = GET_NODE_ID(V(Nslave)) && (V(Nslave) >= V(Nmax_slave) && (V(Nrest_slot) > 0) && (V(Nretry) > 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts)	IO_RSnd
20	IO_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) = GET_NODE_ID(V(Nslave)) && (V(Nslave) >= V(Nmax_slave) && (V(Nrest_slot) > 0) && (V(Nretry) = 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts)	C1Msg_Snd
21	IO_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) = GET_NODE_ID(V(Nslave)) && (V(Nslave) >= V(Nmax_slave) && (V(Nrest_slot) <= 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts)	Sync

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
22	IO_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) <> GET_NODE_ID(V(Nslave))) => (none)	IO_Rcv
23	IO_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts <> OK) && (V(Nslave) < V(Nmax_slave)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts) PUT_RETRY_LIST(V(Nslave)) V(Nslave):= V(Nslave) + 1	IO_Snd
24	IO_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts <> OK) && (V(Nslave) >= V(Nmax_slave) && (V(Nrest_slot) > 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts) PUT_RETRY_LIST(V(Nslave)) V(Nslave):= V(Nslave) + 1	IO_RSnd
25	IO_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts <> OK) && (V(Nslave) >= V(Nmax_slave) && (V(Nrest_slot) <= 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts) PUT_RETRY_LIST(V(Nslave)) V(Nslave):= V(Nslave) + 1	Sync

26	IO_Rcv	EXPIRED_TIMER (T(Tslot)) = "True"
		/ V(Nslave) < V(Nmax_slave)
		=> STORE_PDU_IN(V(Nslave), SRCSDU, NG)
		If V(Nrest_slot) > 0 then
		PUT_RETRY_LIST(V(Nslave))
		endif
		V(Nslave):= V(Nslave) + 1
			IO_Snd

27	IO_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) > 0) => STORE_PDU_IN(V(Nslave), SRCSDU, NG) PUT_RETRY_LIST(V(Nslave)) V(Nslave):= GET_RETRY_LIST() SRCSDU:= BUILD_PDU_OUT(V(Nslave)) Node_ID:= GET_DA(SRCSDU) Length:= GET_LEN(SRCSDU) SRC_Send_Frame.req { Node_ID, Length, SRCSDU}	IO_RSnd
28	IO_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) = 0) => STORE_PDU_IN(V(Nslave), SRCSDU, NG)	Sync
29	IO_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot) STORE_PDU_IN(V(Nslave), SRCSDU, NG)	Sync
30	IO_RSnd	EXPIRED_TIMER (T(Tslot)) = "True"	IO_RSnd

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		=> V(Nslave):= GET_RETRY_LIST() SRCSDU:= BUILD_PDU_OUT(V(Nslave)) Node_ID:= GET_DA(SRCSDU) Length:= GET_LEN(SRCSDU) SRC_Send_Frame.req { Node_ID, Length, SRCSDU}	
31	IO_RSnd	SRC_Send_Frame.cnf { } => (none)	IO_RRcv
32	IO_RSnd	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot)	Sync
33	IO_RRcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) = GET_NODE_ID(V(Nslave)) && (V(Nrest_slot) >= 1) && (V(Nretry) > 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts) V(Nrest_slot):= V(Nrest_slot) – 1	IO_RSnd
34	IO_RRcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) = GET_NODE_ID(V(Nslave)) && (V(Nrest_slot) >= 1) && (V(Nretry) <= 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts)	C1Msg_Snd
35	IO_RRcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) = GET_NODE_ID(V(Nslave)) && (V(Nrest_slot) < 1)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts)	Sync
36	IO_RRcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) <> GET_NODE_ID(V(Nslave))) => (none)	IO_RRcv
37	IO_RRcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts <> OK) && (V(Nrest_slot) >= 1) && (V(Nretry) > 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts) V(Nrest_slot):= V(Nrest_slot) – 1	IO_RSnd
38	IO_RRcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts <> OK) && (V(Nrest_slot) >= 1) && (V(Nretry) <= 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts) V(Nrest_slot):= V(Nrest_slot) – 1	C1Msg_Snd
39	IO_RRcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts <> OK) && (V(Nrest_slot) < 1)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts)	Sync
40	IO_RRcv	EXPIRED_TIMER (T(Tslot)) = "True" / ((V(Nrest_slot) >= 1) && (V(Nretry) > 0)) => STORE_PDU_IN(V(Nslave), None, TOUT) V(Nrest_slot):= V(Nrest_slot) – 1 V(Nslave):= GET_RETRY_LIST()	IO_RSnd

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		SRCSDU:= BUILD_PDU_OUT(V(Nslave)) Node_ID:= GET_DA(SRCSDU) Length:= GET_LEN(SRCSDU) SRC_Send_Frame.req { Node_ID, Length, SRCSDU}	
41	IO_RRcv	EXPIRED_TIMER (T(Tslot)) = "True" / ((V(Nrest_slot) >= 1) && (V(Nretry) <= 0)) => STORE_PDU_IN(V(Nslave), None, TOUT) V(Nrest_slot):= V(Nrest_slot) – 1 EXEC_MSPM_IS(Ec1msg, SRCSDU) Node_ID:= GET_DA(SRCSDU) Length:= GET_LEN(SRCSDU) SRC_Send_Frame.req { Node_ID, Length, SRCSDU}	C1Msg_Snd
42	IO_RRcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nrest_slot) < 1) => STORE_PDU_IN(V(Nslave), None, TOUT)	Sync
43	IO_RRcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot) STORE_PDU_IN(V(Nslave), SRCSDU, TOUT)	Sync
44	C1Msg_Snd	EXPIRED_TIMER (T(Tslot)) = "True" => EXEC_MSPM_IS(Ec1msg, SndSRCSDU) Node_ID:= GET_DA(SndSRCSDU) Length:= GET_LEN(SndSRCSDU) SRC_Send_Frame.req { Node_ID, Length, SndSRCSDU}	C1Msg_Snd
45	C1Msg_Snd	SRC_Send_Frame.cnf { } => (none)	C1Msg_Rcv
46	C1Msg_Snd	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot)	Sync
47	C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, RcvSRCSDU} / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && GET_MC(RcvSRCSDU) = Ec1msg)) => STOP_TIMER(T(Tslot)) EXEC_MSPM_IR(E1msg, Rcv_sts, RcvSRCSDU)	Sync
48	C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, RcvSRCSDU} / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && GET_MC(RcvSRCSDU) = Ec2msg)) => (none)	C1Msg_Rcv
49	C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, RcvSRCSDU} / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && GET_MC(RcvSRCSDU) = Ec2msg)) => (none)	C1Msg_Rcv
50	C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, RcvSRCSDU}	C1Msg_Rcv

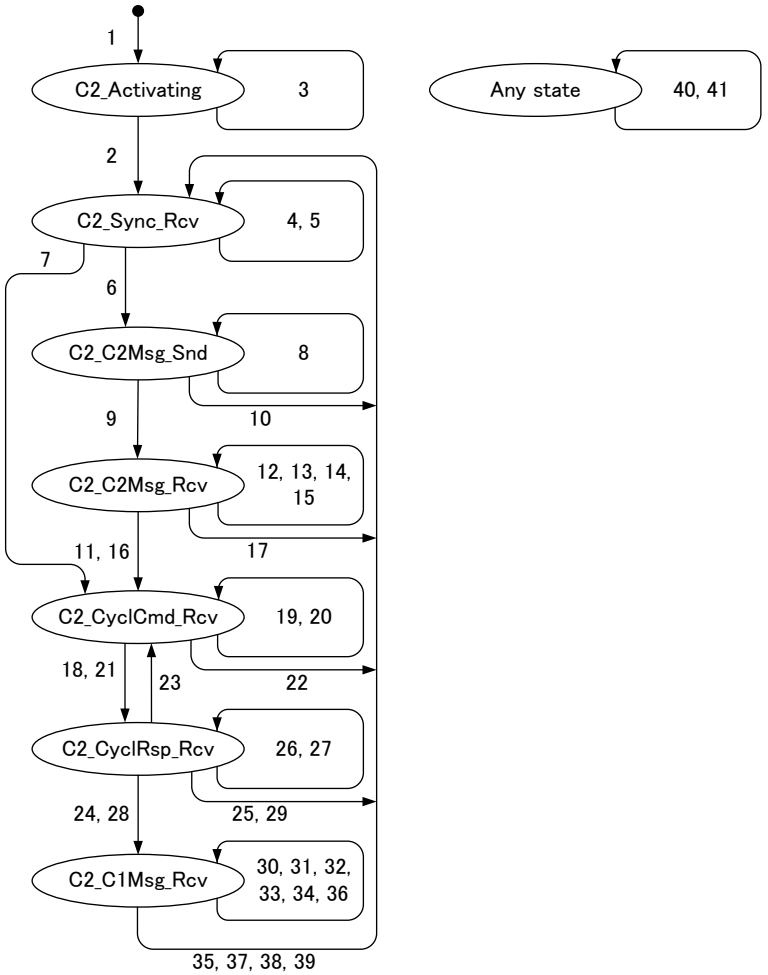
Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		/ ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) <> FT_MSG)) => (none)	
51	C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, RcvSRCSDU} / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) <> V(MA))) => (none)	C1Msg_Rcv
52	C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, RcvSRCSDU} / ((Rcv_sts <> OK) => (none)	C1Msg_Rcv
53	C1Msg_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" => STOP_TIMER(T(Tslot))	Sync
54	C1Msg_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot))	Sync
55	N'importe quel état	DL-WRITE-DATA.req(SAP_ID, DLSDU) => Result:= SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	Même état
56	N'importe quel état	DL-READ-DATA.req(SAP_ID) => Result:= GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	Même état

6.2.3.2.1.2 Maître C2

La Figure 10 et le Tableau 48 montrent le schéma d'états et la table d'états du maître C2 adoptant des intervalles de temps de largeur fixe.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe



Légende

Anglais	Français
Any state	N'importe quel état

Figure 10 – Schéma d'états du maître C2 utilisant des intervalles de temps de largeur fixe

Tableau 48 – Table d'états du maître C2 utilisant des intervalles de temps de largeur fixe

N°	État courant	Événement /condition => action	État suivant
1	N'importe quel état	Power on or CTC_Reset.req => (none)	C2_Activating
2	C2_Activating	CTC_Start.req / CHECK_VARS() = True => START_TIMER(T(Tcycle), V(Tcycle)) CTC_Start.cnf { OK }	C2_Sync_Rcv
3	C2_Activating	CTC_Start.req / CHECK_VARS() <> True => CTC_Start.cnf { NG }	C2_Activating
4	C2_Sync_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) <> Cgaddr))	C2_Sync_Rcv

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		=> (none)	
5	C2_Sync_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / (Rcv_sts <> OK) => (none)	C2_Sync_Rcv
6	C2_Sync_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" / V(Fc2msg) = “True” => DL_Event.ind {DL_Ev_Tcycle } START_TIMER(T(Tslot), V(Tslot)) V(Nslave):= 1	C2_C2Msg_Snd
7	C2_Sync_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" / V(Fc2msg) = “False” => DL_Event.ind {DL_Ev_Tcycle } START_TIMER(T(Tslot), V(Tslot)) V(Nslave):= 1	C2_Out_Rcv
8	C2_C2Msg_Snd	EXPIRED_TIMER (T(Tslot)) = "True" => EXEC_MSPM_IS(Ec2msg, SndSRCSDU) Node_ID:= GET_DA(SndSRCSDU) Length:= GET_LEN(SndSRCSDU) SRC_Send_Frame.req { Node_ID, Length, SndSRCSDU}	C2_C2Msg_Snd
9	C2_C2Msg_Snd	SRC_Send_Frame.cnf { } => (none)	C2_C2Msg_Rcv
10	C2_C2Msg_Snd	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot)	C2_Sync_Rcv
11	C2_C2Msg_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && GET_MC(RcvSRCSDU) = Ec2msg)) => EXEC_MSPM_IR(E1msg, Rcv_sts, RcvSRCSDU)	C2_Out_Rcv
12	C2_C2Msg_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && GET_MC(RcvSRCSDU) = Ec1msg)) => (none)	C2_C2Msg_Rcv
13	C2_C2Msg_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) <> FT_MSG)) => (none)	C2_C2Msg_Rcv
14	C2_C2Msg_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) <> V(MA))) => (none)	C2_C2Msg_Rcv
15	C2_C2Msg_Rcv	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / (Rcv_sts <> OK) => (none)	C2_C2Msg_Rcv
16	C2_C2Msg_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" => (none)	C2_Out_Rcv

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
17	C2_C2Msg_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot))	C2_Sync_Rcv
18	C2_Out_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_FT(RcvSRCSDU) = FT_OUT)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts)	C2_In_Rcv
19	C2_Out_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_FT(RcvSRCSDU) <> FT_OUT)) => (none)	C2_Out_Rcv
20	C2_Out_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / (Rcv_sts <> OK) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts)	C2_Out_Rcv
21	C2_Out_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" => (none)	C2_In_Rcv
22	C2_Out_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot))	C2_Sync_Rcv
23	C2_In_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_FT(RcvSRCSDU) = FT_IN) && (V(Nslave) < V(Nmax_slave))) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts) V(Nslave):= V(Nslave) + 1	C2_Out_Rcv
24	C2_In_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_FT(RcvSRCSDU) = FT_IN) && (V(Nslave) >= V(Nmax_slave)) && (V(Nmax_retry) > 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts) V(Nslave):= V(Nslave) + 1 V(Nretry):= V(Nmax_retry)	C2_C1Msg_Rcv
25	C2_In_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_FT(RcvSRCSDU) = FT_IN) && (V(Nslave) >= V(Nmax_slave)) && (V(Nmax_retry) <= 0)) => STORE_PDU_IN(V(Nslave), SRCSDU, Rcv_sts) V(Nslave):= V(Nslave) + 1	C2_Sync_Rcv
26	C2_In_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_FT(RcvSRCSDU) <> FT_IN)) => (none)	C2_In_Rcv
27	C2_In_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts <> OK) && (V(Nslave) < V(Nmax_slave))) => (none)	C2_In_Rcv
28	C2_In_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts <> OK) && (V(Nslave) >= V(Nmax_slave)) && (V(Nmax_retry) > 0)) => V(Nretry):= V(Nmax_retry)	C2_C1Msg_Rcv
29	C2_In_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts <> OK) && (V(Nslave) >= V(Nmax_slave)) && (V(Nmax_retry) <= 0))	C2_Sync_Rcv

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		=> (none)	
30	C2_C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && (GET_MC(RcvSRCSDU) = Ec1msg)) => EXEC_MSPM_R(Ec1msg, Rcv_sts, RcvSRCSDU, SndSRCSDU) Node_ID:= GET_DA(SndSRCSDU) Length:= GET_LEN(SndSRCSDU) SRC_Send_Frame.req { Node_ID, Length, SRCSDU}	C2_C1Msg_Rcv
31	C2_C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && (GET_MC(RcvSRCSDU) = Ec2msg)) => (none)	C2_C1Msg_Rcv
32	C2_C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) <> FT_MSG)) => (none)	C2_C1Msg_Rcv
33	C2_C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (GET_DA(SRCSDU) <> V(MA))) => (none)	C2_C1Msg_Rcv
34	C2_C1Msg_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / (Rcv_sts <> OK) => (none)	C2_C1Msg_Rcv
35	C2_C1Msg_Rcv	SRC_Send_Frame.cnf { } => (none)	C2_Sync_Rcv
36	C2_C1Msg_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / V(Nretry) > 0 => V(Nretry):= V(Nmax_retry) – 1	C2_C1Msg_Rcv
37	C2_C1Msg_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / V(Nretry) <= 0 => STOP_TIMER(T(Tslot)	C2_Sync_Rcv
38	C2_C1Msg_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot))	C2_Sync_Rcv
39	N'importe quel état	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = Cgaddr) && (GET_FT(RcvSRCSDU) = FT_SYNC)) => DL_Event.ind {DL_Ev_Tcycle } RESTART_TIMER(T(Tcycle)) START_TIMER(T(Tslot), V(Tslot)) V(Nslave):= 1	C2_Sync_Rcv
40	N'importe quel état	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = Cgaddr) && (GET_FT(RcvSRCSDU) <> FT_SYNC)) => (none)	Même état

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
41	N'importe quel état	DL-READ-DATA.req(SAP_ID) => Result:= GET_CYC_DATA(SAP_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	Même état

6.2.3.2.1.3 Esclave

La Figure 11 et le Tableau 49 montrent le schéma d'états et la table d'états de l'esclave adoptant des intervalles de temps de largeur fixe.

