



BSI Standards Publication

Industrial communication networks – Fieldbus specifications

Part 6-23: Application layer protocol specification –
Type 23 elements (IEC 61158-6-23:2019)

National foreword

This British Standard is the UK implementation of EN IEC 61158-6-23:2019. It is identical to IEC 61158-6-23:2019. It supersedes BS EN 61158-6-23:2014, which will be withdrawn on 25 July 2022.

The UK participation in its preparation was entrusted to Technical Committee GEL/65/3, Industrial communications: process measurement and control, including fieldbus.

A list of organizations represented on this committee can be obtained on request to its secretary.

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

© The British Standards Institution 2019
Published by BSI Standards Limited 2019

ISBN 978 0 539 05642 6

ICS 25.040.40; 35.100.70; 35.110

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 30 September 2019.

Amendments/corrigenda issued since publication

Date	Text affected
------	---------------

English Version

**Industrial communication networks - Fieldbus specifications -
Part 6-23: Application layer protocol specification - Type 23
elements
(IEC 61158-6-23:2019)**

Réseaux de communication industriels - Spécifications des
bus de terrain - Partie 6-23: Spécification du protocole de la
couche application - Eléments de type 23
(IEC 61158-6-23:2019)

Industrielle Kommunikationsnetze - Feldbusse - Teil 6-23:
Protokollspezifikation des Application Layer
(Anwendungsschicht) - Typ 23-Elemente
(IEC 61158-6-23:2019)

This European Standard was approved by CENELEC on 2019-07-25. CENELEC members are bound to comply with the CEN/CENELEC Internal Regulations which stipulate the conditions for giving this European Standard the status of a national standard without any alteration.

Up-to-date lists and bibliographical references concerning such national standards may be obtained on application to the CEN-CENELEC Management Centre or to any CENELEC member.

This European Standard exists in three official versions (English, French, German). A version in any other language made by translation under the responsibility of a CENELEC member into its own language and notified to the CEN-CENELEC Management Centre has the same status as the official versions.

CENELEC members are the national electrotechnical committees of Austria, Belgium, Bulgaria, Croatia, Cyprus, the Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, the Netherlands, Norway, Poland, Portugal, Republic of North Macedonia, Romania, Serbia, Slovakia, Slovenia, Spain, Sweden, Switzerland, Turkey and the United Kingdom.



European Committee for Electrotechnical Standardization
Comité Européen de Normalisation Electrotechnique
Europäisches Komitee für Elektrotechnische Normung

CEN-CENELEC Management Centre: Rue de la Science 23, B-1040 Brussels

European foreword

The text of document 65C/948/FDIS, future edition 2 of IEC 61158-6-23, prepared by SC 65C "Industrial networks" of IEC/TC 65 "Industrial-process measurement, control and automation" was submitted to the IEC-CENELEC parallel vote and approved by CENELEC as EN IEC 61158-6-23:2019.

The following dates are fixed:

- latest date by which the document has to be implemented at national level by publication of an identical national standard or by endorsement (dop) 2020-04-25
- latest date by which the national standards conflicting with the document have to be withdrawn (dow) 2022-07-25

This document supersedes EN 61158-6-23:2014 and all of its amendments and corrigenda (if any).

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

Endorsement notice

The text of the International Standard IEC 61158-6-23:2019 was approved by CENELEC as a European Standard without any modification.

In the official version, for Bibliography, the following notes have to be added for the standards indicated:

IEC 61784-1	NOTE	Harmonized as EN IEC 61784-1
IEC 61784-2	NOTE	Harmonized as EN IEC 61784-2

Annex ZA

(normative)

Normative references to international publications with their corresponding European publications

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

NOTE 1 Where an International Publication has been modified by common modifications, indicated by (mod), the relevant EN/HD applies.

NOTE 2 Up-to-date information on the latest versions of the European Standards listed in this annex is available here: www.cenelec.eu.

<u>Publication</u>	<u>Year</u>	<u>Title</u>	<u>EN/HD</u>	<u>Year</u>
IEC 61158-1	2019	Industrial communication networks - Fieldbus specifications - Part 1: Overview and guidance for the IEC 61158 and IEC 61784 series	EN IEC 61158-1	2019
IEC 61158-5-23	-	Industrial communication networks - Fieldbus specifications - Part 5-23: Application layer service definition - Type 23 elements	EN IEC 61158-5-23	-
IEC 61158	series	Digital data communications for measurement and control - Fieldbus for use in industrial control systems - Part 6: Application layer protocol specification		-
ISO/IEC 7498-1	-	Information technology - Open Systems Interconnection - Basic Reference Model: The Basic Model		-
ISO/IEC 8822	-	Information technology - Open Systems Interconnection - Presentation service definition		-
ISO/IEC 8824-1	-	Information technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation		-
ISO/IEC 9545	-	Information technology - Open Systems Interconnection - Application Layer structure		-
ISO/IEC 10731	-	Information technology - Open Systems Interconnection - Basic Reference Model - Conventions for the definition of OSI services		-