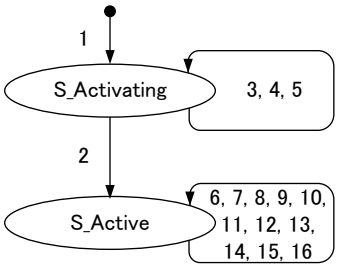


Figure 11 – Schéma d'états de l'esclave utilisant des intervalles de temps de largeur fixe

Tableau 49 – Table d'états de l'esclave utilisant des intervalles de temps de largeur fixe

N°	État courant	Événement /condition =>action	État suivant
1	N'importe quel état	Power on or CTC_Reset.req => (none)	S_Activating
2	S_Activating	CTC_Start.req / CHECK_VARS() = True => START_TIMER(T(Tcycle), V(Tcycle)) CTC_Start.cnf { OK }	S_Active
3	S_Activating	CTC_Start.req / CHECK_VARS() <> True => CTC_Start.cnf { NG }	Activation en cours
4	S_Activating	DL-WRITE-DATA.req(SAP_ID, DLSDU) => Result:= SET_CYC_DATA(SAP_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	S_Activating
5	S_Activating	DL-READ-DATA.req(SAP_ID) => Result:= GET_CYC_DATA(SAP_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	S_Activating
6	S_Active	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle }	S_Active
7	S_Active	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = Cgaddr) && (GET_FT(RcvSRCSDU) = FT_SYNC)) => DL_Event.ind {DL_Ev_Tcycle } RESTART_TIMER(T(Tcycle))	S_Active

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

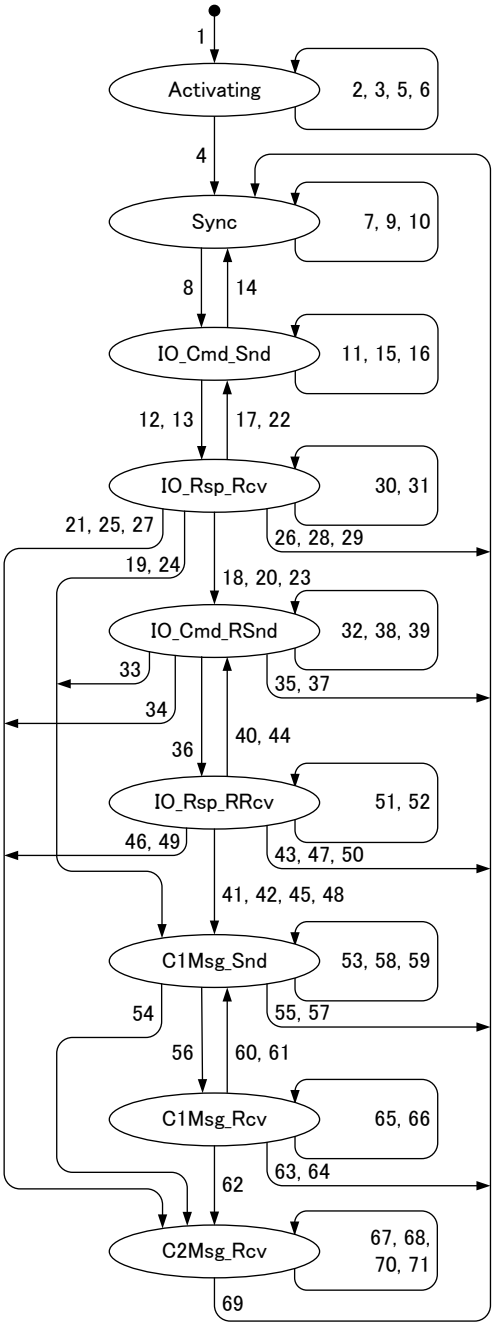
N°	État courant	Événement /condition =>action	État suivant
8	S_Active	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = Cgaddr) && (GET_FT(RcvSRCSDU) <> FT_SYNC)) => (none)	S_Active
9	S_Active	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_OUT)) => SndSRCSDU:= BUILD_PDU_IN(1) Node_ID:= V(MA) Length:= GET_LEN(SndSRCSDU) SRC_Send_Frame.req { Node_ID, Length, SndSRCSDU} STORE_PDU_OUT(1, RcvSRCSDU, Rcv_sts)	S_Active
10	S_Active	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && GET_MC(RcvSRCSDU) = Ec1msg)) => EXEC_MSPM_R(Ec1msg, Rcv_sts, RcvSRCSDU, SndSRCSDU) Node_ID:= MA Length:= GET_LEN(SndSRCSDU) SRC_Send_Frame.req { Node_ID, Length, SndSRCSDU}	S_Active
11	S_Active	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) = FT_MSG) && GET_MC(RcvSRCSDU) = Ec2msg)) => EXEC_MSPM_R(Ec2msg, Rcv_sts, RcvSRCSDU, SndSRCSDU) Node_ID:= MA Length:= GET_LEN(SndSRCSDU) SRC_Send_Frame.req { Node_ID, Length, SndSRCSDU}	S_Active
12	S_Active	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_FT(RcvSRCSDU) <> FT_MSG) && (GET_FT(RcvSRCSDU) <> FT_OUT)) => (none)	S_Active
13	S_Active	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) <> V(MA))) => (none)	S_Active
14	S_Active	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / (Rcv_sts <> OK) => (none)	S_Active
15	S_Active	DL-WRITE-DATA.req(SAP_ID, DLSDU) => Result:= SET_CYC_DATA(SAP_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	S_Active
16	S_Active	DL-READ-DATA.req(SAP_ID) => Result:= GET_CYC_DATA(SAP_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	S_Active

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

6.2.3.2.2 La machine protocolaire utilisée pour la communication cyclique comprend des intervalles de temps configurables

6.2.3.2.2.1 Maître C1

La Figure 12 et le Tableau 50 montrent le schéma d'états et la table d'états du maître C1 adoptant des intervalles de temps configurables.



Légende

Anglais	Français
Activating	Activation en cours
Sync	Sync

Figure 12 – Schéma d'états du maître C1 utilisant des intervalles de temps configurables

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Tableau 50 – Table d'états du maître C1 utilisant des intervalles de temps configurables

N°	État courant	Événement /condition =>action	État suivant
1	N'importe quel état	Power on or CTC_Reset.req => (none)	Activation en cours
2	Activation en cours	CTC_Set_Par.req { Par_ID, Val } => CTC_SET_PAR(Par_ID, Val, Result) CTC_Set_Par.cnf { Result }	Activation en cours
3	Activation en cours	CTC_Get_Par.req { Par_ID } => CTC_GET_PAR(Par_ID, Val, Result) CTC_Get_Par.cnf { Result, Par_ID, Val }	Activation en cours
4	Activation en cours	CTC_Start.req => START_TIMER(T(Tcycle), V(Tcycle)) CTC_Start.cnf	Sync
5	Activation en cours	DL-WRITE-DATA.req(SAP_ID, DLSDU) => Result:= SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	Activation en cours
6	Activation en cours	DL-READ-DATA.req(SAP_ID) => Result:= GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	Activation en cours
7	Sync	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } SRCSDU:= BUILD_PDU_SYNC() SRC_Send_Frame.req {SRCSDU} V(Nslave):= 0 START_TIMER(T(Tslot), V(Tslot(V(Nslave))))	Sync
8	Sync	SRC_Send_Frame.cnf { } => V(Nslave):= 1 V(Nretry):= 0 V(Nrest_slot):= 0	IO_Cmd_Snd
9	Sync	DL-WRITE-DATA.req(SAP_ID, DLSDU) => Result:= SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	Sync
10	Sync	DL-READ-DATA.req(SAP_ID) => Result:= GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	Sync
11	IO_Cmd_Snd	EXPIRED_TIMER (T(Tslot)) = "True" => SRCSDU:= BUILD_PDU_CYCC(V(Nslave)) SRC_Send_Frame.req {SRCSDU} STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot), V(Tslot(V(Nslave))))	IO_Cmd_Snd
12	IO_Cmd_Snd	SRC_Send_Frame.cnf { } / V(Nslave) < V(Nmax_slave) => (none)	IO_Rsp_Rcv
13	IO_Cmd_Snd	SRC_Send_Frame.cnf { }	IO_Rsp_Rcv

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition =>action	État suivant
		/ V(Nslave) >= V(Nmax_slave) => V(Nrest_slot):= V(Nmax_retry)	
14	IO_Cmd_Snd	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle} STOP_TIMER(T(Tslot)) SRCSDU:= BUILD_PDU_SYNC() SRC_Send_Frame.req {SRCSDU} V(Nslave):= 0 START_TIMER(T(Tslot), V(Tslot(V(Nslave))))	Sync
15	IO_Cmd_Snd	DL-WRITE-DATA.req(SAP_ID, DLSDU) => result:= SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	IO_Cmd_Snd
16	IO_Cmd_Snd	DL-READ-DATA.req(SAP_ID) => result:= GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	IO_Cmd_Snd

17	IO_Rsp_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU}
		/ ((Rcv_sts = OK) && V(Nslave) < V(Nmax_slave))
		=> STORE_PDU_CYCR(V(Nslave), SRCSDU, Rcv_sts)
		If ((Rcv_sts <> OK) && (V(Nrest_slot) > 0)) then
		PUT_RETRY_LIST(V(Nslave))
		Endif
		V(Nslave):= V(Nslave) + 1
			IO_Cmd_Snd

18	IO_Rsp_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU}
		/ ((Rcv_sts = OK) && (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) > 0) && (V(Nretry) > 0))
		=> STORE_PDU_CYCR(V(Nslave), SRCSDU, Rcv_sts)
		If (Result <> OK) then
		PUT_RETRY_LIST(V(Nslave))
		Endif
			IO_Cmd_RSnd

19	IO_Rsp_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) > 0) && (V(Nretry) = 0)) => STORE_PDU_CYCR(V(Nslave), SRCSDU, Rcv_sts)	C1Msg_Snd
20	IO_Rsp_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) > 0) && (V(Nretry) = 0)) => STORE_PDU_CYCR(V(Nslave), SRCSDU, Rcv_sts) PUT_RETRY_LIST(V(Nslave))	IO_Cmd_RSnd
21	IO_Rsp_Rcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / ((Rcv_sts = OK) && (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) <= 0)) => STORE_PDU_CYCR(V(Nslave), SRCSDU, Result) SND_MSG_TOKEN()	C2Msg_Rcv
22	IO_Rsp_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / V(Nslave) < V(Nmax_slave)	IO_Cmd_Snd

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition =>action	État suivant

		=> STORE_PDU_CYCR(V(Nslave), SRCSDU, NG)
		If V(Nrest_slot) > 0 then
		PUT_RETRY_LIST(V(Nslave))
		Endif
		V(Nslave):= V(Nslave) + 1	

23	IO_Rsp_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) > 0) && CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) > 0 => STORE_PDU_CYCR(V(Nslave), SRCSDU, NG) PUT_RETRY_LIST(V(Nslave)) V(Nslave):= GET_RETRY_LIST() SRCSDU:= BUILD_PDU_CYCC(V(Nslave)) SRC_Send_Frame.req {SRCSDU} STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot), V(Tslot(V(slave))))	IO_Cmd_RSnd
24	IO_Rsp_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) > 0) && CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) = 0 && CHECK_MSG_EN(C1) <> 0 => STORE_PDU_CYCR(V(Nslave), SRCSDU, NG) EXEC_MSGC_SEG(Ec1msg, OK, None, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU } slot:= GET_MSG_SLOT(C1) STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot),slot))	C1Msg_Snd
25	IO_Rsp_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) > 0) && CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) = 0 && CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) <> 0 => STORE_PDU_CYCR(V(Nslave), SRCSDU, NG) SND_MSG_TOKEN()	C2Msg_Rcv
26	IO_Rsp_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) > 0) && CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) = 0 && CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) = 0 => STORE_PDU_CYCR(V(Nslave), SRCSDU, NG)	Sync
27	IO_Rsp_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) <= 0) && CHECK_MSG_EN(C2) <> 0 => STORE_PDU_CYCR(V(Nslave), SRCSDU, NG) SND_MSG_TOKEN()	C2Msg_Rcv
28	IO_Rsp_Rcv	EXPIRED_TIMER (T(Tslot)) = "True"	Sync

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition =>action	État suivant
		/ (V(Nslave) >= V(Nmax_slave)) && (V(Nrest_slot) <= 0) && CHECK_MSG_EN(C2) = 0 => STORE_PDU_CYCR(V(Nslave), SRCSDU, NG)	
29	IO_Rsp_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot)) STORE_PDU_CYCR(V(Nslave), SRCSDU, NG) SRCSDU:= BUILD_PDU_SYNC() SRC_Send_Frame.req {SRCSDU} V(Nslave):= 0 START_TIMER(T(Tslot), V(Tslot(V(Nslave))))	Sync
30	IO_Rsp_Rcv	DL-WRITE-DATA.req(SAP_ID, DLSDU) => result:= SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	IO_Rsp_Rcv
31	IO_Rsp_Rcv	DL-READ-DATA.req(SAP_ID) => result:= GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	IO_Rsp_Rcv
32	IO_Cmd_RSnd	EXPIRED_TIMER (T(Tslot)) = "True" / CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) > 0 => V(Nslave):= GET_RETRY_LIST() SRCSDU:= BUILD_PDU_CYCC(V(Nslave)) SRC_Send_Frame.req {SRCSDU} STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot), V(Tslot(V(slave))))	IO_Cmd_RSnd
33	IO_Cmd_RSnd	EXPIRED_TIMER (T(Tslot)) = "True" / CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) > 0 && CHECK_MSG_EN(C1) <> 0 =>EXEC_MSGC_SEG(Ec1msg, OK, None, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU } slot:= GET_MSG_SLOT(C1) STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot),slot))	C1Msg_Snd
34	IO_Cmd_RSnd	EXPIRED_TIMER (T(Tslot)) = "True" / CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) > 0 && CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) <> 0 => SND_MSG_TOKEN()	C2Msg_Rcv
35	IO_Cmd_RSnd	EXPIRED_TIMER (T(Tslot)) = "True" / CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) > 0 && CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) = 0 =>(none)	Sync
36	IO_Cmd_RSnd	SRC_Send_Frame.cnf { } => (none)	IO_Rsp_RRcv

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition =>action	État suivant
37	IO_Cmd_RSnd	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot)) SRCSDU:= BUILD_PDU_SYNC() SRC_Send_Frame.req {SRCSDU} V(Nslave):= 0 START_TIMER(T(Tslot), V(Tslot(V(Nslave))))	Sync
38	IO_Cmd_RSnd	DL-WRITE-DATA.req(SAP_ID, DLSDU) => result:= SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	IO_Cmd_RSnd
39	IO_Cmd_RSnd	DL-READ-DATA.req(SAP_ID) => result:= GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	IO_Cmd_RSnd
40	IO_Rsp_RRcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / (V(Nrest_slot) > 1) && (V(Nretry) > 0) => STORE_PDU_CYCR(V(Nslave), SRCSDU, Result) V(Nrest_slot):= V(Nrest_slot) – 1	IO_Cmd_RSnd
41	IO_Rsp_RRcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / (V(Nrest_slot) > 1) && (V(Nretry) <= 0) => STORE_PDU_CYCR(V(Nslave), SRCSDU, Result)	C1Msg_Snd
42	IO_Rsp_RRcv	SRC_Recv_Frame.ind {Rcv_sts, Length, SRCSDU} / (V(Nrest_slot) <= 1) => STORE_PDU_CYCR(V(Nslave), SRCSDU, Result)	C1Msg_Snd
43	IO_Rsp_RRcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot) STORE_PDU_CYCR(V(Nslave), SRCSDU, NG) SRCSDU:= BUILD_PDU_SYNC() SRC_Send_Frame.req {SRCSDU} V(Nslave):= 0 START_TIMER(T(Tslot), V(Tslot(V(Nslave))))	Sync
44	IO_Rsp_RRcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nrest_slot) > 1) && CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) > 0 => STORE_PDU_CYCR(V(Nslave), SRCSDU, NG) V(Nrest_slot):= V(Nrest_slot) – 1 V(Nslave):= GET_RETRY_LIST() SRCSDU:= BUILD_PDU_CYCC(V(Nslave)) SRC_Send_Frame.req {SRCSDU}	IO_Cmd_RSnd
45	IO_Rsp_RRcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nrest_slot) <= 1) && CHECK_MSG_EN(C1) <> 0 =>EXEC_MSGC_SEG(Ec1msg, OK, None, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU }	C1Msg_Snd

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition =>action	État suivant
		slot:= GET_MSG_SLOT(C1) STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot),slot))	
46	IO_Rsp_RRcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nrest_slot) <= 1) && CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) <> 0 => SND_MSG_TOKEN()	C2Msg_Rcv
47	IO_Rsp_RRcv	EXPIRED_TIMER (T(Tslot)) = "True" / (V(Nrest_slot) <= 1) && CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) = 0 =>(none)	Sync
48	IO_Rsp_RRcv	EXPIRED_TIMER (T(Tslot)) = "True" /CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) = 0 && CHECK_MSG_EN(C1) <> 0 =>EXEC_MSGC_SEG(Ec1msg, OK, None, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU } slot:= GET_MSG_SLOT(C1) STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot),slot))	C1Msg_Snd
49	IO_Rsp_RRcv	EXPIRED_TIMER (T(Tslot)) = "True" /CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) = 0 && CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) <> 0 => SND_MSG_TOKEN()	C2Msg_Rcv
50	IO_Rsp_RRcv	EXPIRED_TIMER (T(Tslot)) = "True" /CMP_RTIM(T(Tcycle), V(Tslot(V(GET_RETRY_LIST)))) = 0 && CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) = 0 =>(none)	Sync
51	IO_Rsp_RRcv	DL-WRITE-DATA.req(SAP_ID, DLSDU) => result:= SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	IO_Rsp_RRcv
52	IO_Rsp_RRcv	DL-READ-DATA.req(SAP_ID) => result:= GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	IO_Rsp_RRcv
53	C1Msg_Snd	EXPIRED_TIMER (T(Tslot)) = "True" / CHECK_MSG_EN(C1) <> 0 => EXEC_MSGC_SEG(Ec1msg, OK, None, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU } slot:= GET_MSG_SLOT(C1) STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot),slot))	C1Msg_Snd

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition =>action	État suivant
54	C1Msg_Snd	EXPIRED_TIMER (T(Tslot)) = "True" / CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) <> 0 => SND_MSG_TOKEN()	C2Msg_Rcv
55	C1Msg_Snd	EXPIRED_TIMER (T(Tslot)) = "True" / CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) = 0 =>(none)	Sync
56	C1Msg_Snd	SRC_Send_Frame.cnf { } => (none)	C1Msg_Rcv
57	C1Msg_Snd	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot) SRCSDU:= BUILD_PDU_SYNC() SRC_Send_Frame.req {SRCSDU} V(Nslave):= 0 START_TIMER(T(Tslot), V(Tslot(V(Nslave))))	Sync
58	C1Msg_Snd	DL-WRITE-DATA.req(SAP_ID, DLSDU) => result:= SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	C1Msg_Snd
59	C1Msg_Snd	DL-READ-DATA.req(SAP_ID) => result:= GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	C1Msg_Snd
60	C1Msg_Rcv	SRC_Recv_Frame.ind {Result, RcvSRCSDU} => STOP_TIMER(T(Tslot)) EXEC_MSGC_SEG(Ec1msg, Result, RcvSRCSDU, None)	C1Msg_Snd
61	C1Msg_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / CHECK_MSG_EN(C1) <> 0 => EXEC_MSGC_SEG(Ec1msg, OK, None, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU } slot:= GET_MSG_SLOT(C1) STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot),slot))	C1Msg_Snd
62	C1Msg_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) <> 0 => SND_MSG_TOKEN()	C2Msg_Rcv
63	C1Msg_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / CHECK_MSG_EN(C1) = 0 && CHECK_MSG_EN(C2) = 0 =>(none)	Sync
64	C1Msg_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot))	Sync

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition =>action	État suivant
		SRCSDU:= BUILD_PDU_SYNC() SRC_Send_Frame.req {SRCSDU} V(Nslave):= 0 START_TIMER(T(Tslot), V(Tslot(V(Nslave))))	
65	C1Msg_Rcv	DL-WRITE-DATA.req(SAP_ID, DLSDU) => result:= SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	C1Msg_Rcv
66	C1Msg_Rcv	DL-READ-DATA.req(SAP_ID) => result:= GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	C1Msg_Rcv
67	C2Msg_Rcv	SRC_Recv_Frame.ind { Result, RcvSRCSDU } => EXEC_MSGS_SEG(Ec2msg, Result, RcvSRCSDU, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU }	C2Msg_Rcv
68	C2Msg_Rcv	SRC_Send_Frame.cnf { } => (none)	C2Msg_Rcv
69	C2Msg_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot)) SRCSDU:= BUILD_PDU_SYNC() SRC_Send_Frame.req {SRCSDU} V(Nslave):= 0 START_TIMER(T(Tslot), V(Tslot(V(Nslave))))	Sync
70	C2Msg_Rcv	DL-WRITE-DATA.req(SAP_ID, DLSDU) => result:= SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	C2Msg_Rcv
71	C2Msg_Rcv	DL-READ-DATA.req(SAP_ID) => result:= GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	C2Msg_Rcv

6.2.3.2.2.2 Maître C2

La Figure 13 et le Tableau 51 montrent le schéma d'états et la table d'états du maître C2 adoptant des intervalles de temps configurables.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

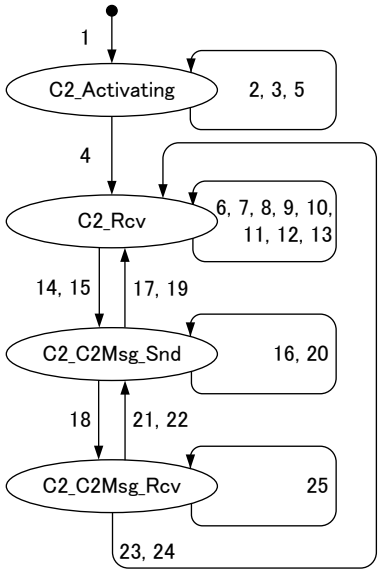


Figure 13 – Schéma d'états du maître C2 utilisant des intervalles de temps configurables

Tableau 51 – Table d'états du maître C2 utilisant des intervalles de temps configurables

N°	État courant	Événement /condition =>action	État suivant
1	N'importe quel état	Power on or CTC_Reset.req => (none)	C2_Activating
2	C2_Activating	CTC_Set_Par.req { Par_ID, Val } => CTC_SET_PAR(Par_ID, Val, Result) CTC_Set_Par.req { Result }	C2_Activating
3	C2_Activating	CTC_Get_Par.req { Par_ID } => CTC_GET_PAR(Par_ID, Val, Result) CTC_Get_Par.cnf { Result, Par_ID, Val }	C2_Activating
4	C2_Activating	CTC_Start.req => START_TIMER(T(Tcycle), V(Tcycle)) START_TIMER(T(Tmsg), V(Tmsg)) CTC_Start.cnf	C2_Rcv
5	C2_Activating	DL-READ-DATA.req(SAP_ID) => result:= GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	C2_Activating
6	C2_Rcv	SRC_Recv_Frame.ind { Result, SRCSDU } / (SRCSDU.TYPLEN = TYP1) => DL_Event.ind {DL_Ev_Tcycle } RESTART_TIMER(T(Tcycle)) RESTART_TIMER(T(Tmsg))	C2_Rcv
7	C2_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tcycle)) STOP_TIMER(T(Tmsg))	C2_Rcv

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition =>action	État suivant
		START_TIMER(T(Tcycle), V(Tcycle)) START_TIMER(T(Tmsg), V(Tmsg))	
8	C2_Rcv	SRC_Recv_Frame.ind { Result, RcvSRCSDU } / RcvSRCSDU. SA = C1 RcvSRCSDU. TYPLEN = TYP2 => V(Nslave):= EX_ADD(DA) STORE_PDU_CYCC(V(Nslave), SRCSDU, Result)	C2_Rcv
9	C2_Rcv	SRC_Recv_Frame.ind { Result, RcvSRCSDU } / RcvSRCSDU. SA = Slave RcvSRCSDU. TYPLEN = TYP2 => V(Nslave):= EX_ADD(DA) STORE_PDU_CYCR(V(Nslave), SRCSDU, Result)	C2_Rcv
10	C2_Rcv	SRC_Recv_Frame.ind { Result, RcvSRCSDU } / SRCSDU. SA = C1 SRCSDU.TYPLEN = TYP12 => EXEC_MSGS_SEG(Ec1msg, Result, RcvSRCSDU, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU }	C2_Rcv
11	C2_Rcv	SRC_Send_Frame.cnf { } => (none)	C2_Rcv
12	C2_Rcv	EXPIRED_TIMER (T(Tmsg)) = "True" / CHECK_MSG_EN(C2) = 0 => (none)	C2_Rcv
13	C2_Rcv	DL-READ-DATA.req(SAP_ID) => result:= GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	C2_Rcv
14	C2_Rcv	SRC_Recv_Frame.ind { Result, RcvSRCSDU } / RcvSRCSDU. SA = V(MA) RcvSRCSDU. TYPLEN = TYP5 => EXEC_MSGC_SEG(Ec2msg, OK, None, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU } slot:= GET_MSG_SLOT(C1) STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot),slot))	C2_C2Msg_Snd
15	C2_Rcv	EXPIRED_TIMER (T(Tmsg)) = "True" / CHECK_MSG_EN(C2) <> 0 => EXEC_MSGC_SEG(Ec2msg, OK, None, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU } slot:= GET_MSG_SLOT(C1) STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot),slot))	C2_C2Msg_Snd
16	C2_C2Msg_Snd	EXPIRED_TIMER (T(Tslot)) = "True" / CHECK_MSG_EN(C2) <> 0 => EXEC_MSGC_SEG(Ec2msg, OK, None, SndSRCSDU)	C2_C2Msg_Snd

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition =>action	État suivant
		SRC_Send_Frame.req { SndSRCSDU } slot:= GET_MSG_SLOT(C1) STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot),slot))	
17	C2_C2Msg_Snd	EXPIRED_TIMER (T(Tslot)) = "True" / CHECK_MSG_EN(C2) = 0 => (none)	C2_Rcv
18	C2_C2Msg_Snd	SRC_Send_Frame.cnf { } => (none)	C2_C2Msg_Rcv
19	C2_C2Msg_Snd	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot)) STOP_TIMER(T(Tcycle)) STOP_TIMER(T(Tmsg)) START_TIMER(T(Tcycle), V(Tcycle)) START_TIMER(T(Tmsg), V(Tmsg))	C2_Rcv
20	C2_C2Msg_Snd	DL-READ-DATA.req(SAP_ID) => result:= GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	C2_C2Msg_Snd
21	C2_C2Msg_Rcv	SRC_Recv_Frame.ind { Result, RcvSRCSDU } => EXEC_MSGC_SEG(Ec2msg, Result, RcvSRCSDU, None)	C2_C2Msg_Snd
22	C2_C2Msg_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / CHECK_MSG_EN(C2) <> 0 => EXEC_MSGC_SEG(Ec2msg, OK, None, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU } slot:= GET_MSG_SLOT(C1) STOP_TIMER(T(Tslot)) START_TIMER(T(Tslot),slot))	C2_C2Msg_Snd
23	C2_C2Msg_Rcv	EXPIRED_TIMER (T(Tslot)) = "True" / CHECK_MSG_EN(C2) = 0 => (none)	C2_Rcv
24	C2_C2Msg_Rcv	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle } STOP_TIMER(T(Tslot)) STOP_TIMER(T(Tcycle)) STOP_TIMER(T(Tmsg)) START_TIMER(T(Tcycle), V(Tcycle)) START_TIMER(T(Tmsg), V(Tmsg))	C2_Rcv
25	C2_C2Msg_Rcv	DL-READ-DATA.req(SAP_ID) => result:= GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	C2_C2Msg_Rcv

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

6.2.3.2.2.3 Esclave

La Figure 14 et le Tableau 52 montrent le schéma d'états et la table d'états de l'esclave adoptant des intervalles de temps configurables.

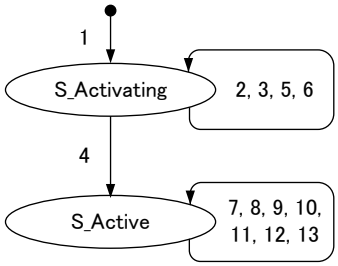


Figure 14 – Schéma d'états de l'esclave utilisant des intervalles de temps configurables

Tableau 52 – Table d'états de l'esclave utilisant des intervalles de temps configurables

N°	État courant	Événement /condition => action	État suivant
1	N'importe quel état	Power on or CTC_Reset.req => (none)	S_Activating
2	S_Activating	CTC_Set_Par.req { Par_ID, Val } => CTC_SET_PAR(Par_ID, Val, Result) CTC_Set_Par.req { Result }	S_Activating
3	S_Activating	CTC_Get_Par.req { Par_ID } => CTC_GET_PAR(Par_ID, Val, Result) CTC_Get_Par.cnf { Result, Par_ID, Val }	S_Activating
4	S_Activating	CTC_Start.req => START_TIMER(T(Tcycle), V(Tcycle)) CTC_Start.cnf	S_Active
5	S_Activating	DL-WRITE-DATA.req(SAP_ID, DLSDU) => result:= SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	S_Activating
6	S_Activating	DL-READ-DATA.req(SAP_ID) => result:= GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	S_Activating
7	S_Active	EXPIRED_TIMER (T(Tcycle)) = "True" => DL_Event.ind {DL_Ev_Tcycle }	S_Active
8	S_Active	SRC_Recv_Frame.ind { Result, RcvSRCSDU } / (SRCSDU.TYPLEN = TYP1) => DL_Event.ind {DL_Ev_Tcycle } RESTART_TIMER(T(Tcycle))	S_Active
9	S_Active	SRC_Recv_Frame.ind { Result, RcvSRCSDU } / RcvSRCSDU.DA = V(MA) && RcvSRCSDU. TYPLEN = TYP2 => STORE_PDU_CYCC(1, SRCSDU, Result) SndSRCSDU:= BUILD_PDU_CYCR(V(MA)) SRC_Send_Frame.req { SndSRCSDU}	S_Active

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
10	S_Active	SRC_Recv_Frame.ind { Result, RcvSRCSDU } / RcvSRCSDU.DA = V(MA) RcvSRCSDU.SA = C1 && RcvSRCSDU. TYPLEN = TYP12 => EXEC_MSGS_SEG(Ec1msg, Result, RcvSRCSDU, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU }	S_Active
11	S_Active	SRC_Recv_Frame.ind { Result, RcvSRCSDU } / RcvSRCSDU.DA = V(MA) RcvSRCSDU.SA = C2 && RcvSRCSDU. TYPLEN = TYP12 => EXEC_MSGS_SEG(Ec2msg, Result, RcvSRCSDU, SndSRCSDU) SRC_Send_Frame.req { SndSRCSDU }	S_Active
12	S_Active	DL-WRITE-DATA.req(SAP_ID, DLSDU) => result:= SET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-WRITE-DATA.cnf (Result)	S_Active
13	S_Active	DL-READ-DATA.req(SAP_ID) => result:= GET_CYC_DATA(SAP_ID, Node_ID, DLSDU) DL-READ-DATA.cnf (Result, DLSDU)	S_Active

6.2.3.2.3 Fonctions utilisées par la machine de transmission cyclique

Les fonctions utilisées par la machine protocolaire à intervalles de temps de largeur fixe sont résumées dans le Tableau 53.

Tableau 53 – Liste des fonctions utilisées par la machine de transmission cyclique

Nom de la fonction	Paramètre		Renvoi	Fonctionnement
	Entrée	Sortie	Valeur	
BUILD_PDU_SYNC	none	Aucun	SRCSDU	Cette fonction construit la SRCSDU servant à demander à la SRC d'envoyer la trame synchrone.
BUILD_PDU_OUT	Slave_index, DLSDU	Aucun	SRCSDU	Cette fonction construit la SRCSDU servant à demander à la SRC d'émettre la trame de données. Slave_Index est le numéro d'esclave, prenant des valeurs comprises entre 1 et V(Nmax_slave).
BUILD_PDU_IN	Slave_index, DLSDU	Aucun	SRCSDU	Cette fonction construit la SRCSDU servant à demander à la SRC de recevoir la trame d'entrée. Slave_Index est le numéro d'esclave, prenant des valeurs comprises entre 1 et V(Nmax_slave).
STORE_PDU_OUT	Slave_index, SRCSDU, Rcv_Sts	Aucun	Aucun	Cette fonction permet de stocker la SRCSDU récupérée de la SRC et de recevoir le statut Rcv_Sts comme donnée de sortie reçue. Slave_Index est le numéro d'esclave, prenant des valeurs comprises entre 1 et V(Nmax_slave).
STORE_PDU_IN	Slave_index, SRCSDU,	Aucun	Aucun	Cette fonction permet de stocker la SRCSDU récupérée de la SRC et de

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Nom de la fonction	Paramètre		Renvoi	Fonctionnement
	Entrée	Sortie	Valeur	
	Rcv_Sts			recevoir le statut Rcv_Sts comme donnée d'entrée reçue. Slave_Index est le numéro d'esclave, prenant des valeurs comprises entre 1 et V(Nmax_slave).
PUT_RETRY_LIST	Slave_index	Aucun	Aucun	Cette fonction inscrit le numéro d'esclave spécifié par Slave_index dans la liste des nouvelles tentatives et incrémente la valeur de V(Nretry_list), qui contient le numéro de l'esclave inscrit.
GET_RETRY_LIST	aucun	Aucun	Slave_index	Cette fonction récupère un numéro d'esclave dans la liste des nouvelles tentatives et décrémente la valeur de V(Nretry_list), qui contient le numéro de l'esclave inscrit.
EXEC_MSGC_SEG	Fc1c2, Rcv_sts, RcvSRCSDU	SndSRCSDU	aucun	Cette fonction appelle la machine de segmentation de messages pour l'initiateur de la communication de messages. Le paramètre Fc1c2 est un indicateur spécifiant s'il s'agit du message C1 ou du message C2. Rcv_sts et RcvSRCSDU sont respectivement le statut de réception et la SRCSDU reçue, qui ont été passés par la SRC. SndSRCSDU est la SRCSDU à envoyer.
EXEC_MSGS_SEG	Fc1c2, Rcv_sts, RcvSRCSDU	SndSRCSDU	aucun	Cette fonction appelle la machine de segmentation de messages pour le répondeur de la communication de messages. Les paramètres de cette fonction sont les mêmes que ceux de la fonction EXEC_MSGC_SEG.
EXEC_MSPM_IR	Ec1c2, Rcv_sts, RcvSRCSDU	Aucun	Aucun	Cette fonction appelle la machine protocolaire de segmentation de messages pour que l'initiateur de la communication de messages traite une trame reçue. Les paramètres de cette fonction sont les mêmes que les paramètres de même nom de la fonction EXEC_MSPM_R.
START_TIMER	Tim_ID, Val	Aucun		Ce fonction lance le temporisateur spécifié par Tim_ID dans la valeur Val définie. Le temporisateur est de type à rechargement automatique: il recharge la valeur définie lorsque la temporisation est écoulée.
STOP_TIMER	Tim_ID			Cette fonction arrête le temporisateur spécifié par Tim_ID.
RESTART_TIMER	Tim_ID			Ce fonction relance le temporisateur spécifié par Tim_ID avec la valeur définie spécifiée par START_TIMER.
EXPIRED_TIMER	Tim_ID			Cette fonction indique le statut du temporisateur spécifié par Tim_ID.
CTC_SET_PAR	Var_ID, Val			Cette fonction met à jour la variable spécifiée par Var_ID dans la DLE avec la valeur spécifiée par Val.
CTC_GET_PAR	Var_ID	Val		Cette fonction lit la valeur courante de la variable spécifiée par Var_ID dans la DLE.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Nom de la fonction	Paramètre		Renvoi	Fonctionnement
	Entrée	Sortie	Valeur	
SET_CYC_DATA	SRCSDU			Cette fonction met à jour le tampon d'envoi pour l'échange de données d'E/S existant dans la CTC avec les données spécifiées.
GET_CYC_DATA	SRCSDU			Cette fonction met à jour le tampon de réception pour l'échange de données d'E/S existant dans la CTC avec les données spécifiées.
CMP_RTIM	Tim_ID, Val			Ce fonction compare le temps restant du temporisateur spécifié par Tim_ID à la valeur Val définie. Si le temps restant est important, True (vrai) est renvoyé et, s'il est faible, False (faux) est renvoyé.
CHECK_MSG_EN	Msg_mst	Enable_flag		Cette fonction détermine si la communication de messages que la station spécifiée par Msg_mst actionne en tant qu'initiateur peut être exécutée. Si les trois conditions suivantes sont satisfaites, l'indicateur d'autorisation enable_flag est activé: - La bande de communication de messages est configurée; - L'utilisateur de DLS a émis une demande; - Il reste du temps pour la communication de messages.
GET_MSG_SLOT	Msg_mst	msg_slot		Cette fonction émet la période d'intervalle de temps pour la communication de messages que la station spécifiée par Msg_mst actionne en tant qu'initiateur.
EX_ADD	Node_address	Slave_index		Cette fonction convertit l'adresse de la station en numéro d'esclave.
SND_MSG_TOKEN				Cette fonction émet la trame de jeton s'il reste du temps grâce auquel la trame de jeton de message peut être émise avant l'instant de début de message C2.

6.2.3.3 Machine protocolaire de segmentation de message

6.2.3.3.1 Généralités

La spécification de segmentation et d'assemblage de messages de la machine de segmentation de messages possède deux sortes de formats de trame. Elles sont respectivement définies dans le paragraphe suivant.

Dans la communication de messages, elles sont classées en deux catégories selon la bande allouée. L'une est la communication de messages C1, dans laquelle le maître C1 fonctionne en tant que station principale (initiateur) et l'esclave ou le maître C2 en tant que station secondaire (répondeur). L'autre est la communication de messages C2, dans laquelle le maître C2 fonctionne en tant que station principale, l'esclave ou le maître C1 devenant une station secondaire. La station principale et la station secondaire fonctionnent de la même façon, bien qu'elles utilisent des bandes allouées différentes.

La machine protocolaire de segmentation de messages est exécutée par la machine protocolaire de transmission cyclique avec l'appel de fonction dont le nom est préfixé par

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

"EXEC_MSPM_". Dans la table d'états du paragraphe suivant, l'événement "Appel" désigne les appels de fonction.

6.2.3.3.2 Segmentation pour la DLPDU au format de base

6.2.3.3.2.1 Initiateur pour la DLPDU au format de base

La Figure 15 et le Tableau 54 montrent le schéma d'états et la table d'états de l'initiateur de la machine de segmentation de messages lorsque la DLE adopte une DLPDU au format de base.