This page deliberately left blank

CONTENTS

FOREWORD.....	9
INTRODUCTION.....	11
1 Scope.....	13
1.1 General.....	13
1.2 Specifications	14
1.3 Conformance	14
2 Normative references	14
3 Terms, definitions, symbols, abbreviated terms and conventions	15
3.1 Referenced terms and definitions.....	15
3.1.1 ISO/IEC 7498-1 terms.....	15
3.1.2 ISO/IEC 8822 terms.....	15
3.1.3 ISO/IEC 9545 terms.....	15
3.1.4 ISO/IEC 8824-1 terms.....	15
3.1.5 IEC 61158-1 terms.....	16
3.2 Additional Type 23 terms and definitions.....	16
3.3 Symbols and abbreviated terms	18
3.4 Conventions.....	19
3.4.1 General concept	19
3.4.2 Convention for the encoding of reserved bits and octets	19
3.4.3 Conventions for abstract syntax description.....	19
3.4.4 Conventions for bit description in octets	20
3.4.5 Conventions for state machine descriptions	20
4 FAL syntax description	21
4.1 FALPDU type C abstract syntax.....	21
4.1.1 Basic abstract syntax.....	21
4.1.2 Connect-PDU	22
4.1.3 ConnectAck-PDU.....	22
4.1.4 Scan-PDU	22
4.1.5 Collect-PDU.....	23
4.1.6 Select-PDU.....	23
4.1.7 Launch-PDU	23
4.1.8 Token-PDU.....	23
4.1.9 MyStatus-PDU	24
4.1.10 Transient1-PDU	24
4.1.11 Dummy-PDU.....	25
4.1.12 Transient2-PDU	25
4.1.13 NTNTest-PDU.....	25
4.1.14 CyclicDataW-PDU.....	26
4.1.15 CyclicDataB-PDU	26
4.1.16 CyclicDataOut1-PDU	26
4.1.17 CyclicDataOut2-PDU	27
4.1.18 CyclicDataIn1-PDU	27
4.1.19 CyclicDataIn2-PDU	27
4.2 FALPDU type F abstract syntax	28
4.2.1 Basic abstract syntax.....	28
4.2.2 Persuasion-PDU	29
4.2.3 TestData-PDU	30

4.2.4	TestDataAck-PDU.....	30
4.2.5	Setup-PDU	31
4.2.6	SetupAck-PDU.....	31
4.2.7	F-Token-PDU	32
4.2.8	F-MyStatus-PDU.....	32
4.2.9	Measure-PDU	32
4.2.10	F-Offset-PDU.....	33
4.2.11	F-Update-PDU	33
4.2.12	F-CyclicData-PDU	33
4.2.13	Transient1-PDU	33
4.2.14	TransientAck-PDU	36
4.2.15	Transient2-PDU	37
4.2.16	ParamCheck-PDU	37
4.2.17	Parameter-PDU	38
4.2.18	Timer-PDU	39
4.3	Data type assignments for type C	39
4.4	Data type assignments for type F	40
5	FAL transfer syntax	41
5.1	Encoding rules	41
5.1.1	Unsigned encoding	41
5.1.2	Octet string encoding.....	41
5.1.3	SEQUENCE encoding.....	42
5.1.4	LOctetString encoding	42
5.2	FALPDU type C elements encoding	42
5.2.1	FALARHeader	42
5.2.2	Connect-PDU	44
5.2.3	ConnectAck-PDU.....	45
5.2.4	Scan-PDU	45
5.2.5	Collect-PDU.....	46
5.2.6	Select-PDU.....	48
5.2.7	Launch-PDU	49
5.2.8	Token-PDU.....	49
5.2.9	MyStatus-PDU	49
5.2.10	Transient1-PDU	51
5.2.11	Dummy-PDU.....	55
5.2.12	Transient2-PDU	56
5.2.13	NTNTest-PDU.....	67
5.2.14	CyclicDataW-PDU.....	67
5.2.15	CyclicDataB-PDU	68
5.2.16	CyclicDataOut1-PDU	69
5.2.17	CyclicDataOut2-PDU	69
5.2.18	CyclicDataIn1-PDU	70
5.2.19	CyclicDataIn2-PDU	71
5.3	FALPDU type F elements encoding.....	72
5.3.1	FALARHeader	72
5.3.2	Persuasion-PDU	76
5.3.3	TestData-PDU	77
5.3.4	TestDataAck-PDU.....	77
5.3.5	Setup-PDU	78

5.3.6	SetupAck-PDU.....	80
5.3.7	F-Token-PDU	82
5.3.8	F-Measure-PDU.....	83
5.3.9	F-Offset-PDU.....	83
5.3.10	F-Update-PDU	84
5.3.11	F-MyStatus-PDU.....	84
5.3.12	F-CyclicData-PDU	89
5.3.13	Transient1-PDU	90
5.3.14	TransientAck-PDU	94
5.3.15	Transient2-PDU	95
5.3.16	ParamCheck-PDU	98
5.3.17	Parameter-PDU	99
5.3.18	Timer-PDU	106
6	Structure of the FAL protocol state machine	106
7	FAL service protocol machine (FSPM).....	107
7.1	Overview.....	107
7.2	FSPM type C	107
7.2.1	Overview	107
7.2.2	FSPM	108
7.3	FSPM type F.....	111
7.3.1	Overview	111
7.3.2	FSPM	113
8	Application relationship protocol machine (ARPM).....	118
8.1	ARPM type C	118
8.1.1	Overview	118
8.1.2	Acyclic transmission	118
8.1.3	Cyclic transmission.....	120
8.1.4	Connection control.....	125
8.1.5	Common parameter dist.....	163
8.2	ARPM type F	168
8.2.1	Overview	168
8.2.2	Acyclic transmission	169
8.2.3	Cyclic transmission.....	171
8.2.4	Channel control	174
8.2.5	Parameter dist.....	212
8.2.6	Synchronous trigger.....	215
8.2.7	Timer.....	217
8.2.8	Measure transmission.....	218
9	DLL mapping protocol machine (DMPM).....	222
9.1	DMPM type C.....	222
9.2	DMPM type F	223
	Bibliography.....	225
	Figure 1 – Bit description in octets	20
	Figure 2 – Structure for memory access information retrieve response	59
	Figure 3 – Attribute definitions	59
	Figure 4 – Access code definitions.....	60
	Figure 5 – Structure for RUN request.....	61

Figure 6 – Structure for RUN response	61
Figure 7 – Structure for STOP request	62
Figure 8 – Structure for STOP response	62
Figure 9 – Structure for batch memory read request	62
Figure 10 – Structure for batch memory read response	63
Figure 11 – Structure for random memory read request	63
Figure 12 – Structure for random memory read response	64
Figure 13 – Structure for batch memory write request	65
Figure 14 – Structure for batch memory write response	65
Figure 15 – Structure for random memory write request	66
Figure 16 – Structure for random memory write response	66
Figure 17 – Relationships between protocol machines	107
Figure 18 – Structure of FSPM C	108
Figure 19 – Structure of FSPM F	111
Figure 20 – Structure of ARPM C	118
Figure 21 – Structure of ARPM F	168
Figure 22 – Structure of type C DMPM	222
Figure 23 – Structure of type F DMPM	224
Table 1 – State machine description elements	20
Table 2 – Description of state machine elements	20
Table 3 – Conventions used in state machines	21
Table 4 – afFTType	42
Table 5 – priority	43
Table 6 – portChoice	44
Table 7 – portCheckResult	45
Table 8 – dstPortInfo	45
Table 9 – scanState	45
Table 10 – nodeType	46
Table 11 – loopState	47
Table 12 – Cyclic status	47
Table 13 – Parameter setting mode	47
Table 14 – opState	50
Table 15 – errorState	50
Table 16 – Data type	52
Table 17 – CPW	52
Table 18 – CPWC	53
Table 19 – CPWCR	53
Table 20 – cmParam	53
Table 21 – Details of param area	54
Table 22 – Details of application parameters	54
Table 23 – Details of LB/LW CM area and LB/LW CM additional area	55
Table 24 – Details of LX/LY CM 1 area and LX/LY CM 2 area	55

Table 25 – Destination module flag	57
Table 26 – Command types	58
Table 27 – Access codes of network module memory	60
Table 28 – Access codes of controller memory	60
Table 29 – byteValidity	67
Table 30 – afFType	72
Table 31 – dataType	73
Table 32 – varField	74
Table 33 – nodeType	74
Table 34 – ProtocolVerType	75
Table 35 – Link status	78
Table 36 – Port enable/disable specification	79
Table 37 – Cyclic transmission parameter hold status	86
Table 38 – Detailed application operation status	86
Table 39 – Error detection status	86
Table 40 – Slave-specific event reception status	88
Table 41 – dataSupType of dataType (0x07)	91
Table 42 – FieldSpecificTransient opHeader	91
Table 43 – command (dataType: 0x07, dataSubType: 0x0002)	92
Table 44 – subCommand type for each command type	92
Table 45 – Structure of Deliver node information	92
Table 46 – Structure of Deliver node information – message	93
Table 47 – Structure of Get statistical information response	93
Table 48 – Structure of Acquisition of node details response	94
Table 49 – Execution module specification	96
Table 50 – Command type	97
Table 51 – Cyclic data state table	109
Table 52 – Acyclic data state table	109
Table 53 – Management state table	111
Table 54 – Cyclic data state table	114
Table 55 – Acyclic data state table	114
Table 56 – Management state table	117
Table 57 – Synchronization state table	117
Table 58 – Measurement state table	117
Table 59 – Acyclic transmission state table	119
Table 60 – Acyclic transmission functions	120
Table 61 – Cyclic transmission state table	120
Table 62 – Cyclic transmission functions	125
Table 63 – Connection control state machine – Initial	126
Table 64 – Connection control state machine – Connect	126
Table 65 – Connection control state machine – Scan	128
Table 66 – Connection control state machine – ScanWait	131
Table 67 – Connection control state machine – Collect	134

Table 68 – Connection control state machine – CollectWait	137
Table 69 – Connection control state machine – Select	140
Table 70 – Connection control state machine – TokenStartWait	143
Table 71 – Connection control state machine – LaunchWait	146
Table 72 – Connection control state machine – TokenReleaseWait	149
Table 73 – Connection control state machine – TokenReleased	152
Table 74 – Connection control state machine – TokenWait	157
Table 75 – Connection control state machine – NTNTestMaster	162
Table 76 – Connection control state machine – NTNTestSlave	162
Table 77 – Function list of connection control	163
Table 78 – Common parameter dist state table	163
Table 79 – Function list of connection control	167
Table 80 – Mapping of internal service and acyclic transmission service	168
Table 81 – Acyclic transmission states	169
Table 82 – Acyclic transmission state table	169
Table 83 – Acyclic transmission functions	171
Table 84 – Acyclic transmission variables	171
Table 85 – Cyclic transmission states	172
Table 86 – Cyclic transmission state table	172
Table 87 – Cyclic transmission functions	174
Table 88 – Cyclic transmission variables	174
Table 89 – Master station channel control states	174
Table 90 – Slave station channel control states	175
Table 91 – Master station state table – MasterDown	175
Table 92 – Master station state table – Listen	175
Table 93 – Master station state table – MasterArbitration	177
Table 94 – Master station state table – PrimaryMasterScatterTD	178
Table 95 – Master station state table – PrimaryMasterSettingUp	180
Table 96 – Master station state table – PrimaryMasterHoldToken	183
Table 97 – Master station state table – PrimaryMasterSolicitToken	186
Table 98 – Master station state table – PrimaryMasterInviting	188
Table 99 – Master station state table – MasterWaitTD	190
Table 100 – Master station state table – MasterWaitSetup	192
Table 101 – Master station state table – MasterSolicitToken (without Transmission path delay measurement)	193
Table 102 – Master station state table – MasterSolicitToken (with Transmission path delay measurement)	195
Table 103 – Master station state table – MasterHoldToken	197
Table 104 – Master station state table – MasterMeasurement (without Transmission path delay measurement function)	200
Table 105 – Master station state table – MasterMeasurement (with Transmission path delay measurement function)	200
Table 106 – Slave station state table – SlaveDown	200
Table 107 – Slave station state table – SlaveWaitTD	201