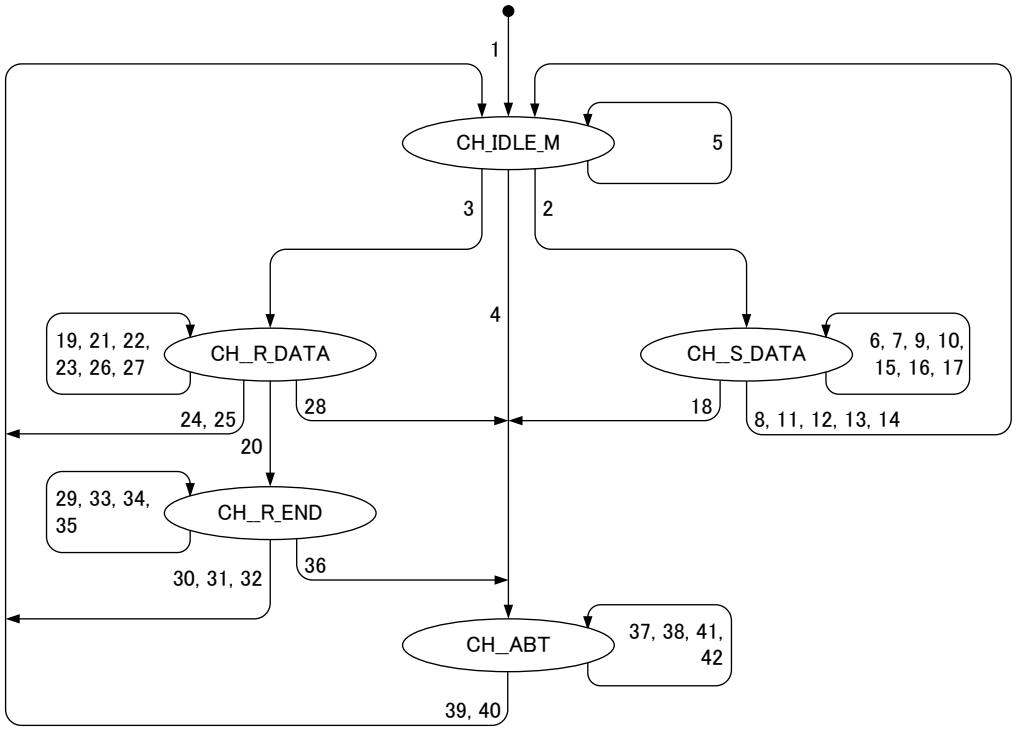


Figure 15 – Schéma d'états de l'initiateur de message utilisant le format de base

Tableau 54 – Table d'états de l'initiateur de message utilisant le format de base

N°	État courant	Événement /condition => action	État suivant
1	N'importe quel état	Power on or CTC_Reset.req => V(Nms_n):= 0 V(Nmr_n):= 0	CH_IDLE_M
2	CH_IDLE_M	DL-SDA.req{SAP_ID,Node_ID,Length,SndDLSDU} => V(Nms_n):= 0 STORE_DLSDU(Ec1c2, Node_ID, SndDLSDU) V(Nmsg_len_n):= Length V(Nmsg_rem_len_n):= Length V(Fmsg_sending_n):= True	CH_S_DATA
3	CH_IDLE_M	EXEC_MSPM_IS(Ec1c2, SndSRCSDU) => V(Nmr_n):= 0 V(Fmsg_sending_n):= False V(Nmp_n):= 1 SndSRCSDU:= BUILD_PDU_BMSG(MF_S_RR, 0, V(Nmr_n), V(Nmp_n))	CH_R_DATA
4	CH_IDLE_M	DL-ABORT-SDA.req { SAP_ID } => V(Fmsg_sending_n):= False	CH_ABT

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
5	CH_S_DATA	EXEC_MSPM_IS(Ec1c2, SndSRCSDU) / V(Npkt_len_n) >= V(Nmsg_rem_len_n) => V(Nmp_n) = 1 SndSRCSDU:= BUILD_PDU_BMSG(MF_I, V(Nms_n), V(Nmr_n), V(Nmp_n))	CH_S_DATA
6	CH_S_DATA	EXEC_MSPM_IS(Ec1c2, SndSRCSDU) / V(Npkt_len_n) < V(Nmsg_rem_len_n) => V(Nmp_n) = 0 SndSRCSDU:= BUILD_PDU_BMSG(MF_I, V(Nms_n), V(Nmr_n),V(Nmp_n)) V(Nmsg_rem_len_n):= V(Nmsg_rem_len_n) - V(Npkt_len_n)	CH_S_DATA
7	CH_S_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RR && GET_MC_NR(RcvSRCSDU) = V(Nms_n)+1 && V(Nmp_n) = 1 => V(Nms_n):= V(Nms_n) + 1 DL-SDA.cnf{ SND_OK }	CH_IDLE_M
8	CH_S_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RR && GET_MC_NR(RcvSRCSDU) = V(Nms_n)+1 && V(Nmp_n) = 0 => V(Nms_n):= V(Nms_n) + 1	CH_S_DATA
9	CH_S_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RR && GET_MC_NR(RcvSRCSDU) <> V(Nms_n)+1 => (none)	CH_S_DATA
10	CH_S_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RNR && GET_MC_NR(RcvSRCSDU) = V(Nms_n)+1 && V(Nmp_n) = 1 => V(Nms_n):= V(Nms_n) + 1 DL-SDA.cnf{ SND_OK }	CH_IDLE_M
11	CH_S_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RNR && GET_MC_NR(RcvSRCSDU) = V(Nms_n)+1 && V(Nmp_n) = 0 => DL-SDA.cnf{ SND_BUSY }	CH_IDLE_M
12	CH_S_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RNR && GET_MC_NR(RcvSRCSDU) <> V(Nms_n)+1 => DL-SDA.cnf{ SND_BUSY }	CH_IDLE_M
13	CH_S_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG	CH_IDLE_M

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		&& GET_MC_FMT(RcvSRCSDU) = MF_S_REJ => V(Nms_n):= 0, V(Nmr_n):= 0 DL-SDA.cnf{ SND_ABT }	
14	CH_S_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I => (none)	CH_S_DATA
15	CH_S_DATA	DL-ABORT-SDA.req { SAP_ID } => (none)	CH_ABT
16	CH_R_DATA	EXEC_MSPM_IS(Ec1c2, SndSRCSDU) => V(Nmp_n):= 1 SndSRCSDU:= BUILD_PDU_BMSG(MF_S_RR,0,V(Nmr_n),V(Nmp_n))	CH_R_DATA
17	CH_R_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I && GET_MC_NS(RcvSRCSDU) = V(Nmr_n) && GET_MC_F(RcvSRCSDU) = 1 => V(Nmr_n):= V(Nmr_n) + 1 STORE_PDU_MSG(Ec1c2, RcvSRCSDU)	CH_R_END
18	CH_R_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I && GET_MC_NS(RcvSRCSDU) = V(Nmr_n) && GET_MC_F(RcvSRCSDU) = 0 => V(Nmr_n):= V(Nmr_n) + 1 STORE_PDU_BMSG(RcvSRCSDU)	CH_R_DATA
19	CH_R_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I && GET_MC_NS(RcvSRCSDU) <> V(Nmr_n) => (none)	CH_R_DATA
20	CH_R_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG) && { (GET_MC_FMT(RcvSRCSDU) = MF_S_RR) || (GET_MC_FMT(RcvSRCSDU) = MF_S_RNR) } => (none)	CH_IDLE_M
21	CH_R_DATA	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_REJ => V(Nms_n):= 0, V(Nmr_n):= 0	CH_IDLE_M
22	CH_R_DATA	DL-ABORT-SDA.req { SAP_ID } => (none)	CH_ABT
23	CH_R_END	EXEC_MSPM_IS(Ec1c2, SndSRCSDU) => V(Nmp_n) = 1 SndSRCSDU:= BUILD_PDU_BMSG(MF_S_RR,0,V(Nmr_n),V(Nmp_n))	CH_R_END
24	CH_R_END	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU)	CH_IDLE_M

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		/ Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && { (GET_MC_FMT(RcvSRCSDU) = MF_S_RR) || (GET_MC_FMT(RcvSRCSDU) = MF_S_RNR) } => Node_ID:= GET_SA(RcvSRCSDU) DLSDU:= GET_DLSDU(Ec1c2) Length:= GET_DLSDU_LEN(DLSDU) DL-SDA.ind{ SAP_ID, Rcv_sts , Node_ID, Length, DLSDU}	
25	CH_R_END	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_REJ => V(Nms_n):= 0, V(Nmr_n):= 0	CH_IDLE_M
26	CH_R_END	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I => (none)	CH_R_END
27	CH_R_END	DL-ABORT-SDA.req { SAP_ID } => (none)	CH_ABT
28	CH_ABT	EXEC_MSPM_IS(Ec1c2, SndSRCSDU) => V(Nmp_n) = 1 SndSRCSDU:= BUILD_PDU_BMSG(MF_S_REJ,0,0,V(Nmp_n))	CH_ABT
29	CH_ABT	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && { (GET_MC_FMT(RcvSRCSDU) = MF_S_RR) || (GET_MC_FMT(RcvSRCSDU) = MF_I) } => (none)	CH_ABT

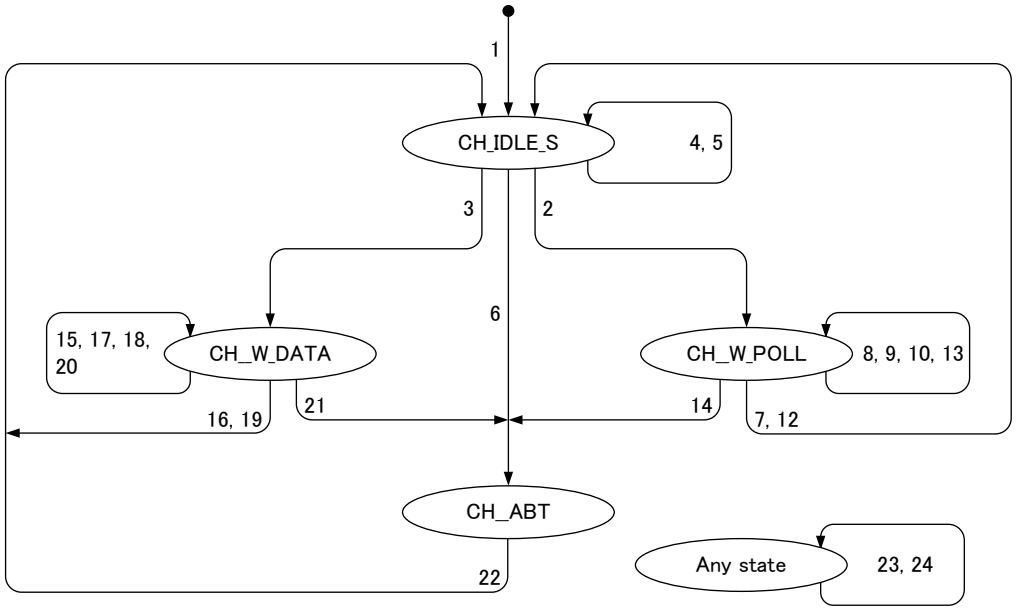
30	CH_ABT	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU)
		/ Rcv_sts = OK
		&& GET_FTYP(RcvSRCSDU) = FT_MSG
		&& { (GET_MC_FMT(RcvSRCSDU) = MF_S_RNR) ||
		(GET_MC_FMT(RcvSRCSDU) = MF_S_REJ) }
		=> V(Nms_n):= 0, V(Nmr_n):= 0
		DL-ABORT-SDA.cnf
		if (V(Fmsg_sending_n) = True) then
		DL-SDA.cnf { SND_ABT }
		endif
			CH_IDLE_M

31	N'importe quel état	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) <> FT_MSG => (none)	Même état
32	N'importe quel état	EXEC_MSPM_IR(Ec1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts <> OK => (none)	Même état

6.2.3.3.2.2 Répondeur pour la DLPDU au format de base

La Figure 16 et le Tableau 55 montrent le schéma d'états et la table d'états du répondeur de la machine de segmentation de messages lorsque la DLE adopte une DLPDU au format de base.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe



Légende

Anglais	Français
Any state	N'importe quel état

Figure 16 – Schéma d'états du répondeur de message utilisant le format de base

Tableau 55 – Table d'états du répondeur de message utilisant le format de base

N°	État courant	Événement /condition => action	État suivant
1	N'importe quel état	Power on or CTC_Reset.req => V(Nms_n):= 0 V(Nmr_n):= 0	CH_IDLE_S
2	CH_IDLE_S	DL-SDA.req{SAP_ID,Node_ID,Length,SndDLSDU} => V(Nms_n):= 0 STORE_DLSDU(Ec1c2, Node_ID, SndDLSDU) V(Nmsg_len_n):= Length V(Nmsg_rem_len_n):= Length V(Fmsg_sending_n):= True	CH_W_POLL
3	CH_IDLE_S	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I && GET_MC_NS(RcvSRCSDU) = 0 && GET_MCTL(RcvSRCSDU,MCTL_P) = 0 => V(Nmr_n):= 1 V(Fmsg_sending_n):= False STORE_PDU_BMSG(Ec1c2, RcvSRCSDU) V(Nmr_n):= 1 V(Nmf_n) = 1 SndSRCSDU:= BUILD_PDU_BMSG(MF_S_RR,0,V(Nmr_n),V(Nmf_n))	CH_W_DATA
4	CH_IDLE_S	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I && GET_MC_NS(RcvSRCSDU) = 0 && GET_MCTL(RcvSRCSDU,MCTL_P) = 1 => V(Nmr_n):= 1 V(Nmf_n) = 1	CH_IDLE_S

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		SndSRCSDU:= BUILD_PDU_BMSG(MF_S_RNR,0,V(Nmr_n),V(Nmf_n)) STORE_PDU_MSG(Ec1c2, RcvSRCSDU) Node_ID:= GET_SA(RcvSRCSDU) DLSDU:= GET_DLSDU(Ec1c2) Length:= GET_DLSDU_LEN(DLSDU) DL-SDA.ind{ SAP_ID, Rcv_sts , Node_ID, Length, DLSDU}	
5	CH_IDLE_S	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && CHK_MCTL_IS(RcvSRCSDU,MF_I) <> OK => V(Nmf_n) = 1 SndSRCSDU:= BUILD_PDU_BMSG(MF_S_RNR,0,V(Nmr_n),V(Nmf_n))	CH_IDLE_S
6	CH_IDLE_S	DL-ABORT-SDA.req { SAP_ID } => V(Fmsg_sending_n):= False	CH_ABT
7	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RR && GET_MC_NR(RcvSRCSDU) = V(Nms_n) && V(Nmsg_rem_len_n) = 0 => V(Nmf_n) = 1 SndSRCSDU:= BUILD_PDU_BMSG(MF_S_RNR,0,V(Nmr_n),V(Nmf_n)) DL-SDA.cnf{ SND_OK }	CH_IDLE_S
8	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RR && GET_MC_NR(RcvSRCSDU) = V(Nms_n) && V(Npkt_len_n) >= V(Nmsg_rem_len_n) => V(Nmf_n) = 1 SndSRCSDU:= BUILD_PDU_BMSG(MF_I,V(Nms_n),V(Nmr_n),V(Nmf_n)) V(Nmsg_rem_len_n):= 0	CH_W_POLL
9	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RR && GET_MC_NR(RcvSRCSDU) = V(Nms_n) && V(Npkt_len_n) < V(Nmsg_rem_len_n) => V(Nmf_n) = 0 SndSRCSDU:= BUILD_PDU_BMSG(MF_I,V(Nms_n),V(Nmr_n),V(Nmf_n)) V(Nms_n) = V(Nms_n) + 1 V(Nmsg_rem_len_n):= V(Nmsg_rem_len_n) - V(Npkt_len_n)	CH_W_POLL
10	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RR && GET_MC_NR(RcvSRCSDU) <> V(Nms_n) => SndSRCSDU:= BUILD_PDU_BMSG(MF_I,V(Nms_n)-1,V(Nmr_n),V(Nmf_n))	CH_W_POLL
11	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RNR => SndSRCSDU:=	CH_W_POLL

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		BUILD_PDU_BMSG(MF_I,V(Nms_n)-1,V(Nmr_n),V(Nmf_n))	
12	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_REJ => V(Nms_n):= 0, V(Nmr_n):= 0, V(Nmf_n) = 1 DL-SDA.cnf{ SND_ABT } SndSRCSDU:= BUILD_PDU_BMSG(MF_S_RNR,0,V(Nmr_n),V(Nmf_n))	CH_IDLE_S
13	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I => V(Nmf) = 1 SndSRCSDU:= BUILD_PDU_BMSG(MF_S_RR,0,V(Nmr_n),V(Nmf_n))	CH_W_POLL
14	CH_W_POLL	DL-ABORT-SDA.req { SAP_ID } => (none)	CH_ABT
15	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I && GET_MC_NS(RcvSRCSDU) = V(Nmr_n) && GET_MCTL(RcvSRCSDU,MCTL_P) = 0 => STORE_PDU_BMSG(Ec1c2, RcvSRCSDU) V(Nmr_n):= V(Nmr_n) + 1 V(Nmf_n) = 1 SndSRCSDU:= BUILD_PDU_MSG(MF_S_RR,0,V(Nmr_n),V(Nmf_n))	CH_W_DATA
16	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I && GET_MC_NS(RcvSRCSDU) = V(Nmr_n) && GET_MCTL(RcvSRCSDU,MCTL_P) = 1 => V(Nmr_n):= V(Nmr_n) + 1 V(Nmf_n) = 1 SndSRCSDU:= BUILD_PDU_MSG(MF_S_RNR,0,V(Nmr_n),V(Nmf_n)) STORE_PDU_MSG(Ec1c2, RcvSRCSDU) Node_ID:= GET_SA(RcvSRCSDU) DLSDU:= GET_DLSDU(Ec1c2) Length:= GET_DLSDU_LEN(DLSDU) DL-SDA.ind{ SAP_ID, Rcv_sts , Node_ID, Length, DLSDU}	CH_IDLE_S
17	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_I && GET_MC_NS(RcvSRCSDU) <> V(Nmr_n) =>V(Nmf_n) = 1 SndSRCSDU:= BUILD_PDU_MSG(MF_S_RR,0,V(Nmr_n),V(Nmf_n))	CH_W_DATA
18	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RR => V(Nmf_n) = 1 SndSRCSDU:= BUILD_PDU_MSG(MF_S_RR,0,V(Nmr_n),V(Nmf_n))	CH_W_DATA

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
19	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_REJ => V(Nms_n):= 0, V(Nmr_n):= 0, V(Nmf_n) = 1 SndSRCSDU:= BUILD_PDU_MSG(MF_S_RNR),0,V(Nmr_n),V(Nmf_n))	CH_IDLE_S
20	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) = FT_MSG && GET_MC_FMT(RcvSRCSDU) = MF_S_RNR => V(Nms_n):= 0, V(Nmr_n):= 0, V(Nmf_n) = 1 SndSRCSDU:= BUILD_PDU_MSG(MF_S_RR),0,V(Nmr_n),V(Nmf_n))	CH_IDLE_S
21	CH_W_DATA	DL-ABORT-SDA.req { SAP_ID } => (none)	CH_ABT

22	CH_ABT	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU)
		/ Rcv_sts = OK
		=> V(Nms_n):= 0, V(Nmr_n):= 0, V(Nmf_n) = 1
		DL-ABORT-SDA.cnf
		if (V(Fmsg_sending_n) = True) then
		DL-SDA.cnf{SND_ABT}
		endif
		SndSRCSDU:= BUILD_PDU_MSG(MF_S_REJ,0,0,V(Nmf_n))
			CH_IDLE_S