Table 108 – Slave station state table – SlaveWaitSetup.....	202
Table 109 – Slave station state table – SlaveSolicitToken (without Transmission path delay measurement)	203
Table 110 – Slave station state table – SlaveSolicitToken (with Transmission path delay measurement)	204
Table 111 – Slave station state table – SlaveHoldToken	206
Table 112 – Master station channel control functions	209
Table 113 – Slave station channel control functions	210
Table 114 – Master station channel control variables	211
Table 115 – Slave station channel control variables	211
Table 116 – Master station channel control timers	212
Table 117 – Slave station channel control timers	212
Table 118 – Master station parameter dist states	212
Table 119 – Slave station parameter dist states	212
Table 120 – Master station parameter dist state table	213
Table 121 – Slave station parameter dist state table	214
Table 122 – Master station parameter dist functions	215
Table 123 – Slave station parameter dist functions	215
Table 124 – Master station synchronous trigger states.....	216
Table 125 – Slave station synchronous trigger states.....	216
Table 126 – Master station synchronous trigger state table.....	216
Table 127 – Slave station synchronous trigger state table.....	216
Table 128 – Synchronous trigger functions	216
Table 129 – Timer states – Best effort type.....	217
Table 130 – Timer states – Fixed cycle type	217
Table 131 – Timer state table – Best effort type	217
Table 132 – Timer state table – Fixed cycle type	217
Table 133 – Timer variables.....	217
Table 134 – Fixed cycle timer	218
Table 135 – Master station measure transmission states	218
Table 136 – Slave station measure transmission states	218
Table 137 – Master station measure transmission state table	219
Table 138 – Slave station measure transmission state table	220
Table 139 – Master station measure transmission functions.....	221
Table 140 – Slave station measure transmission functions.....	221
Table 141 – Master station measure transmission variables.....	222
Table 142 – Mapping of type C DMPM service and DL service.....	223
Table 143 – Destination address for each type C PDU	223
Table 144 – Mapping of type F DMPM service and DL service	224

INTERNATIONAL ELECTROTECHNICAL COMMISSION

**INDUSTRIAL COMMUNICATION NETWORKS –
FIELDBUS SPECIFICATIONS –****Part 6-23: Application layer protocol specification –
Type 23 elements****FOREWORD**

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

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

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

International Standard IEC 61158-6-23 has been prepared by subcommittee 65C: Industrial networks, of IEC technical committee 65: Industrial-process measurement, control and automation.

This second edition cancels and replaces the first edition published in 2014. This edition constitutes a technical revision. This edition includes the following significant technical changes with respect to the previous edition:

- addition of the transmission extended mode and related attribute (Clauses 3.2.28, 4.1.9, 4.4, 5.2.9.2, and 5.3);
- update of Table 4, Table 5, Table 16 and Table 48.

The text of this International Standard is based on the following documents:

FDIS	Report on voting
65C/948/FDIS	65C/956/RVD

Full information on the voting for the approval of this International 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 the parts of the IEC 61158 series, published under the general title *Industrial communication networks – Fieldbus specifications*, can be found on the IEC website.

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

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

A bilingual version of this publication may be issued at a later date.

INTRODUCTION

This document 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 application protocol provides the application service by making use of the services available from the data-link or other immediately lower layer. The primary aim of this document is to provide a set of rules for communication expressed in terms of the procedures to be carried out by peer application entities (AEs) 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:

- as a guide for implementers and designers;
- for use in the testing and procurement of equipment;
- as part of an agreement for the admittance of systems into the open systems environment;
- as a refinement to the understanding of time-critical communications within OSI.

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

The International Electrotechnical Commission (IEC) draws attention to the fact that it is claimed that compliance with this document may involve the use of a patent concerning Type 23 elements and possibly other types given in 8.1 and 8.2 as follows:

JP 05106658 US 9350626 DE 112006003943.1 KR 10-1029201 TWI 427974	[MEC]	Communication management device, communication node, communication system, and data communication method
JP 4503678 DE 112006003895T5 KR 10-1024472 CN 102325071 TW I333356	[MEC]	Communication management device, communication device, and communication method
JP 4995294 US 8687647 DE 112006004225.4 KR 10-1024482 CN10182037 TWI 427974	[MEC]	Communication node, and token issuing method and token-ring communication method in ring communication system
JP 05127977	[MEC]	Synchronization system, time master nodes, time slave nodes and synchronization method
JP 05106658 US 9350626 DE 112008004265.9 KR 10-1277368 CN 102594592 TWI 483586B	[MEC]	Communication management device, communication node, communication system, and data communication method
JP 5220165 US 9270483 DE 112008004268 KR 10-1256767 CN 102710480 TWI 455524	[MEC]	Communication management device, communication device, and communication method

JP 05084916 US 8908566 DE112008004245 KR 10-1253931 CN 102265561 TWI 405436	[MEC]	Communication management device, communication device, and communication method
JP 5172015 US 8842521 DE 112009004913 KR 10-1307092 CN 102461085 TWI 422190	[MEC]	Communication managing apparatus, communication nodes, and data communication method
JP 5449566 US 9270554 KR 10-1479883 CN 103109491	[MEC]	Network performance estimating apparatus, network performance estimating method, network structure recognizing method, communication managing apparatus, and data communication method

IEC takes no position concerning the evidence, validity and scope of these patent rights.