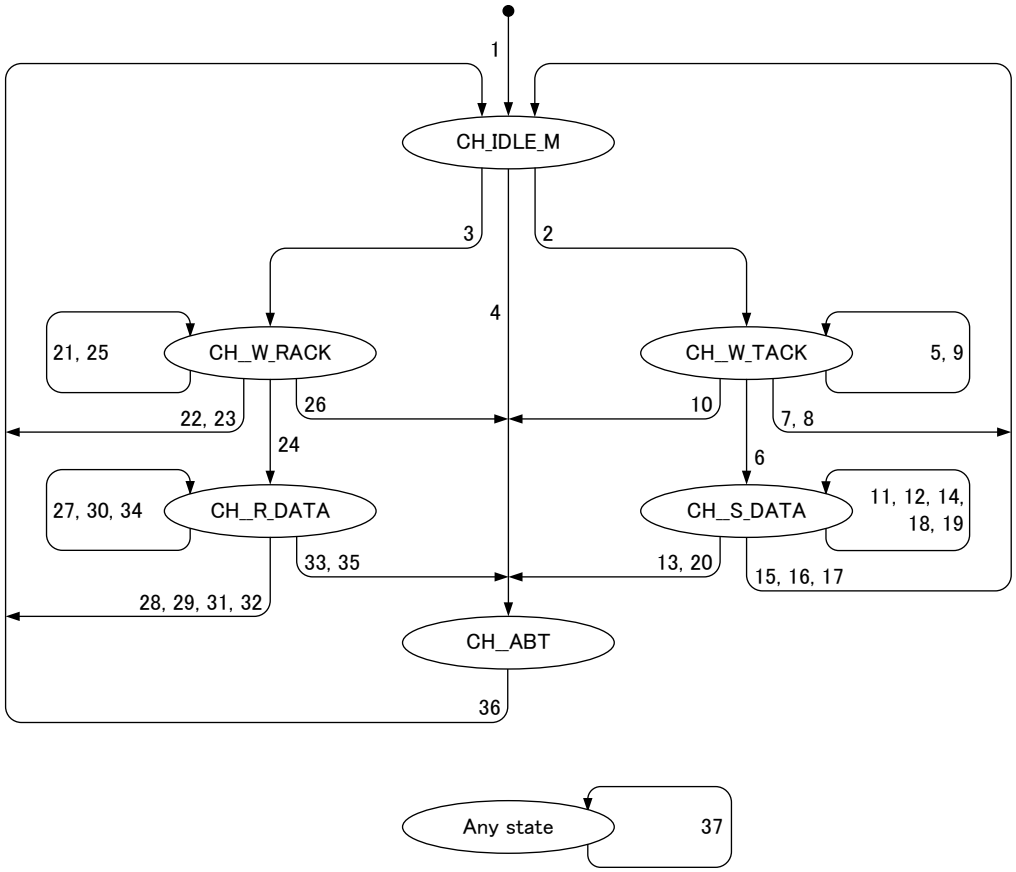
23	N'importe quel état	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && GET_FTYP(RcvSRCSDU) <> FT_MSG => (none)	Same state
24	N'importe quel état	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts <> OK => (none)	Same state

6.2.3.3.3 Segmentation pour la DLPDU de format court

6.2.3.3.3.1 Initiateur pour la DLPDU de format court

La Figure 17 et le Tableau 56 montrent le schéma d'états et la table d'états de l'initiateur de la machine de segmentation de messages lorsque la DLE adopte une DLPDU de format court.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe



Légende

Anglais	Français
Any state	N'importe quel état

Figure 17 – Schéma d'états de l'initiateur de message utilisant le format court

Tableau 56 – Table d'états de l'initiateur de message utilisant le format court

N°	État courant	Événement /condition => action	État suivant
1	N'importe quel état	Power on or CTC_Reset.req => (none)	CH_IDLE_M
2	CH_IDLE_M	DL-SDA.req{SAP_ID,Node_ID,Length,SndDLSDU} => V(Nmno_n) = GET_SMSG_NUM(Length) STORE_DLSDU(Ec1c2, Node_ID, SndDLSDU) V(Fmsg_sending_n):= True	CH_W_TACK
3	CH_IDLE_M	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) => V(Nmsd_n):= 0, V(Nmend_n):= 1, V(Nms_n):= 0 SndSRCSDU:= BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n)) V(Fmsg_sending_n):= False	CH_W_RACK
4	CH_IDLE_M	DL-ABORT-SDA.req{SAP_ID,Node_ID} => V(Fmsg_sending_n):= False	CH_ABT
5	CH_W_TACK	EXEC_MSPM_IS(Fc1c2, SndSRCSDU) => V(Nmsd_n):= 0, V(Nmend_n):= 0, V(Nms_n):= 0 SndSRCSDU:= BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_W_TACK
6	CH_W_TACK	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0	CH_S_DATA

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		&& CHK_END(RcvSRCSDU) = 0 => (none)	
7	CH_W_TACK	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = ABT_NUM => DL-SDA.cnf{ SND_ABT }	CH_IDLE_M
8	CH_W_TACK	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) <> ABT_NUM => DL-SDA.cnf{ SND_ERR }	CH_IDLE_M
9	CH_W_TACK	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1) => (none)	CH_W_TACK
10	CH_W_TACK	DL-ABORT-SDA.req{SAP_ID,Node_ID} => (none)	CH_ABT
11	CH_S_DATA	EXEC_MSPM_IS(Fc1c2, SndSRCSDU) / (V(Nmno_n)) > 1 => V(Nmsd_n):= 1, V(Nmend_n):= 0 SndSRCSDU:= BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_S_DATA
12	CH_S_DATA	EXEC_MSPM_IS(Fc1c2, SndSRCSDU) / (V(Nmno_n)) = 1 => V(Nmsd_n):= 1, V(Nmend_n):= 1 SndSRCSDU:= BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_S_DATA
13	CH_S_DATA	EXEC_MSPM_IS(Fc1c2, SndSRCSDU) / (V(Nmno)) < 1 => (none)	CH_ABT
14	CH_S_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 0 => (none)	CH_S_DATA
15	CH_S_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = ABT_NUM => DL-SDA.cnf{ SND_ABT }	CH_IDLE_M
16	CH_S_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) <> ABT_NUM => DL-SDA.cnf{ SND_ERR }	CH_IDLE_M
17	CH_S_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) = V(Nms_n) && V(Nmno_n) = 1 => DL-SDA.cnf{ SND_OK }	CH_IDLE_M
18	CH_S_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK	CH_S_DATA

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		&& CHK_SD(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) = V(Nms_n) => V(Nms_n):= V(Nms_n) + 1 V(Nmno_n):= V(Nmno_n) – 1	
19	CH_S_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) <> V(Nms_n) => (none)	CH_S_DATA
20	CH_S_DATA	DL-ABORT-SDA.req{SAP_ID,Node_ID} => (none)	CH_ABT
21	CH_W_RACK	EXEC_MSPM_IS(Fc1c2, SndSRCSDU) => V(Nmsd_n):= 0, V(Nmend_n):= 1, V(Nms_n):= 0 SndSRCSDU:= BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_W_RACK
22	CH_W_RACK	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 0 => (none)	CH_IDLE_M
23	CH_W_RACK	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) = ABT_NUM => (none)	CH_IDLE_M
24	CH_W_RACK	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) <> ABT_NUM => (none)	CH_R_DATA
25	CH_W_RACK	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1) => (none)	CH_W_RACK
26	CH_W_RACK	DL-ABORT-SDA.req{SAP_ID,Node_ID} => (none)	CH_ABT
27	CH_R_DATA	EXEC_MSPM_IS(Fc1c2, SndSRCSDU) => V(Nmsd_n):= 0, V(Nmend_n):= 1 SndSRCSDU:= BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_R_DATA
28	CH_R_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 0 => (none)	CH_IDLE_M
29	CH_R_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) = ABT_NUM => (none)	CH_IDLE_M
30	CH_R_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) <> ABT_NUM	CH_R_DATA

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		=> (none)	
31	CH_R_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) = V(Nms_n) && (CHK_END(RcvSRCSDU) = 1) => Node_ID:= RcvSRCSDU.SA STORE_PDU_MSG(Ec1c2, RcvSRCSDU) DLSDU:= GET_DLSDU(Ec1c2)) Length:= GET_DLSDU_LEN(DLSDU) DL-SDA.ind{ SAP_ID, Rcv_sts , Node_ID, Length, DLSDU}	CH_IDLE_M
32	CH_R_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) = V(Nms_n) && (CHK_END(RcvSRCSDU) = 0) && CHK_MRBUF() = OK => V(Nms_n):= V(Nms_n) + 1 STORE_PDU_MSG(Ec1c2, RcvSRCSDU)	CH_IDLE_M
33	CH_R_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) = V(Nms_n) && (CHK_END(RcvSRCSDU) = 0) && CHK_MRBUF() <> OK => (none)	CH_ABT
34	CH_R_DATA	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1) && CHK_S(RcvSRCSDU) <> V(Nms_n)) => (none)	CH_R_DATA
35	CH_R_DATA	DL-ABORT-SDA.req{SAP_ID,Node_ID} => (none)	CH_ABT

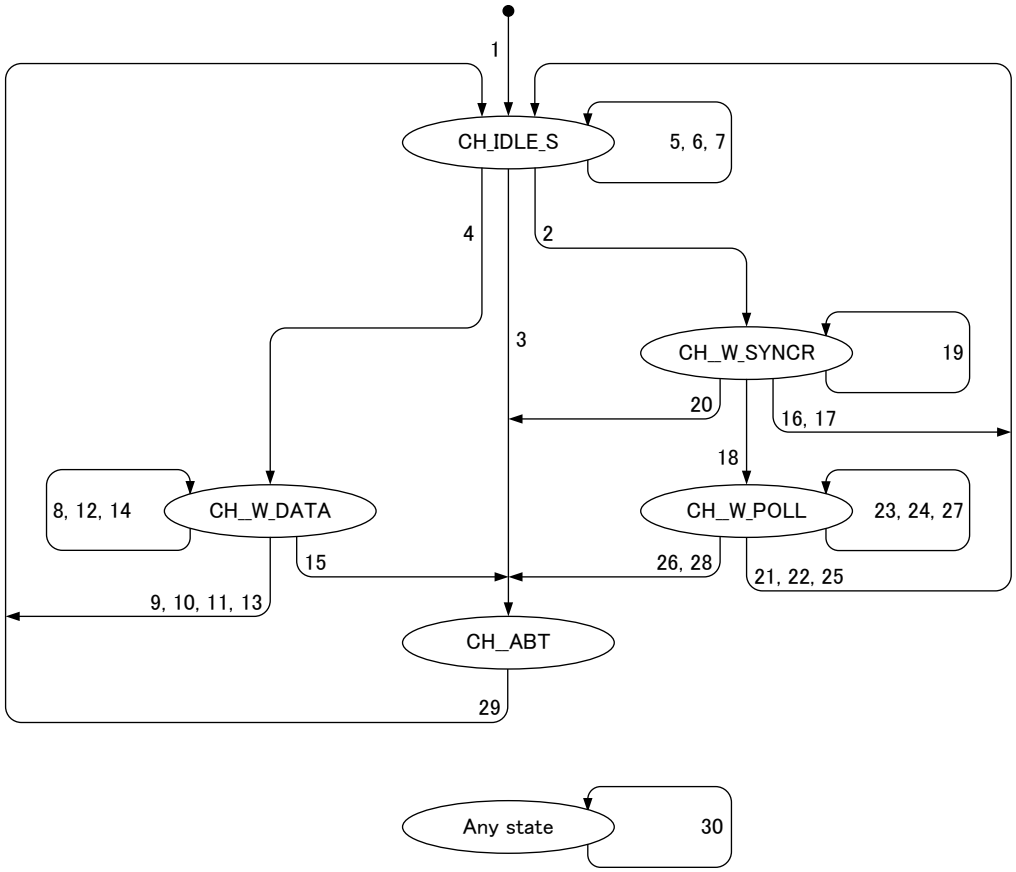
36	CH_ABT	EXEC_MSPM_IS(Fc1c2, SndSRCSDU)
		=> V(Nmsd_n):= 0, V(Nmend_n):= 1, V(Nms_n):= ABT_NUM
		SndSRCSDU:=
		BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))
		DL-ABORT-SDA.cnf
		if (V(Fmsg_sending_n) = True) then
		DL-SDA.cnf{SND_ABT}
		endif
			CH_IDLE_M

37	N'importe quel état	EXEC_MSPM_IR(Fc1c2, Rcv_sts, RcvSRCSDU) / Rcv_sts <> OK => (none)	Même état

6.2.3.3.3.2 Répondeur pour la DLPDU de format court

La Figure 18 et le Tableau 57 montrent le schéma d'états et la table d'états du répondeur de la machine de segmentation de messages lorsque la DLE adopte une DLPDU de format court.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe



Légende

Anglais	Français
Any state	N'importe quel état

Figure 18 – Schéma d'états du répondeur de message utilisant le format court

Tableau 57 – Table d'états du répondeur de message utilisant le format court

N°	État courant	Événement /condition => action	État suivant
1	N'importe quel état	Power on or CTC_Reset.req => (none)	CH_IDLE_S
2	CH_IDLE_S	DL-SDA.req{SAP_ID,Node_ID,Length,SndDLSDU} => V(Nmno) = GET_SMSG_NUM(Length) STORE_DLSDU(Ec1c2, Node_ID, SndDLSDU) V(Fmsg_sending_n):= True	CH_W_SYNCR
3	CH_IDLE_S	DL-ABORT-SDA.req{SAP_ID,Node_ID} => V(Fmsg_sending_n):= False	CH_ABT
4	CH_IDLE_S	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 0 => V(Nmsd_n):= 0, V(Nmend_n):= 0, V(Nms_n):= 0 SndSRCSDU:= BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n)) V(Fmsg_sending_n):= False	CH_W_DATA
5	CH_IDLE_S	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK	CH_IDLE_S

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		&& CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = ABT_NUM => V(Nmsd_n):= 0, V(Nmend_n):= 1, V(Nms_n):= ABT_NUM SndSRCSDU:= BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	
6	CH_IDLE_S	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) <> ABT_NUM => V(Nmsd_n):= 0, V(Nmend_n):= 0, V(Nms_n):= 0 SndSRCSDU:= BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_IDLE_S
7	CH_IDLE_S	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1 => V(Nmsd_n):= 0, V(Nmend_n):= 0, V(Nms_n):= 0 SndSRCSDU:= BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_IDLE_S
8	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 0 => SndSRCSDU:= BUILD_PDU_LAST_SMSG()	CH_W_DATA
9	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = ABT_NUM => V(Nmsd_n):= 0, V(Nmend_n):= 1, V(Nms_n):= ABT_NUM SndSRCSDU:= BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_IDLE_S
10	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) <> ABT_NUM => SndSRCSDU:= BUILD_PDU_LAST_SMSG()	CH_IDLE_S
11	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = V(Nms_n) && CHK_END(RcvSRCSDU) = 1 => V(Nmsd_n):= CHK_SD(RcvSRCSDU) V(Nmend_n):= CHK_END(RcvSRCSDU) V(Nms_n):= CHK_S(RcvSRCSDU) SndSRCSDU:= BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n)) STORE_PDU_MSG(Ec1c2, RcvSRCSDU) Node_ID:= RcvSRCSDU.SA DLSDU:= GET_DLSDU(Ec1c2)) Length:= GET_DLSDU_LEN(DLSDU) DL-SDA.ind{ SAP_ID, Rcv_sts , Node_ID, Length, DLSDU}	CH_IDLE_S
12	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = V(Nms_n) && CHK_END(RcvSRCSDU) = 0	CH_W_DATA

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		&& CHK_MRBUF() = OK => V(Nmsd_n):= CHK_SD(RcvSRCSDU) V(Nmend_n):= CHK_END(RcvSRCSDU) V(Nms_n):= CHK_S(RcvSRCSDU) SndSRCSDU:= BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n)) V(Nms_n):= V(Nms_n) + 1 STORE_PDU_MSG(Ec1c2, RcvSRCSDU)	
13	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = V(Nms_n) && CHK_END(RcvSRCSDU) = 0 && CHK_MRBUF() <> OK => V(Nmsd_n):= 0, V(Nmend_n):= 1, V(Nms_n):= ABT_NUM SndSRCSDU:= BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_IDLE_S
14	CH_W_DATA	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) <> V(Nms_n) => SndSRCSDU:= BUILD_PDU_LAST_SMSG()	CH_W_DATA
15	CH_W_DATA	DL-ABORT-SDA.req{SAP_ID,Node_ID} => (none)	CH_ABT
16	CH_W_SYNCR	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 0 => SndSRCSDU:= BUILD_PDU_LAST_SMSG() DL-SDA.cnf{ SND_ERR }	CH_IDLE_S
17	CH_W_SYNCR	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = ABT_NUM => V(Nmsd_n):= 0, V(Nmend_n):= 1, V(Nms_n):= ABT_NUM SndSRCSDU:= BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n)) DL-SDA.cnf{ SND_ABT }	CH_IDLE_S
18	CH_W_SYNCR	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) <> ABT_NUM => V(Nmsd_n):= 0, V(Nmend_n):= 1, V(Nms_n):= 0 SndSRCSDU:= BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_W_POLL
19	CH_W_SYNCR	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1 => V(Nmsd_n):= 0, V(Nmend_n):= 1, V(Nms_n):= 0 SndSRCSDU:= BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))	CH_W_SYNCR
20	CH_W_SYNCR	DL-ABORT-SDA.req{SAP_ID,Node_ID} => (none)	CH_ABT
21	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK	CH_IDLE_S

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		&& CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 0 => SndSRCSDU:= BUILD_PDU_LAST_SMSG() DL-SDA.cnf{ SND_ERR }	
22	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && (CHK_S(RcvSRCSDU) = ABT_NUM => V(Nmsd_n):= 0, V(Nmend_n):= 1, V(Nms_n):= ABT_NUM SndSRCSDU:= BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n)) DL-SDA.cnf{ SND_ABT }	CH_IDLE_S
23	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 0 && CHK_END(RcvSRCSDU) = 1 && (CHK_S(RcvSRCSDU) <> ABT_NUM => SndSRCSDU:= BUILD_PDU_LAST_SMSG()	CH_W_POLL
24	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = V(Nms_n) && V(Nmno_n) > 1 => V(Nmsd_n):= CHK_SD(RcvSRCSDU) V(Nmend_n):= CHK_END(RcvSRCSDU) V(Nms_n):= CHK_S(RcvSRCSDU) BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n)) V(Nms_n):= V(Nms_n) + 1 V(Nmno_n):= V(Nmno_n) – 1	CH_W_POLL
25	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = V(Nms_n) && V(Nmno_n) = 1 => V(Nmsd_n):= CHK_SD(RcvSRCSDU) V(Nmend_n):= 0 V(Nms_n):= CHK_S(RcvSRCSDU) BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n)) DL-SDA.cnf{ SND_OK }	CH_IDLE_S
26	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) = V(Nms_n) && V(Nmno_n) < 1 => V(Nmsd_n):= 0, V(Nmend_n):= 1, V(Nms_n):= ABT_NUM SndSRCSDU:= BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n)) DL-SDA.cnf{ SND_ERR }	CH_ABT
27	CH_W_POLL	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts = OK && CHK_SD(RcvSRCSDU) = 1 && CHK_S(RcvSRCSDU) <> V(Nms_n) => SndSRCSDU:= BUILD_PDU_LAST_SMSG()	CH_W_POLL
28	CH_W_POLL	DL-ABORT-SDA.req{SAP_ID,Node_ID} => (none)	CH_ABT
29	CH_ABT	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) => V(Nmsd_n):= 0, V(Nmend_n):= 1, V(Nms_n):= ABT_NUM	CH_IDLE_S

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant

		SndSRCSDU:= BUILD_PDU_SMSG(V(Nmsd_n),V(Nmend_n),V(Nms_n))
		DL-ABORT-SDA.cnf
		if (V(Fmsg_sending_n) = True) then
		DL-SDA.cnf{SND_ABT}
		endif	

30	N'importe quel état	EXEC_MSPM_R(Fc1c2, Rcv_sts, RcvSRCSDU, SndSRCSDU) / Rcv_sts <> OK => (none)	Même état

6.2.3.3.4 Fonctions utilisées par la machine protocolaire de segmentation de messages

Le Tableau 58 donne la liste des fonctions utilisées par la machine de segmentation de messages.

Tableau 58 – Liste des fonctions utilisées par la machine de segmentation de messages

Nom de la fonction	Paramètre		Renvoi	Fonctionnement
	Entrée	Sortie	Valeur	
STORE_DLSDU	Ec1c2, DLS-user_message	aucun	aucun	Cette fonction stocke dans le tampon interne les données d'utilisateur de DLS du message.
GET_DLSDU	Ec1c2	aucun	DLS-user_message	Cette fonction récupère le message d'utilisateur de DLS reçu de la station distante.
BUILD_PDU_BMSG	mframe_type, Ns, Nr, flag	aucun	SRCSDU	Cette fonction construit la SRCSDU à envoyer. Les paramètres d'entrée sont les valeurs qui doivent être contenues dans le champ MCTL de la trame de message.
STORE_PDU_BMSG	Ec1c2, SRCSDU	aucun	aucun	Cette fonction stocke la SRCSDU reçue dans le tampon interne pour assembler la DLSPDU.
GET_TYP	SRCSDU	aucun	frame_type	Cette fonction extrait la valeur du champ de type de la SRCSDU spécifiée.
CHK_MC_TYP	SRCSDU	aucun	mframe_type	Cette fonction extrait le type de message du champ MCTL de la SRCSDU spécifiée.
GET_MCTL_NR	SRCSDU	aucun	Nr	Cette fonction extrait la valeur de Nr du champ MCTL de la SRCSDU spécifiée.
GET_MCTL_NS	SRCSDU	aucun	Ns	Cette fonction extrait la valeur de Ns du champ MCTL de la SRCSDU spécifiée.
GET_MCTL_F	SRCSDU	aucun	Indicateur	Cette fonction extrait la valeur de P/F du champ MCTL de la SRCSDU spécifiée.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Nom de la fonction	Paramètre		Renvoi Valeur	Fonctionnement
	Entrée	Sortie		
GET_SMSG_NUM		msg_packet_num		Ce paramètre calcule le nombre de segments de message.
CHK_SD		Résultat		Cette fonction renvoie la valeur de l'indicateur S/D provenant de la trame de message reçue.
CHK_END		Résultat		Cette fonction renvoie la valeur de l'indicateur END provenant de la trame de message reçue.
CHK_S		s_number		Cette fonction renvoie la valeur de l'indicateur S(n) provenant de la trame de message reçue.
CHK_MRBUF		Résultat		Cette fonction vérifie s'il existe de l'espace libre dans le tampon de réception.
BUILD_PDU_SMSG	SD_flag, END_flag, S_number	SRCSDU		Cette fonction construit la trame de message à envoyer à partir des données spécifiées.
BUILD_PDU_LAST_SMSG		SRCSDU		Cette fonction construit la trame de message qui est envoyée au cycle précédent.

6.2.3.4 Machine protocolaire de transmission acyclique

6.2.3.4.1 Présentation

Le présent paragraphe décrit la machine protocolaire qui est utilisée dans le mode de transmission acyclique. Lorsque la CTC entre dans l'état opérationnel, elle lance cette machine protocolaire sur demande de la DLM au lieu de lancer la machine protocolaire de transmission cyclique. Il n'existe qu'une sorte de machine protocolaire, que la station soit du type maître C1, maître C2 ou esclave.

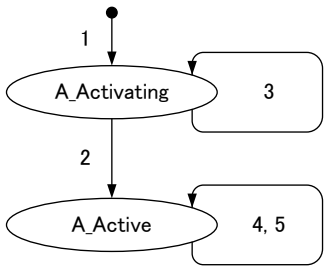


Figure 19 – Schéma d'états de la machine protocolaire de transmission acyclique

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Tableau 59 – Table d'états de la machine protocolaire de transmission acyclique

N°	État courant	Événement /condition =>action	État suivant
1	N'importe quel état	Power on or CTC_Reset.req => (none)	A_Activating
2	A_Activating	CTC_Start.req { } / ((V(Cyc_Sel) = CMode_Ayclic)) && (V(PDUType) = PDUBasic)) => CTC_Start.cnf { OK }	A_Active
3	A_Activating	CTC_Start.req / ((V(Cyc_Sel) <> CMode_Ayclic)) || (V(PDUType) <> PDUBasic)) => CTC_Start.cnf { NG }	A_Activating
4	A_Active	DL-SDN.req {SAP_ID, Node_ID, Length, DLSDU } / (V(Cyc_Sel) = CMode_Ayclic)) => SRCSDU:= BUILD_PDU_ACYC(Node_ID, Length, DLSDU) Len:= GET_LEN(SRCSDU) SRC_Send_Frame.req { Node_ID, Len, SRCSDU } DL-SDN.cnf { OK }	A_Active
5	A_Active	SRC_Recv_Frame.ind {Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) =>Node_ID:= GET_DA(RcvSRCSDU) Len:= GET_LEN(SRCSDU) DLSDU:= GET_DATA(SRCSDU) DL-SDN.ind {SAP_ID, Node_ID, Len, DLSDU }	A_Active

6.2.3.4.2 Fonctions utilisées par la machine protocolaire de transmission acyclique

Toutes les fonctions utilisées par la machine protocolaire de transmission acyclique sont résumées dans le Tableau 60.

Tableau 60 – Liste des fonctions utilisées par la machine protocolaire de transmission acyclique

Nom de la fonction	Entrée	Sortie	Fonctionnement
GET_DA	SRCSDU	Node_ID	Cette fonction extrait la valeur du champ d'adresse de destination de la SRCSDU spécifiée.
GET_LEN	SRCSDU	Length	Cette fonction extrait la valeur du champ de longueur de la SRCSDU spécifiée.
BUILD_PDU_ASYNC	Node_ID, Length, DLSDU		Cette fonction construit la SRCSDU servant à demander la SRC dans les données d'utilisateur de DLS.

6.2.4 Interface CTC-DLM

6.2.4.1 Présentation

Le présent paragraphe décrit l'interface entre la CTC et la gestion de DL.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

La CTC fournit les services suivants à la gestion de DL.

- Service de réinitialisation (CTC_Reset)
- Début de transmission (CTC_Start)
- Service d'événement d'erreur (CTC_Err_Event)

6.2.4.2 Définitions détaillées des primitives et des paramètres

Le Tableau 61 indique les primitives et les paramètres échangés entre la CTC et la DLM. Le détail de ces paramètres est indiqué dans les sections suivantes.

Tableau 61 – Primitives et paramètres échangés entre la CTC et la DLM

Primitive	Source	Paramètre	Fonction
CTC_Reset.req	DLM		Réinitialisation de la CTC
CTC_Reset.cnf	CTC	Résultat	
CTC_Start.req	DLM		Lancement de la transmission cyclique ou acyclique
CTC_Start.cnf	CTC	Résultat	
CTC_Err_Event.ind	CTC	ErrEvent_ID	Événement d'erreur

6.2.4.2.1 CTC_Reset

Aucun

6.2.4.2.2 CTC_Start

Aucun

6.2.4.2.3 CTC_Err_Event

Le Tableau 62 indique les primitives et les paramètres du service d'événement.

Tableau 62 – Primitive et paramètres d'événement d'erreur

CTC_Err_Even	Indication
Nom du paramètre	Sortie
Err_Event_ID	M

6.3 Commande d'envoi et réception

6.3.1 Généralités

La SRC comporte les fonctions suivantes.

- Envoi de trame
- Trame de réception
- Répétition ou rebouclage de la trame reçue

6.3.1.1 Envoi de trame

La SRC exécute la transmission d'une trame sur une demande provenant de la CTC et de la DLM. Lorsqu'elles demandent l'envoi d'une trame, la SRC sérialise la trame à envoyer et la remet à la couche PHY. Si deux interfaces PHY ou plus sont mises en œuvre dans la DLE, la SRC émet la trame sur chacun des ports.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

6.3.1.2 Trame de réception

Lorsque la SRC reçoit les données de la couche PHY, elle assemble la trame à partir des données reçues et informe la CTC ou la DLM de la réception.

6.3.1.3 Répétition ou rebouclage de la trame reçue

La répétition et le rebouclage sont des fonctions qui s'exécutent en s'excluant mutuellement. La répétition est la fonction active par défaut. La DLM envoie à la SRC la demande de basculement de l'une à l'autre de ces fonctions. Lorsque la DLM active la fonction de rebouclage, la fonction de répétition de la SRC se désactive.

Lorsque deux interfaces PHY ou plus sont mises en œuvre dans la DLE et que la fonction de répétition est active, la SRC répète les données reçues d'une interface PHY en direction des autres interfaces. Lorsque la fonction de rebouclage est active, la SRC renvoie les données reçues à l'interface PHY par laquelle elles sont entrées.

6.3.2 Interface SRC-CTC

6.3.2.1 Présentation

La SRC fournit les services suivants à la CTC.

- Service d'envoi de trame (SRC_Send_Frame)
- Service de réception de trame (SRC_Recv_Frame)

6.3.2.2 Définitions détaillées des primitives de service et des paramètres

Le Tableau 63 indique les primitives et les paramètres échangés entre la SRC et la CTC. Le détail de ces paramètres est indiqué dans les sections suivantes.

Tableau 63 – Primitives et paramètres pour l'interface SRC-CTC

Primitives	Source	Paramètres
SRC_Send_Frame.req	CTC	Longueur, SRCSDU
SRC_Send_Frame.cnf	SRC	Résultat
SRC_Recv_Frame.ind	SRC	Rcv_Sts, Longueur, SRCSDU

6.3.2.2.1 SRC_Send_Frame

Le Tableau 64 indique les primitives et les paramètres du service d'envoi de trame.

Tableau 64 – Primitive et paramètres d'envoi de trame

SRC_Send_Frame	Demande	Confirmation
Nom du paramètre	Entrée	Sortie
Length	M	
SRCSDU	M	
Result		M

6.3.2.2.2 SRC_Recv_Frame

Le Tableau 65 indique les primitives et les paramètres du service de réception de trame.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

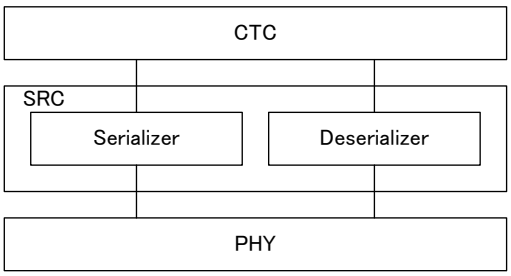
Tableau 65 – Primitive et paramètres de réception de trame

SRC_Recv_Frame	indication
Nom du paramètre	**Sortie**
Rcv_Sts	M
Length	M
SRCSDU	M

6.3.3 Spécification détaillée de la SRC

6.3.3.1 Présentation

Il existe deux sortes de SRC, selon le nombre d'interfaces PHY montées par l'entité SRC. La Figure 20 montre l'architecture interne d'une entité SRC qui n'a qu'une interface PHY, la Figure 21 montre cette architecture dans le cas où il existe deux interfaces ou plus. Si deux interfaces PHY ou plus sont montées, la fonction de répétition doit être montée dans l'entité SRC.

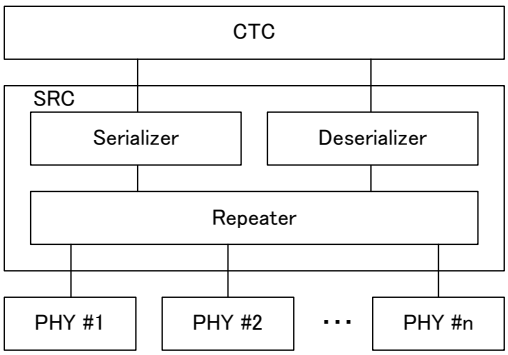


Légende

Anglais	Français
CTC	CTC
SRC	SRC
Serializer	Sérialiseur
Deserializer	Désérialiseur
PHY	PHY

Figure 20 – Architecture interne de la SRC à un port

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe



Légende

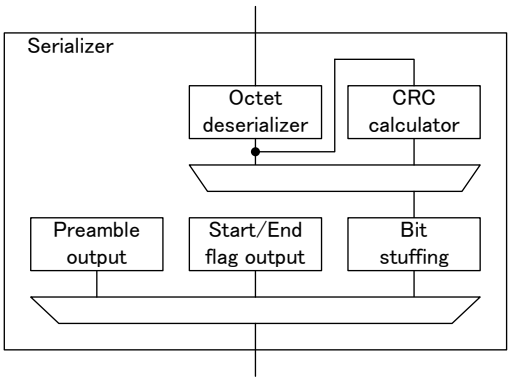
Anglais	Français
CTC	CTC
SRC	SRC
Serializer	Sérialiseur
Deserializer	Désérialiseur
Repeater	Répéteur
PHY #1	PHY n° 1
PHY #2	PHY n° 2
PHY #n	PHY n° n

Figure 21 – Architecture interne de la SRC à plusieurs ports

6.3.3.2 Sérialiseur

6.3.3.2.1 Architecture interne

La Figure 22 montre la structure interne du sérialiseur. Le sérialiseur comprend cinq blocs de fonctions.



Légende

Anglais	Français
Serializer	Sérialiseur
Octet deserializer	Désérialiseur d'octets
CRC calculator	Calculateur de CRC
Preamble output	Transmission préambule
Start/End flag output	Transmission indicateur début/fin
Bit stuffing	Remplissage de bits

Figure 22 – Architecture interne du sérialiseur

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

6.3.3.2.2 Comportement

6.3.3.2.2.1 Présentation

Lorsque la SRC a reçu la primitive de demande SRC_Send_Frame d'envoi de trame, le sérialiseur est activé et envoie la trame. Lorsque le sérialiseur a terminé l'envoi, la SRC envoie SRC_Send_Frame.cnf pour présenter le résultat au demandeur.

Le sérialiseur utilise la procédure suivante pour émettre une trame:

1) Le bloc de transmission de préambule émet la demande Ph_data
2) Le bloc de transmission d'indicateur début/fin émet la demande Ph_data de transmission de l'indicateur de début
3) Activation du bloc de remplissage de bits et du bloc de calcul de CRC
4) Le sérialiseur d'octets sérialise tous les octets de la SRCSDU.
5) Le calculateur de CRC émet la demande Ph_data
6) Désactivation du bloc de remplissage de bits et du bloc de calcul de CRC
7) Le bloc de transmission d'indicateur début/fin émet la demande Ph_data de transmission de l'indicateur de fin.

Le sérialiseur doit alors mettre la demande Ph_frame à True (vrai) lorsqu'il émet la demande Ph_data.

Le paragraphe suivant décrit les exigences de comportement de chacun des blocs de fonction dont est constitué le sérialiseur. Les exigences varient en fonction du format de DLPDU.

6.3.3.2.2.2 Sérialiseur pour trame au format de base

a) Sortie de préambule

Ce bloc de fonction émet le motif du préambule (voir 5.2.1.1).

b) Transmission de l'indicateur de début et de l'indicateur de fin

Ce bloc de fonction émet la séquence du motif binaire (voir 5.2.1.2). La fonction n'émet rien lorsque la transmission de l'indicateur de fin est demandée.

c) Sérialiseur d'octets

Ce bloc de fonction sérialise la SRCSDU du bit de poids faible au bit de poids fort d'un octet, puis émet les données traitées vers le bloc de remplissage de bits et le bloc de calcul de CRC. Lorsque la sérialisation est terminée pour la longueur spécifiée par le paramètre de la demande, ce bloc informe le calculateur de CRC que la SRCSDU est achevée. Si, durant le traitement, le sérialiseur détecte que la longueur de la SRCSDU d'entrée n'est pas égale à celle spécifiée par le paramètre de la demande, il met fin au traitement et présente l'erreur au calculateur de CRC.

d) Calculateur de CRC

Ce bloc de fonction possède deux états, activation et désactivation. Dans l'état d'activation, il exécute le calcul de CRC (voir 5.2.1.8) sur le train binaire d'entrée. Il émet le résultat du calcul sous forme de train binaire entre le bit de poids fort et le bit de poids faible lorsqu'il est informé par le sérialiseur d'octets que la SRCSDU est achevée. Dans l'état de désactivation, ce fonctionnement n'a pas lieu.

e) Remplissage de bits

Ce bloc de fonction émet le train binaire d'entrée tel quel.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

6.3.3.2.2.3 Sérialiseur pour trame au format court

a) Sortie de préambule

Ce bloc de fonction émet le motif du préambule (voir 5.3.1.1).

b) Transmission de l'indicateur de début et de l'indicateur de fin

Ce bloc de fonction émet la séquence du motif binaire (voir 5.3.1.2). La fonction émet le même motif binaire lorsque la transmission de l'indicateur de fin est demandée (voir 5.3.1.7.)

c) Sérialiseur d'octets

Voir 6.3.3.2.2.2.

d) Calculateur de CRC

Ce bloc de fonction possède deux états, activation et désactivation. Dans l'état d'activation, il exécute le calcul de CRC (voir 5.3.1.6) sur le train binaire d'entrée. Il émet le résultat du calcul sur le train binaire entre le bit de poids fort et le bit de poids faible lorsqu'il est informé de la fin de la SRCSDU par le sérialiseur d'octets. Dans l'état de désactivation, ce fonctionnement n'a pas lieu.

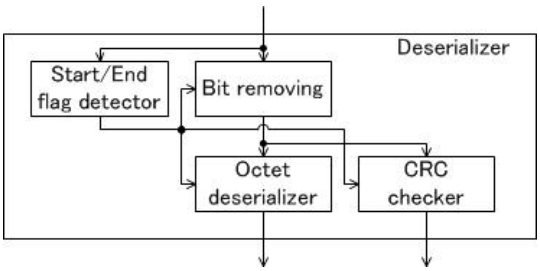
e) Remplissage de bits

Ce bloc de fonction possède deux états, activation et désactivation. Dans l'état d'activation, son comportement est le même que celui du sérialiseur HDLC. Lorsqu'il détecte cinq "1" consécutifs dans le train binaire d'entrée, il insère un "0" juste après ces "1". Sinon il émet le train binaire d'entrée tel quel. Dans l'état de désactivation, ce fonctionnement n'a pas lieu.

6.3.3.3 Désérialiseur

6.3.3.3.1 Architecture interne

La Figure 23 montre la structure interne du désérialiseur. Le désérialiseur comporte deux fonctions.



Légende

Anglais	Français
Deserializer	Désérialiseur
Octet deserializer	Désérialiseur d'octets
CRC checker	Appareil de vérification de CRC
Start/End flag detector	Détecteur d'indicateur de début/fin
Bit removing	Suppression de bits

Figure 23 – Architecture interne du désérialiseur

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

6.3.3.3.2 Comportement

6.3.3.3.2.1 Présentation

Lorsque la SRC a reçu la primitive d'indication Ph_lock de l'entité PHY, le désérialiseur est activé et reçoit le train d'entrée. Lorsque le désérialiseur a terminé la réception, la SRC émet SRC_Recv_Frame.ind pour passer la SRCPDU et le statut de réception. Le statut de réception n'est égal à OK que si la vérification de CRC est réussie et si Ph_status est Normal, sinon il est égal à NG.