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:

[MEC] Mitsubishi Electric Corporation
Corporate Licensing Division
7-3, Marunouchi 2-chome, Chiyoda-ku,
Tokyo 100-8310, Japan

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.

INDUSTRIAL COMMUNICATION NETWORKS – FIELDBUS SPECIFICATIONS –

Part 6-23: Application layer protocol specification – Type 23 elements

1 Scope

1.1 General

The Fieldbus Application Layer (FAL) provides user programs with a means to access the fieldbus communication environment. In this respect, the FAL can be viewed as a “window between corresponding application programs”.

This part of IEC 61158 provides common elements for basic time-critical and non-time-critical messaging communications between application programs in an automation environment and material specific to Type 23 fieldbus. The term “time-critical” is used to represent the presence of a time-window, within which one or more specified actions are required to be completed with some defined level of certainty. Failure to complete specified actions within the time window risks failure of the applications requesting the actions, with attendant risk to equipment, plant and possibly human life.

This International Standard defines in an abstract way the externally visible behavior provided by the different Types of the fieldbus Application Layer in terms of:

- a) the abstract syntax defining the application layer protocol data units conveyed between communicating application entities,
- b) the transfer syntax defining the application layer protocol data units conveyed between communicating application entities,
- c) the application context state machine defining the application service behavior visible between communicating application entities; and
- d) the application relationship state machines defining the communication behavior visible between communicating application entities.

The purpose of this document is to define the protocol provided to:

- a) define the wire-representation of the service primitives defined in IEC 61158-5-23, and
- b) define the externally visible behavior associated with their transfer.

This document specifies the protocol of the IEC fieldbus Application Layer, in conformance with the OSI Basic Reference Model (ISO/IEC 7498) and the OSI Application Layer Structure (ISO/IEC 9545).

FAL services and protocols are provided by FAL application-entities (AE) contained within the application processes. The FAL AE is composed of a set of object-oriented Application Service Elements (ASEs) and a Layer Management Entity (LME) that manages the AE. The ASEs provide communication services that operate on a set of related application process object (APO) classes. One of the FAL ASEs is a management ASE that provides a common set of services for the management of the instances of FAL classes.

Although these services specify, from the perspective of applications, how request and responses are issued and delivered, they do not include a specification of what the requesting and responding applications are to do with them. That is, the behavioral aspects of the applications are not specified; only a definition of what requests and responses they can

send/receive is specified. This permits greater flexibility to the FAL users in standardizing such object behavior. In addition to these services, some supporting services are also defined in this document to provide access to the FAL to control certain aspects of its operation.

1.2 Specifications

The principal objective of this document is to specify the syntax and behavior of the application layer protocol that conveys the application layer services defined in IEC 61158-5-23.

A secondary objective is to provide migration paths from previously-existing industrial communications protocols. It is this latter objective which gives rise to the diversity of protocols standardized in subparts of IEC 61158-6.

1.3 Conformance

This document does not specify individual implementations or products, nor does it constrain the implementations of application layer entities within industrial automation systems.

There is no conformance of equipment to the application layer service definition document. Instead, conformance is achieved through implementation of this application layer protocol specification.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. 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 61158-1:2019, *Industrial communication networks – Fieldbus specifications – Part 1: Overview and guidance for the IEC 61158 and IEC 61784 series*

IEC 61158-5-23, *Industrial communication networks – Fieldbus specifications – Part 5-23: Application layer service definition – Type 23 elements*

IEC 61158-6 (all parts), *Industrial communication networks – Fieldbus specifications – Part 6: Application layer protocol specification*

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

ISO/IEC 8822, *Information technology – Open Systems Interconnection – Presentation service definition*

ISO/IEC 8824-1, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation*

ISO/IEC 9545, *Information technology – Open Systems Interconnection – Application Layer structure*

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

3 Terms, definitions, symbols, abbreviated terms and conventions

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

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- IEC Electropedia: available at <http://www.electropedia.org/>
- ISO Online browsing platform: available at <http://www.iso.org/obp>

3.1 Referenced terms and definitions

3.1.1 ISO/IEC 7498-1 terms

For the purposes of this document, the following terms given in ISO/IEC 7498-1 apply:

- a) application entity
- b) application process
- c) application protocol data unit
- d) application service element
- e) application entity invocation
- f) application process invocation
- g) application transaction
- h) real open system
- i) transfer syntax

3.1.2 ISO/IEC 8822 terms

For the purposes of this document, the following terms given in ISO/IEC 8822 apply:

- a) abstract syntax
- b) presentation context

3.1.3 ISO/IEC 9545 terms

For the purposes of this document, the following terms given in ISO/IEC 9545 apply:

- a) application-association
- b) application-context
- c) application context name
- d) application-entity-invocation
- e) application-entity-type
- f) application-process-invocation
- g) application-process-type
- h) application-service-element
- i) application control service element

3.1.4 ISO/IEC 8824-1 terms

For the purposes of this document, the following terms given in ISO/IEC 8824-1 apply:

- a) object identifier
- b) type

3.1.5 IEC 61158-1 terms

For the purposes of this document, the following terms given in IEC 61158-1 apply:

- a) DLL mapping protocol machine
- b) fieldbus application layer
- c) FAL service protocol machine
- d) protocol data unit

3.2 Additional Type 23 terms and definitions

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

3.2.1

cyclic transmission

transmission that is performed periodically used for the link device update

3.2.2

intelligent device station

node capable of performing 1:n bit data and word data cyclic transmission and transient transmission with the master station, and transient transmission with slave stations, excluding remote I/O stations and having client functions and server functions during transient transmission

3.2.3

link bit

link relay bit data that are shared by all the nodes through the cyclic transmission and is used as one bit unit shared memory of the n:n type