Le désérialiseur utilise la procédure suivante pour recevoir des données:

1) Le détecteur d'indicateur début/fin commence à accepter l'indication Ph_data comme train d'entrée lorsqu'il reçoit l'indication Ph_lock.
2) Le détecteur d'indicateur début/fin active tous les autres blocs du désérialiseur lorsqu'il détecte l'indicateur de début dans le train d'entrée.
3) Le désérialiseur d'octets reconstitue une SRCSDU à partir du train d'entrée lorsque l'indication Ph_frame est à True (vrai).
4) Le détecteur d'indicateur début/fin informe tous les autres blocs du désérialiseur que le train est terminé lorsqu'il détecte l'indicateur de fin dans le train d'entrée ou que l'indication Ph_frame devient False (fausse).
5) L'appareil de vérification de CRC vérifie le champ de CRC et présente le résultat lorsqu'il est informé que le train est terminé.

La destination de SRC_Recv_Frame.ind est uniquement la CTC lorsque la DLM est dans un état inactif. Sinon, la destination est la DLM.

6.3.3.3.2.2 Désérialiseur pour trame au format de base

a) Détecteur d'indicateur de début et d'indicateur de fin

Ce bloc de fonction commence à accepter l'indication Ph_data comme train d'entrée lorsqu'il reçoit l'indication Ph_lock. Il active tous les autres blocs du désérialiseur lorsqu'il détecte le SFD (voir 5.2.1.2) dans le train d'entrée. Il informe tous les autres blocs du désérialiseur que le train est terminé lorsque l'indication Ph_frame devient False (fausse).

b) Suppression de bits

Ce bloc de fonction émet le train binaire d'entrée tel quel.

c) Désérialiseur d'octets

Ce bloc de fonction possède deux états, activation et désactivation. Dans l'état d'activation, il assemble la rangée d'octets à partir du train binaire d'entrée pour construire une SRCSDU. Au cours de ce processus, chaque octet est assemblé en partant du bit de poids faible et en allant jusqu'au bit de poids fort. Lorsque le détecteur d'indicateur début/fin informe le bloc que le train est terminé, le bloc passe à l'état de désactivation. Dans l'état de désactivation, ce fonctionnement n'a pas lieu.

d) Appareil de vérification de CRC

Ce bloc de fonction possède deux états, activation et désactivation. Dans l'état d'activation, il exécute le calcul de CRC (voir 5.2.1.8) sur le train binaire d'entrée. Lorsque le détecteur d'indicateur début/fin informe le bloc que le train est terminé, le bloc compare la valeur calculée aux dernières données de 32 bits du train binaire, et présente le résultat.

6.3.3.3.2.3 Désérialiseur pour trame au format court

a) Détecteur de l'indicateur de début et de l'indicateur de fin

Ce bloc de fonction commence à accepter l'indication Ph_data comme train d'entrée lorsqu'il reçoit l'indication Ph_lock. Il active tous les autres blocs du désérialiseur lorsqu'il détecte

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

l'indicateur de début (voir 5.3.1.2) dans le train d'entrée. Lorsqu'il détecte l'indicateur de fin dans le train d'entrée, il informe tous les autres blocs du désérialiseur que le train est terminé. Ou bien il informe que le traitement a été abandonné si l'indication Ph_frame est fausse avant la réception de l'indicateur de fin.

b) Suppression de bits

Ce bloc de fonction possède deux états, activation et désactivation. Dans l'état d'activation, son comportement est le même que celui du désérialiseur HDLC. Lorsque ce bloc de fonction détecte qu'un motif "0" suit cinq "1" consécutifs du train binaire d'entrée, il supprime le "0" de la sortie. Lorsque le détecteur d'indicateur début/fin informe le bloc que le train est terminé, le bloc passe à l'état de désactivation.

c) Désérialiseur d'octets

Voir 6.3.3.3.2.2 c).

d) Appareil de vérification de CRC

Ce bloc de fonction possède deux états, activation et désactivation. Dans l'état d'activation, il exécute le calcul de CRC (voir 5.3.1.6) sur le train binaire d'entrée. Lorsque le détecteur d'indicateur début/fin informe le bloc que le train est terminé, le bloc compare la valeur calculée aux dernières données de 32 bits du train binaire, et présente le résultat. S'il est informé de l'abandon du traitement par le détecteur d'indicateur début/fin, il présente l'erreur.

Ce bloc de fonction possède deux états, activation et désactivation. Dans l'état d'activation, il exécute le calcul de CRC (voir 5.2.1.8) sur le train binaire d'entrée. Il émet le résultat du calcul sous forme de train binaire entre le bit de poids fort et le bit de poids faible lorsqu'il est informé par le sérialiseur d'octets que la SRCSDU est achevée. Dans l'état de désactivation, ce fonctionnement n'a pas lieu.

Si, durant le traitement, le sérialiseur détecte que la longueur de la SRCSDU d'entrée n'est pas égale à celle spécifiée par le paramètre de la demande, il met fin au traitement et présente l'erreur au calculateur de CRC.

6.3.3.4 Répéteur

Le répéteur a deux états, l'état normal et l'état de rebouclage. L'état normal est l'état par défaut. Dans l'état normal, le répéteur émet le train entrant par une interface PHY vers toutes les autres.

Le répéteur passe à l'état rebouclage et lance le temporisateur de répétition-surveillance lorsqu'il reçoit la primitive de demande SRC_Enable_Loopback de la gestion de DL. Dans l'état de rebouclage, le répéteur renvoie le train entrant à l'interface PHY par laquelle il était entré au lieu de le transmettre aux autres interfaces PHY. Le répéteur passe à l'état normal lorsque le nombre d'exécutions spécifié par le paramètre de la primitive de demande a été atteint ou lorsque le temporisateur de répétition-surveillance est expiré. Le répéteur passe ensuite la primitive de confirmation SRC_Enable_Loopback à la gestion de DL pour signaler la réussite de la demande de service correspondante.

6.3.4 Interface SRC-DLM

6.3.4.1 Présentation

Le présent paragraphe décrit l'interface entre la CTC et la gestion de DL.

La CTC fournit les services suivants à la gestion de DL.

- Service de réinitialisation (SRC_Reset)
- Envoi de trame (SRC_Send_Frame)

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

- Réception de trame (SRC_Recv_Frame)
- Activation rebouclage (SRC_Enable_Loopback)
- Service d'événement d'erreur (SRC_Err_Event)

6.3.4.2 Définitions détaillées des primitives et des paramètres

Le Tableau 66 indique les primitives échangées entre la SRC et la DLM. Les paragraphes suivants décrivent les primitives et les paramètres.

Tableau 66 – Primitives et paramètres échangés entre la SRC et la DLM

Primitive	Source	paramètres
SRC_Reset.req	DLM	<aucun>
SRC_Reset.cnf	SRC	Résultat
SRC_Send_Frame.req	DLM	Longueur, SRCSDU
SRC_Send_Frame.cnf	SRC	Résultat
SRC_Recv_Frame.ind	SRC	Rcv_Sts, Longueur, SRCSDU
SRC_Enable_Loopback.req	DLM	LB_Cnt
SRC_Enable_Loopback.cnf	SRC	Résultat
SRC_Err_Event.ind	SRC	Err_Event_ID

6.3.4.2.1 SRC_Reset

Aucun

6.3.4.2.2 SRC_Send_Frame

Voir 6.3.2.2.1.

6.3.4.2.3 SRC_Recv_Frame

Voir 6.3.2.2.2.

6.3.4.2.4 SRC_Enable_Loopback

Le Tableau 67 indique les primitives et les paramètres du service d'événement.

Tableau 67 – Primitives et paramètres d'obtention de valeur

SRC_Enable_Loopback	Demande	Confirmation
Nom du paramètre	Entrée	Sortie
LB_Cnt	M	
Result		M

6.3.4.2.5 SRC_Err_Event

Le Tableau 68 indique les primitives et les paramètres du service d'événement.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Tableau 68 – Primitive et paramètres d'événement d'erreur

SRC_Err_Event	Indication
Nom du paramètre	**Sortie**
Err_Event_ID	M

7 Couche de gestion de DL (DLM)

7.1 Présentation

Le protocole d'interface entre la DLM et l'utilisateur de DLMS est décrit dans le présent paragraphe. Le présent paragraphe du protocole de gestion de DL donne les services de gestion de DL spécifiés à l'Article 5 de la CEI 61158-3-24 en utilisant les services disponibles pour l'utilisateur de DLMS. La DLM fournit les services permettant à l'utilisateur de DLMS d'initialiser les stations.

Dans le présent paragraphe, les aspects de la mise en œuvre complète et les aspects locaux ont été intentionnellement omis.

7.2 Définitions des primitives

7.2.1 Primitives échangées entre l'utilisateur de DLMS et la DLM

Les primitives échangées entre l'utilisateur de DLMS et la DLM sont indiquées ci-dessous. Le Tableau 69 concerne celles émises par l'utilisateur de DLMS, le Tableau 70 celles émises par la DLM.

Tableau 69 – Liste des primitives et des paramètres (source utilisateur de DLMS)

Primitive	Source	Paramètres
DLM-RESET.req	DLMS-user	
DLM-SET-VALUE.req	DLMS-user	Var_ID, Val
DLM-GET-VALUE.req	DLMS-user	Var_ID
DLM-MEAS-DELAY.req	DLMS-user	
DLM-SET-COMMOD.req	DLMS-user	
DLM-START.req	DLMS-user	
DLM-CLR-ERR.req	DLMS-user	Err_Status

Tableau 70 – Liste des primitives et des paramètres (source DLM)

Primitive	Source	Paramètres
DLM-RESET.cnf	DLM	Résultat
DLM-SET-VALUE.cnf	DLM	Résultat
DLM-GET-VALUE.cnf	DLM	Résultat, Val
DLM-MEAS-DELAY.cnf	DLM	Résultat
DLM-MEAS-DELAY.ind	DLM	Delay_time
DLM-SET-COMMOD.cnf	DLM	Résultat
DLM-START.cnf	DLM	Résultat
DLM-START.ind	DLM	Com_Mode, Cycle_time, C2_stime, Max_Delay, TM_unit
DLM-CLR-ERR.cnf	DLM	Résultat
DLM-EVENT.ind	DLM	Event_ID

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

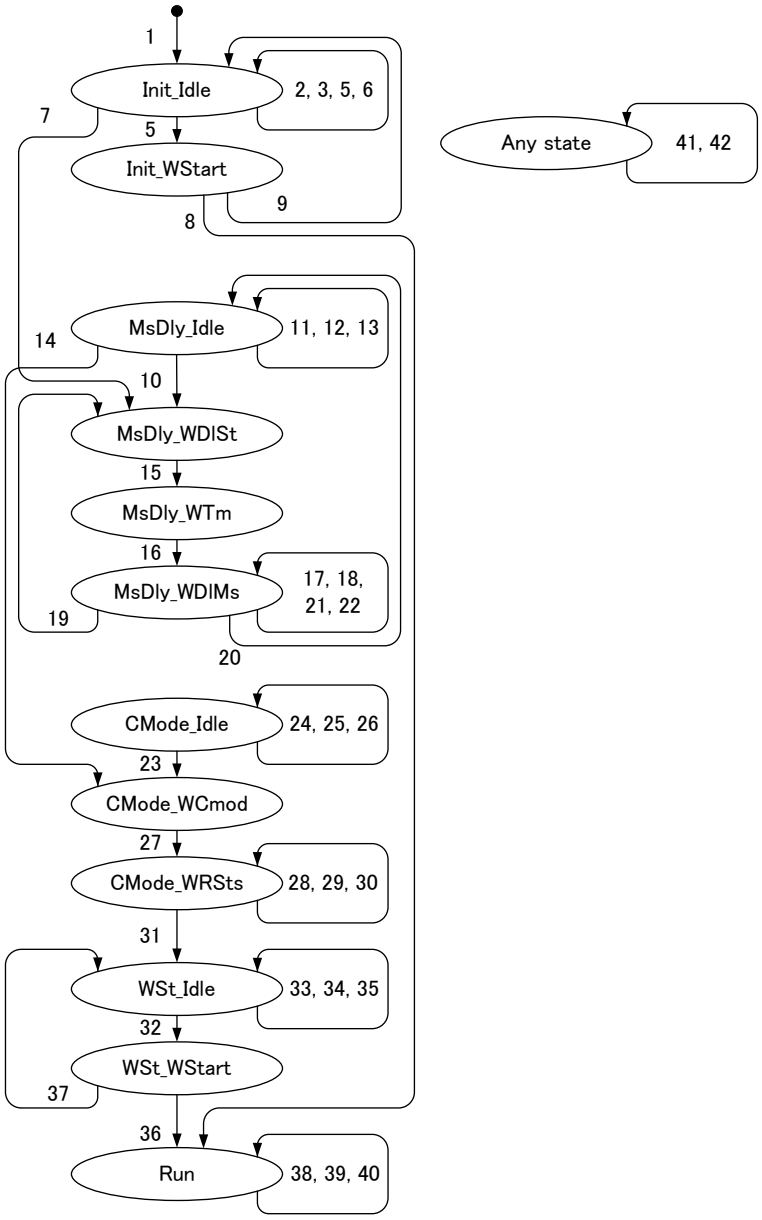
7.2.2 Paramètres utilisés avec les primitives de la DLM

Voir en 5.3 de la CEI 61158-3-24 les paramètres de la primitive de service.

7.3 Machine protocolaire DLM

7.3.1 Maître C1

Les schémas de transition d'états DLM du maître C1 sont représentés sur la Figure 24 et dans le Tableau 71.



Légende

Anglais	Français
Any state	N'importe quel état
Run	Exécuter

Figure 24 – Schéma d'états de la DLM avec maître C1

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Tableau 71 – Table d'états de la DLM avec maître C1

N°	État courant	Événement /condition => action	État suivant
1	N'importe quel état	Power on or DLM-RESET.req => DLM-RESET.cnf CTC_Reset.req { } SRC_Reset.req { }	Init_Idle
2	Init_Idle	DLM-SET-VALUE.req { Var_ID, Val } => Result:= SET_VALUE(Var_ID, Val) DLM-SET-VALUE.cnf { Result }	Init_Idle
3	Init_Idle	DLM-GET-VALUE.req { Par_ID } => Result:= GET_VALUE(Var_ID, CurVal) DLM-GET-VALUE.cnf { Result, CurVal }	Init_Idle
4	Init_Idle	DLM-CLR-ERR.req { Err_ID } => CLR_ERR_STS (Err_ID) DLM-CLR-ERR.cnf { OK }	Init_Idle
5	Init_Idle	DLM-START.req { } / (((V(Cyc_Sel) = CMode_Cyclic) && (V(SlotType) = TSFixed)) || (V(Cyc_Sel) = CMode_Ayclic)) => CTC_Start.req { }	Init_WStart
6	Init_Idle	DLM-START.req { } / (Result = OK) && (V(Cyc_Sel) = CMode_Cyclic) && (V(SlotType) = TSConfig) => DLM-START.cnf { NG }	Init_Idle
7	Init_Idle	DLM-MEAS-DELAY.req { } => V(Nslave):= 0 SRCSDU:= BUILD_PDU_DLST(V(Nslave), V(Nmax_dl_cnt)) SRC_Send_Frame.req { SRCSDU} START_TIMER_FR( )	MsDly_WDlSt
8	Init_WStart	CTC_Start.cnf { Result } / (Result = OK) => DLM-START.cnf { Result }	Run
9	Init_WStart	CTC_Start.cnf { Result } / (Result <> OK) => DLM-START.cnf { Result }	Init_Idle
10	MsDly_Idle	DLM-MEAS-DELAY.req { } => V(Nslave):= 0 SRCSDU:= BUILD_PDU_DLST(V(Nslave), V(Nmax_dl_cnt)) SRC_Send_Frame.req { SRCSDU} START_TIMER_FR( )	MsDly_WDlSt
11	MsDly_Idle	DLM-SET-VALUE.req { Var_ID, Val } => Result:= SET_VALUE(Var_ID, Val) DLM-SET-VALUE.cnf { Result }	MsDly_Idle
12	MsDly_Idle	DLM-GET-VALUE.req { Par_ID } => Result:= GET_VALUE(Var_ID, CurVal)	MsDly_Idle

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		DLM-GET-VALUE.cnf { Result, CurVal }	
13	MsDly_Idle	DLM-CLR-ERR.req { Err_ID } => CLR_ERR_STS (Err_ID) DLM-CLR-ERR.cnf { OK }	MsDly_Idle
14	MsDly_Idle	DLM-SET-COMMOD.req { } => SRCSDU:= BUILD_PDU_COMMOD(V(Cyc_Sel), V(Tcycle), V(Tc2_dly), V(Tidly), V(Tunit)) SRC_Send_Frame.req { SRCSDU}	CMode_WCmod
15	MsDly_WDlSt	SRC_Send_Frame.cnf { } => START_TIMER(T(Tdl_dlms), V(Tdl_dlms))	MsDly_WTm
16	MsDly_WTm	EXPIRED_TIMER (T(Tdl_dlms)) = True => V(Ndly_cnt) = 0 V(Ten):= LATCH_TSTAMP( ) SRCSDU:= BUILD_PDU_DLMS(V(Nslave), 0) SRC_Send_Frame.req { SRCSDU} V(Tst):= LATCH_TSTAMP( )	MsDly_WDlMs
17	MsDly_WDlMs	SRC_Send_Frame.cnf { } => (None)	MsDly_WDlMs
18	MsDly_WDlMs	SRC_Rev_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (ST_NO(V(Nslave) = RcvSRCSDU.SA) && (V(Ndly_cnt) <= V(Nmax_dly_cnt))) => V(Ten):= LATCH_TSTAMP( ) V(Tdly):= COMP_DELAY(V(Tst), V(Ten)) SRCSDU:= BUILD_PDU_MDLY(V(Nslave), V(Tdly)) SRC_Send_Frame.req { SRCSDU} V(Tst):= LATCH_TSTAMP( ) V(Ndly_cnt) = V(Ndly_cnt) + 1	MsDly_WDlMs
19	MsDly_WDlMs	SRC_Rev_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (ST_NO(V(Nslave) = RcvSRCSDU.SA) && (V(Ndly_cnt) > V(Nmax_dly_cnt)) && (V(Nslave) <= V(Nmax_slave))) => V(Nslave):= V(Nslave) + 1 SRCSDU:= BUILD_PDU_DLST(V(Nslave), V(Nmax_dl_cnt)) SRC_Send_Frame.req { SRCSDU}	MsDly_WDlSt
20	MsDly_WDlMs	SRC_Rev_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (ST_NO(V(Nslave) = RcvSRCSDU.SA) && (V(Ndly_cnt) > V(Nmax_dly_cnt)) && (V(Nslave) > V(Nmax_slave))) => STOP_TIMER_FR( ) DLM-MEAS-DELAY.cnf { }	MsDly_Idle
21	MsDly_WDlMs	SRC_Rev_Frame.ind { Rcv_sts, Length, RcvSRCSDU } /((Rcv_sts = OK) && (ST_NO(V(Nslave) <> RcvSRCSDU.SA)) => SRCSDU:= BUILD_PDU_MDLY(V(Nslave), V(Tdly))	MsDly_WDlMs

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		SRC_Send_Frame.req { SRCSDU} V(Tst):= LATCH_TSTAMP( )	
22	MsDly_WDlMs	SRC_Rev_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / (Rcv_sts <> OK) => SRCSDU:= BUILD_PDU_MDLY(V(Nslave), V(Tdly)) SRC_Send_Frame.req { SRCSDU} V(Tst):= LATCH_TSTAMP( )	MsDly_WDlMs
23	CMode_Idle	DLM-SET-COMMOD.req { } => SRCSDU:= BUILD_PDU_COMMOD(V(Cyc_Sel), V(Tcycle), V(Tc2_dly), V(Tidly), V(Tunit)) SRC_Send_Frame.req { SRCSDU}	CMode_WCmod
24	CMode_Idle	DLM-SET-VALUE.req { Var_ID, Val } => Result:= SET_VALUE(Var_ID, Val) DLM-SET-VALUE.cnf { Result }	CMode_Idle
25	CMode_Idle	DLM-GET-VALUE.req { Par_ID } => Result:= GET_VALUE(Var_ID, CurVal) DLM-GET-VALUE.cnf { Result, CurVal }	CMode_Idle
26	CMode_Idle	DLM-CLR-ERR.req { Err_ID } => CLR_ERR_STS (Err_ID) DLM-CLR-ERR.cnf { OK }	CMode_Idle
27	CMode_WCmod	SRC_Send_Frame.cnf { } => V(Nslave):= 0 SRCSDU:= BUILD_PDU_STS(V(Nslave)) SRC_Send_Frame.req { SRCSDU}	CMode_WRSts
28	CMode_WRSts	SRC_Send_Frame.cnf { } => (none)	CMode_WRSts
29	CMode_WRSts	SRC_Recv_Frame.ind { Rcv_sts, RcvSRCSDU} / (Rcv_sts = OK) && (V(Nslave) <= V(Nmax_slave)) => UPDATE_STI(V(Nslave), RcvSRCSDU.ST_NO) V(Nslave):= V(Nslave) + 1 SRCSDU:= BUILD_PDU_STS(V(Nslave)) SRC_Send_Frame.req { SRCSDU}	CMode_WRSts
30	CMode_WRSts	SRC_Recv_Frame.ind { Rcv_sts, RcvSRCSDU} / (Rcv_sts = OK) && (V(Nslave) > V(Nmax_slave)) => UPDATE_STI(V(Nslave), RcvSRCSDU.ST_NO) DLM-SET-COMMOD.ind { }	WSt_Idle
31	CMode_WRSts	SRC_Recv_Frame.ind { Rcv_sts, RcvSRCSDU} / (Rcv_sts <> OK) => SRCSDU:= BUILD_PDU_STS(V(Nslave)) SRC_Send_Frame.req { SRCSDU}	CMode_WRSts
32	WSt_Idle	DLM-START.req { } => CTC_Start.req	WSt_WStart
33	WSt_Idle	DLM-SET-VALUE.req { Var_ID, Val } => Result:= SET_VALUE(Var_ID, Val)	WSt_Idle

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		DLM-SET-VALUE.cnf { Result }	
34	WSt_Idle	DLM-GET-VALUE.req { Par_ID } => Result:= GET_VALUE(Var_ID, CurVal) DLM-GET-VALUE.cnf { Result, CurVal }	WSt_Idle
35	WSt_Idle	DLM-CLR-ERR.req { Err_ID } => CLR_ERR_STS (Err_ID) DLM-CLR-ERR.cnf { OK }	WSt_Idle
36	WSt_WStart	CTC_Start.cnf { Result } / (Result = OK) => DLM-START.cnf { Result }	Run
37	WSt_WStart	CTC_Start.cnf { Result } / (Result <> OK) => DLM-START.cnf { Result }	WSt_Idle
38	Run	DLM-SET-VALUE.req { Var_ID, Val } => Result:= SET_VALUE(Var_ID, Val) DLM-SET-VALUE.cnf { Result }	Run
39	Run	DLM-GET-VALUE.req { Par_ID } => Result:= GET_VALUE(Var_ID, CurVal) DLM-GET-VALUE.cnf { Result, CurVal }	Run
40	Run	DLM-CLR-ERR.req { Err_ID } => CLR_ERR_STS (Err_ID) DLM-CLR-ERR.cnf { OK }	Run

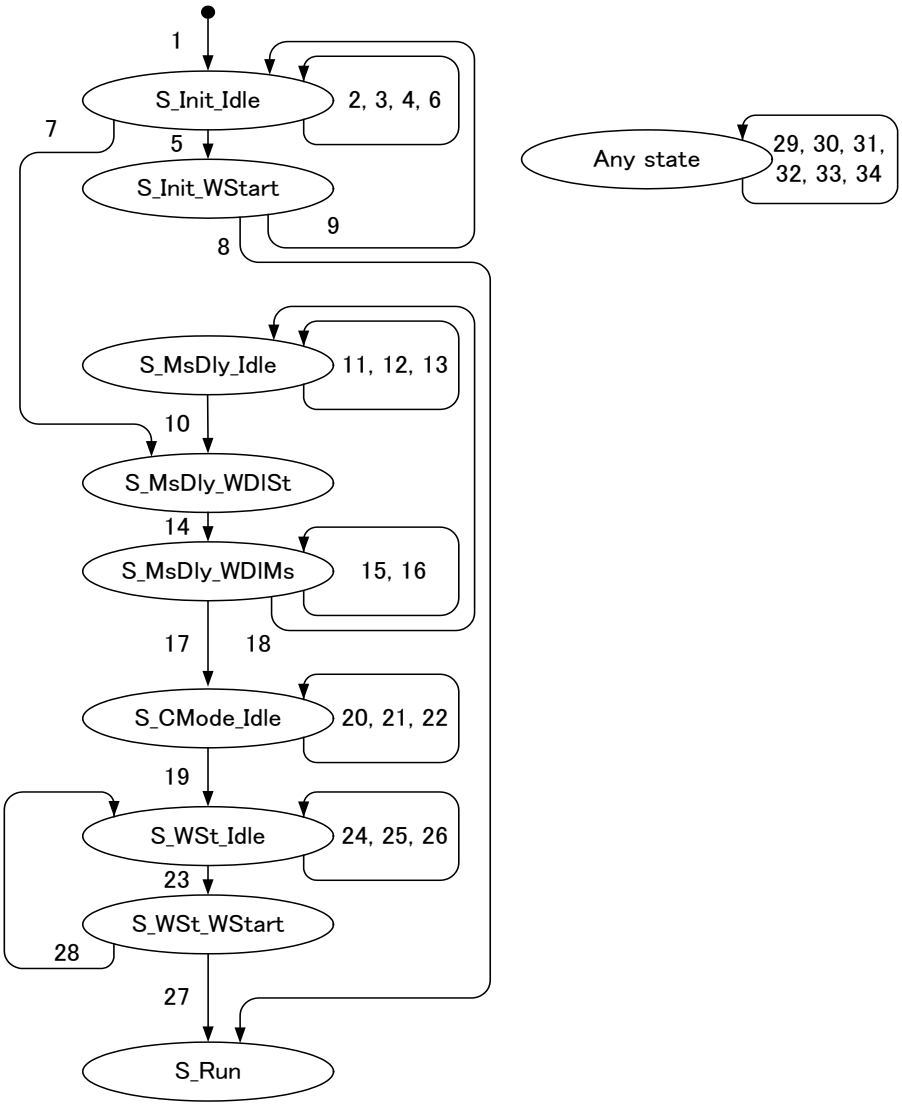
41	N'importe quel état	CTC_Err_Event.ind
		=> FsendEvent:= UPDATE_ERR_STS (Err_Event_ID)
		If (FsendEvent = True) then
		DLM-EVENT.ind { Err_Event_ID }
		Endif
			Même état
			(Pas de transition)

42	N'importe quel état	SRC_Err_Event.ind
		=> FsendEvent:= UPDATE_ERR_STS (Err_Event_ID)
		If (FsendEvent = True) then
		DLM-EVENT.ind { Err_Event_ID }
		Endif
			Même état
			(Pas de transition)

7.3.2 Esclave et maître C2

Les schémas de transition d'états DLM de l'esclave et du maître C2 sont représentés sur la Figure 25 et dans le Tableau 72.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe



Légende

Anglais	Français
Any state	N'importe quel état

Figure 25 – Schéma d'états de la DLM avec esclave et maître C2

Tableau 72 – Table d'états de la DLM avec esclave et maître C2

N°	État courant	Événement /condition => action	État suivant
1	N'importe quel état	Power on or DLM-RESET.req => DLM-RESET.cnf CTC_Reset.req { } SRC_Reset.req { }	S_Init_Idle
2	S_Init_Idle	DLM-SET-VALUE.req { Var_ID, Val } => Result:= SET_VALUE(Var_ID, Val) DLM-SET-VALUE.cnf { Result }	S_Init_Idle
3	S_Init_Idle	DLM-GET-VALUE.req { Par_ID } => Result:= GET_VALUE(Var_ID, CurVal)	S_Init_Idle

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		DLM-GET-VALUE.cnf { Result, CurVal }	
4	S_Init_Idle	DLM-CLR-ERR.req { Err_ID } => CLR_ERR_STS (Err_ID) DLM-CLR-ERR.cnf { OK }	S_Init_Idle
5	S_Init_Idle	DLM-START.req { } / (((V(Cyc_Sel) = CMode_Cyclic) && (V(SlotType) = TSFixed)) || (V(Cyc_Sel) = CMode_Ayclic)) => CTC_Start.req { }	S_Init_WStart
6	S_Init_Idle	DLM-START.req { } / ((V(Cyc_Sel) = CMode_Cyclic) && (V(SlotType) = TSConfig)) => DLM-START.cnf { NG }	S_Init_Idle
7	S_Init_Idle	DLM-MEAS-DELAY.req { } => (none)	S_MsDly_WDISt
8	S_Init_WStart	CTC_Start.cnf { Result } / (Result = OK) => DLM-START.cnf { Result }	S_Run
9	S_Init_WStart	CTC_Start.cnf { Result } / (Result <> OK) => DLM-START.cnf { Result }	S_Init_Idle
10	S_MsDly_Idle	DLM-MEAS-DELAY.req { } => (None)	S_MsDly_WDISt
11	S_MsDly_Idle	DLM-SET-VALUE.req { Var_ID, Val } => Result:= SET_VALUE(Var_ID, Val) DLM-SET-VALUE.cnf { Result }	S_MsDly_Idle
12	S_MsDly_Idle	DLM-GET-VALUE.req { Par_ID } => Result:= GET_VALUE(Var_ID, CurVal) DLM-GET-VALUE.cnf { Result, CurVal }	S_MsDly_Idle
13	S_MsDly_Idle	DLM-CLR-ERR.req { Err_ID } => CLR_ERR_STS (Err_ID) DLM-CLR-ERR.cnf { OK }	S_MsDly_Idle
14	S_MsDly_WDlSt	SRC_Recv_Frame.ind { Rcv_sts, RcvSRCSDU} / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_TYPE(RcvSRCSDU) = DLST)) => V(Ndly_cnt):= 0 V(Nmax_dly_cnt):= GET_DLY_CNT(RcvSRCSDU) CTC_Enable_LB.req { True, V(Nmax_dly_cnt) } START_TIMER(T(Twrpt), V(Twrpt))	S_MsDly_WDIMs
15	S_MsDly_WDlMs	CTC_Enable_LB.cnf { } => (none)	S_MsDly_WDIMs
16	S_MsDly_WDlMs	SRC_Recv_Frame.ind { Rcv_sts, RcvSRCSDU} / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_TYPE(RcvSRCSDU) = DLMS) && (V(Ndly_cnt) <= V(Nmax_dly_cnt)) => SAVE_DELAY(RcvSRCSDU) V(Ndly_cnt):= V(Ndly_cnt) + 1	S_MsDly_WDIMs

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
17	S_MsDly_WDlMs	SRC_Recv_Frame.ind { Rcv_sts, RcvSRCSDU} / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_TYPE(RcvSRCSDU) = DLMS) && (V(Ndly_cnt) > V(Nmax_dly_cnt)) => STOP_TIMER(T(Twrpt), V(Twrpt)) DLM-MEAS-DELAY.cnf { OK }	S_CMode_WCMod
18	S_MsDly_WDlMs	EXPIRED_TIMER(T(Twrpt)) => STOP_TIMER(T(Twrpt), V(Twrpt)) CTC_Enable_LB.req { False, 0 } DLM-MEAS-DELAY.cnf { NG }	S_MsDly_Idle
19	S_CMode_Idle	DLM-SET-COMMOD.ind { } => V(Cyc_Sel):= P(Cyc_Sel) V(Tcycle):= P(Tcycle) V(Tc2_dly):= P(Tc2_dly) V(Tidly):= P(Tc2_dly) V(Tunit):= P(Tunit)	S_WSt_Idle
20	S_CMode_Idle	DLM-SET-VALUE.req { Var_ID, Val } => Result:= SET_VALUE(Var_ID, Val) DLM-SET-VALUE.cnf { Result }	S_CMode_Idle
21	S_CMode_Idle	DLM-GET-VALUE.req { Par_ID } => Result:= GET_VALUE(Var_ID, CurVal) DLM-GET-VALUE.cnf { Result, CurVal }	S_CMode_Idle
22	S_CMode_Idle	DLM-CLR-ERR.req { Err_ID } => CLR_ERR_STS (Err_ID) DLM-CLR-ERR.cnf { OK }	S_CMode_Idle
23	S_WSt_Idle	DLM-START.req { } => CTC_Start.req	S_WSt_WCTC
24	S_WSt_Idle	DLM-SET-VALUE.req { Var_ID, Val } => Result:= SET_VALUE(Var_ID, Val) DLM-SET-VALUE.cnf { Result }	S_WSt_Idle
25	S_WSt_Idle	DLM-GET-VALUE.req { Par_ID } => Result:= GET_VALUE(Var_ID, CurVal) DLM-GET-VALUE.cnf { Result, CurVal }	S_WSt_Idle
26	S_WSt_Idle	DLM-CLR-ERR.req { Err_ID } => CLR_ERR_STS (Err_ID) DLM-CLR-ERR.cnf { OK }	S_WSt_Idle
27	S_WSt_WCTC	CTC_Start.cnf { Result } / (Result = OK) => DLM-START.cnf { Result }	S_Run
28	S_WSt_WCTC	CTC_Start.cnf { Result } / (Result = OK) => DLM-START.cnf { Result }	S_WSt_Idle
29	N'importe quel état	SRC_Err_Event.ind => FsendEvent:= UPDATE_ERR_STS (Err_Event_ID) If (FsendEvent = True) then	Même état (Pas de transition)

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

N°	État courant	Événement /condition => action	État suivant
		DLM-EVENT.ind { Err_Event_ID } Endif	
30	N'importe quel état	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (GET_TYP(RcvSRCSDU) = STS)) => SndSRCSDU:= BUILD_PDU_STS( ) SRC_Send_Frame.req { Node_ID_C1, Len_Sts, SndSRCSDU }	Même état
31	N'importe quel état (Voir NOTE)	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && (GET_DA(RcvSRCSDU) = V(MA)) && (RcvSRCSDU.TYPE <> STS)) => (none)	Même état
32	N'importe quel état	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts = OK) && ((GET_DA(RcvSRCSDU) <> V(MA)) => (none)	Même état
33	N'importe quel état	SRC_Send_Frame.cnf { } => (none)	Même état
34	N'importe quel état	SRC_Recv_Frame.ind { Rcv_sts, Length, RcvSRCSDU } / ((Rcv_sts <> OK) => (none)	Même état

7.4 Fonctions

La liste des fonctions qui sont utilisées par la machine protocolaire DLM est donnée dans le Tableau 73.

Tableau 73 – Liste des fonctions utilisées par la machine protocolaire DLM

Nom de la fonction	Entrée	Sortie	Fonctionnement
GET_VALUE	Var_ID	CurVal	Cette fonction lit la valeur courante de la variable spécifiée.
SET_VALUE	Var_ID, Val		Cette fonction stocke la valeur spécifiée dans la variable spécifiée.
BUILD_PDU_DLST	Node_ID, Nmax_dl_cnt	SRCSDU	Cette fonction construit la SRCSDU servant à demander à la SRC d'envoyer la trame de début de mesure de retard.
BUILD_PDU_DLMS	Node_ID, Tdly	SRCSDU	Cette fonction construit la SRCSDU servant à demander à la SRC d'envoyer la trame de mesure de retard.
BUILD_PDU_COMMOD	Cyc_Sel, Tcycle, Tc2_dly, Tidly, Tunit	SRCSDU	Cette fonction construit la SRCSDU servant à demander à la SRC d'envoyer la trame d'information cyclique.
BUILD_PDU_STS	Node_ID	SRCSDU	Cette fonction construit la SRCSDU servant à demander à la SRC d'envoyer la trame de statut.
START_TIMER	Tim_ID, Val	Aucun	(Voir Tableau 53)
STOP_TIMER	Tim_ID		(Voir Tableau 53)
START_TIMER_FR	Aucun	Aucun	Cette fonction lance le temporisateur à comptage libre.
STOP_TIMER_FR	Aucun	Aucun	Cette fonction arrête le temporisateur à

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Nom de la fonction	Entrée	Sortie	Fonctionnement
			comptage libre.
LATCH_TSTAMP		Timestamp	Cette fonction renvoie la valeur courante du temporisateur à comptage libre.
UPDATE_STI	Node_ID, Sts		Cette fonction met à jour le statut de connexion de la station spécifiée dans les données spécifiées par Sts.
UPDATA_ERR_STS	Err_Event_ID	FSendEvent	Cette fonction enregistre les facteurs d'erreur qui ont été signalés par la CTC et la SRC. Elle renvoie True (vrai) si l'erreur signalée est une nouvelle erreur, False (faux) si l'erreur est détectée par la valeur renvoyée.
GET_DA	SRCPDU		Cette fonction permet d'obtenir les facteurs d'erreur qui ont été signalés par la CTC et la SRC. Elle renvoie True (vrai) si l'erreur signalée est une nouvelle erreur, False (faux) si l'erreur est détectée par la valeur renvoyée.
GET_TYPE	SRCSDU		Cette fonction extrait la valeur du champ de type de la SRCSDU spécifiée.
GET_DLY_CNT	SRCSDU		Cette fonction extrait la valeur du champ de retard mesuré de la SRCSDU spécifiée.
SAVE_DLY	SRCSDU		Cette fonction met à jour V(Tdly) par la valeur du champ de retard mesuré de la SRCSDU spécifiée.
CLR_ERR_STS	Err_ID		Cette fonction efface le facteur d'erreur spécifié par Err_ID. Malgré l'exécution de cette fonction, les erreurs nécessitant une remise sous tension ou une réinitialisation ne peuvent cependant pas être effacées.

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Bibliographie

CEI 61158-1, *Réseaux de communication industriels – Spécifications des bus de terrain – Partie 1: Présentation et lignes directrices des séries CEI 61158 et CEI 61784*

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*

_____

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when print

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

Copyrighted material licensed to BR Demo by Thomson Reuters (Scientific), Inc., subscriptions.techstreet.com, downloaded on Nov-27-2014 by James Madison. No further reproduction or distribution is permitted. Uncontrolled when printe