3.2.4

link device

link bit, link word, link x and link y or RX, RY, RWr, and RWw

3.2.5

link word

link register two octet unit data that are shared by all the nodes through the cyclic transmission and is used as two octet unit shared memory of the n:n type

3.2.6

link x

link input received bit data that are transmitted from each node through the cyclic transmission and is used as an input shared memory of the 1:n type

3.2.7

link y

link output bit data that are sent to each node through the cyclic transmission and is used as an output shared memory of the 1:n type

3.2.8

local station

node capable of performing n:n bit data and word data cyclic transmission and transient transmission with the master station and other local stations, and transient transmission with slave stations, excluding remote I/O stations and having server functions and client functions during transient transmission

3.2.9

management node

node in which parameters are set

3.2.10**master station**

node that has control information (parameters) and manages cyclic transmission

3.2.11**node**

element that forms a network and performs data transmission, receiving, and transfer

3.2.12**node-to-node test**

physical layer test between two nodes

3.2.13**normal node**

node other than a management node

3.2.14**remote device station**

node capable of performing 1:n bit data and word data cyclic transmission and transient transmission with the master station, and transient transmission with slave stations, excluding remote I/O stations and having server functions during transient transmission

3.2.15**remote I/O station**

node capable of performing 1:n bit data cyclic transmission with the master station

3.2.16**reserve node**

node that is not yet connected, but counted in the total node number of the network not performing cyclic transmission, but always regarded as normal from applications

3.2.17**RX**

remote input as viewed from the master station with bit data that are periodically updated by cyclic transmission, slave to master, or in local station as viewed from the master station is RY of the local station

3.2.18**RY**

remote output as viewed from the master station with bit data that are periodically updated by cyclic transmission, master to slave, or in local station as viewed from the master station is RX of the local station

3.2.19**RWr**

remote register (input) as viewed from the master station with word data that are periodically updated by cyclic transmission, slave to master, or in local station as viewed from the master station is RWw of the local station

3.2.20**RWw**

remote register (output) as viewed from the master station with word data that are periodically updated by cyclic transmission, master to slave, or in local station as viewed from the master station is RWr of the local station

3.2.21**slave station**

node other than the master station

3.2.22

station

node of a network

3.2.23

synchronization manager

single node in a master station role per network that manages synchronization, distributing synchronization timing to other nodes

3.2.24

transient transmission

transmission that is performed upon each request

3.2.25

transient transmission client function

function that issues a transient request

3.2.26

transient transmission server function

function that receives a transient request and issues a response

3.2.27

transmission control manager

single node in a master station role per network that performs token passing management

3.2.28

transmission extended mode

operation mode to extend the number of link bit and link word.

3.2.29

word

unit representing data, 16 bits in length

3.3 Symbols and abbreviated terms

AE	Application Entity
AL	Application Layer
AP	Application Process
APDU	Application Protocol Data Unit
APO	Application Process Object
AR	Application Relationship
AREP	Application Relationship Endpoint
ASE	Application Service Element
ASN.1	Abstract Syntax Notation 1
CRC	Cyclic Redundancy Check
DLL	Data-link Layer
DMPM	DLL Mapping Protocol Machine
FAL	Fieldbus Application Layer
FSPM	FAL Service Protocol Machine
LB	Link Bit
LSB	Least Significant Bit
LW	Link Word
LX	Link X
LY	Link Y

MSB	Most Significant Bit
OSI	Open Systems Interconnection
PDU	Protocol Data Unit

3.4 Conventions

3.4.1 General concept

The FAL is defined as a set of object-oriented ASEs. Each ASE is specified in a separate subclause. Each ASE specification is composed of three parts: its class definitions, its services, and its protocol specification. The first two are contained in IEC 61158-5-23. The protocol specification for each of the ASEs is defined in this document.

The class definitions define the attributes of the classes supported by each ASE. The attributes are accessible from instances of the class using the Management ASE services specified in IEC 61158-5-23. The service specification defines the services that are provided by the ASE.

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

3.4.2 Convention for the encoding of reserved bits and octets

The term "reserved" may be used to describe bits in octets or whole octets. All bits or octets that are reserved should be set to zero at the sending side and shall not be tested at the receiving side except it is explicitly stated or if the reserved bits or octets are checked by a state machine.

The term "reserved" may also be used to indicate that certain values within the range of a parameter are reserved for future extensions. In this case the reserved values should not be used at the sending side and shall not be tested at the receiving side except it is explicitly stated or if the reserved values are checked by a state machine.

3.4.3 Conventions for abstract syntax description

This description of FAL Type 23 uses a subset of ASN.1 according to ISO/IEC 8824-1. The following structures are used.

Selective type (CHOICE) – Represents a selection from candidate types

Sequence type (SEQUENCE) – Represents a fixed-order list as in the following example:

```

DLPDU ::= SEQUENCE {
    preamble      Preamble,
    sfd            SFD,
    destaddr       DestAddr,
    srcaddr        SrcAddr,
    lt             LT,
    dlsdu          FAL-PDU,
    fcs            FCS
}

```

NOTE This example shows that the DLPDU which represents the Ethernet frame is defined as SEQUENCE. The DLPDU consists of Preamble, SFD, DestAddr, SrcAddr, LT, FAL-PDU and FCS.

3.4.4 Conventions for bit description in octets

When identifying each bit in an octet, each bit is identified by a number as shown in Figure 1 and described as Bit n.

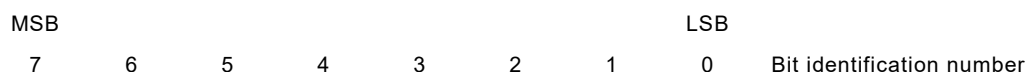


Figure 1 – Bit description in octets

When specifying multiple bits sequentially located, the range symbol (..) is used (e.g.: 7..0, specifies bits 7 through 0, inclusive).

When specifying multiple octets, the LSB of the lowest octet is considered 0, and bit identification numbers are assigned in an ascending order.

NOTE For example, when specifying 4 octets, the MSB of the highest octet is Bit 31, the MSB of the second octet is Bit 23, the MSB of the third octet is Bit 15, and the MSB of the lowest octet is Bit 7.

3.4.5 Conventions for state machine descriptions

The state machine description is defined in tabular form as shown in Table 1. The meaning of the elements is shown in Table 2. The conventions used in the state machines are shown in Table 3.

Each row of state table represents a state transition. The first column shows the state transition name or number. The second column shows the current state. The third column shows the events, conditions and actions. The fourth column shows the next state. When an event or condition is fulfilled, the action is performed and the state machine transitions to the next state.

Table 1 – State machine description elements

#	Current state	Event/condition => action	Next state
---	---------------	---------------------------	------------

Table 2 – Description of state machine elements

Heading	Description
#	state transition name or number
Current state	current state
Next state	destination state
Event	description of event
Condition	logical expression representing the condition
=> Action	action performed upon satisfaction of the event or condition

Table 3 – Conventions used in state machines

Notation	Description
=	Substitution of the right side for the left side
==	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 not 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.
&&	Logical "AND"
	Logical "OR"
!	Negation operator
+ – * /	Arithmetic operator
;	Breakpoint

4 FAL syntax description

4.1 FALPDU type C abstract syntax

4.1.1 Basic abstract syntax

The definitions of FALPDU are shown below.

```

FAL-PDU ::= CHOICE {
    connect-PDU           [0] Connect-PDU,
    connectAck-PDU        [1] ConnectAck-PDU,
    scan-PDU              [2] Scan-PDU,
    collect-PDU           [3] Collect-PDU,
    select-PDU            [4] Select-PDU,
    launch-PDU            [5] Launch-PDU,
    token-PDU             [6] Token-PDU,
    myStatus-PDU          [7] MyStatus-PDU,
    transient1-PDU        [8] Transient1-PDU,
    dummy-PDU             [9] Dummy-PDU,
    transient2-PDU        [10] Transient2-PDU,
    ntnTest-PDU           [11] NTNTest-PDU,
    cdata-PDU             CData-PDU
}

CData-PDU ::= CHOICE {
    cyclicDataW-PDU       [12] CyclicDataW-PDU,
    cyclicDataB-PDU       [13] CyclicDataB-PDU,
    cyclicDataOut1-PDU    [14] CyclicDataOut1-PDU,
    cyclicDataOut2-PDU    [15] CyclicDataOut2-PDU,
    cyclicDataIn1-PDU     [16] CyclicDataIn1-PDU,
    cyclicDataIn2-PDU     [17] CyclicDataIn2-PDU
}

```

FALARHeader to be used in each PDU are shown as follows.

```
FALARHeader ::= SEQUENCE {
    arFType          ARFType,
    priority          Priority,
    scanNumber       ScanNumber,
    reserved1        Unsigned8,
    srcNodeNumber    NodeNumber,
    reserved2        Unsigned16,
    hec              Hec
}
```

4.1.2 Connect-PDU

```
Connect-PDU ::= SEQUENCE {
    falArHeader      FALARHeader,
    portChoice       PortChoice,
    padding          OctetString SIZE(28),
    dcs              DCS
}
```

4.1.3 ConnectAck-PDU

```
ConnectAck-PDU ::= SEQUENCE {
    falArHeader      FALARHeader,
    portCheckResult  PortCheckResult,
    destPortInfo     DestPortInfo,
    padding          OctetString SIZE(20),
    dcs              DCS
}
```

4.1.4 Scan-PDU

```
Scan-PDU ::= SEQUENCE {
    falArHeader      FALARHeader,
    scanState        ScanState,
    sendTime         SendTime,
    padding          OctetString SIZE(20),
    dcs              DCS
}
```

4.1.5 Collect-PDU

```

Collect-PDU ::= SEQUENCE {
    falArHeader          FALARHeader,
    vendorCode           VendorCode,
    nodeType             NodeType,
    netNumber            NetNumber,
    sendTime             SendTime,
    loopState            LoopState,
    parmTypeCyclicStatus ParmTypeCyclicStatus,
    commonParamId        CommonParamId,
    padding              OctetString SIZE(8),
    dcs                  DCS
}

CommonParamId ::= SEQUENCE {
    date                 ParamDate,
    timeNodeId           ParamTime,
    checksum              ParamChecksum
}

```

4.1.6 Select-PDU

```

Select-PDU ::= SEQUENCE {
    falArHeader          FALARHeader,
    padding              OctetString SIZE(28),
    dcs                  DCS
}

```

4.1.7 Launch-PDU

```

Launch-PDU ::= SEQUENCE {
    falArHeader          FALARHeader,
    padding              OctetString SIZE(28),
    dcs                  DCS
}

```

4.1.8 Token-PDU

```

Token-PDU ::= SEQUENCE {
    falArHeader          FALARHeader,
    padding              OctetString SIZE(28),
    dcs                  DCS
}

```

4.1.9 MyStatus-PDU

```
MyStatus-PDU ::= SEQUENCE {
    falArHeader          FALARHeader,
    availableFuncs       AvailableFunc,
    nodeType             NodeType,
    netNumber            NetNumber,
    reserved2            Unsigned16,
    loopState            LoopState,
    parmTypeCyclicStatus ParmTypeCyclicStatus,
    commonParamId        CommonParamId,
    inFarNodeMACAddr     MACAddress,
    inFarNodeNumber      NodeNumber,
    reserved3            Unsigned8,
    outFarNodeMACAddr    MACAddress,
    outFarNodeNumber     NodeNumber,
    reserved4            Unsigned8,
    opState              OpState,
    errorState           ErrorState,
    errorCode            ErrorCode,
    vendorCode           VendorCode,
    deviceType           DeviceType,
    unitTypeName         UnitTypeName,
    unitTypeCode         UnitTypeCode,
    reserved5            Unsigned16,
    nodeInfo             NodeInfo,
    dcs                  DCS
}
```

4.1.10 Transient1-PDU

```
Transient1-PDU ::= SEQUENCE {
    falArHeader          FALARHeader,
    destinationGroup     DestinationGroup,
    seqNumber            SeqNumber,
    dataId               TraDataId,
    wholeDataSize        TraWholeDataSize,
    offsetAddr           TraOffsetAddr,
    dataSize             TraDataSize,
    dataType             TraDataType,
    data                 TraData,
    evenPadding          [0] Unsigned8 OPTIONAL,
    dcs                  DCS
}
```

4.1.11 Dummy-PDU

```

Dummy-PDU ::= SEQUENCE {
    falArHeader      FALARHeader,
    dummyData        OctetString SIZE(28..1482),
    dcs              DCS
}

```

4.1.12 Transient2-PDU

```

Transient2-PDU ::= SEQUENCE {
    falArHeader      FALARHeader,
    l               Length,
    gcnt            GateCount,
    typeSeqF        TypeSeqF,
    fno             FrameSequence,
    dt              DataFrameType,
    da              TraDstAddr,
    sa              TraSrcAddr,
    dat             TraDstAppType,
    sat             TraSrcAppType,
    dmf             TraDstModuleFlag,
    smf            TraSrcModuleFlag,
    dna             TraDstNetAddr,
    ds              TraDstStaNo,
    did            TraDstID,
    sna            TraSrcNetAddr,
    ss             TraSrcStaNo,
    sid            TraSrcID,
    l1             TraCmdLen,
    ct             TraCmdType,
    rsv            Unsigned8,
    aps            TraAppSeq,
    data           [0] TraData OPTIONAL,
    evenPadding    [1] Unsigned8 OPTIONAL,
    dcs           DCS
}

```

4.1.13 NTNTest-PDU

```

NTNTest-PDU ::= SEQUENCE {
    falArHeader      FALARHeader,
    ntnTestData      NTNTestData,
    dcs              DCS
}

```

4.1.14 CyclicDataW-PDU

```

CyclicDataW-PDU ::= SEQUENCE {
    falArHeader          FALARHeader,
    seqNumber            SeqNumber,
    byteValidity         ByteValidity,
    dataSize             CycDataSize,
    offsetAddr           CycOffsetAddr,
    exSeqNumber          CycExSeqNumber,
    reserved             Unsigned16,
    wData               CycWData,
    evenPadding          [0] Unsigned8 OPTIONAL,
    dcs                 DCS
}

```

4.1.15 CyclicDataB-PDU

```

CyclicDataB-PDU ::= SEQUENCE {
    falArHeader          FALARHeader,
    seqNumber            SeqNumber,
    byteValidity         ByteValidity,
    dataSize             CycDataSize,
    offsetAddr           CycOffsetAddr,
    reserved1            Unsigned16,
    reserved2            Unsigned16,
    bData               CycBData,
    evenPadding          [0] Unsigned8 OPTIONAL,
    dcs                 DCS
}

```

4.1.16 CyclicDataOut1-PDU

```

CyclicDataOut1-PDU ::= SEQUENCE {
    falArHeader          FALARHeader,
    seqNumber            SeqNumber,
    byteValidity         ByteValidity,
    dataSize             CycDataSize,
    offsetAddr           CycOffsetAddr,
    reserved1            Unsigned16,
    reserved2            Unsigned16,
    out1Data            CycOut1Data,
    evenPadding          [0] Unsigned8 OPTIONAL,
    dcs                 DCS
}

```

4.1.17 CyclicDataOut2-PDU

```
CyclicDataOut2-PDU ::= SEQUENCE {  
    falArHeader          FALARHeader,  
    seqNumber            SeqNumber,  
    byteValidity         ByteValidity,  
    dataSize             CycDataSize,  
    offsetAddr           CycOffsetAddr,  
    reserved1            Unsigned16,  
    reserved2            Unsigned16,  
    out2Data             CycOut2Data,  
    evenPadding          [0] Unsigned8 OPTIONAL,  
    dcs                  DCS  
}
```

4.1.18 CyclicDataIn1-PDU

```
CyclicDataIn1-PDU ::= SEQUENCE {  
    falArHeader          FALARHeader,  
    seqNumber            SeqNumber,  
    byteValidity         ByteValidity,  
    dataSize             CycDataSize,  
    offsetAddr           CycOffsetAddr,  
    reserved1            Unsigned16,  
    reserved2            Unsigned16,  
    in1Data              CycIn1Data,  
    evenPadding          [0] Unsigned8 OPTIONAL,  
    dcs                  DCS  
}
```

4.1.19 CyclicDataIn2-PDU

```
CyclicDataIn2-PDU ::= SEQUENCE {  
    falArHeader          FALARHeader,  
    seqNumber            SeqNumber,  
    byteValidity         ByteValidity,  
    dataSize             CycDataSize,  
    offsetAddr           CycOffsetAddr,  
    reserved1            Unsigned16,  
    reserved2            Unsigned16,  
    in2Data              CycIn2Data,  
    evenPadding          [0] Unsigned8 OPTIONAL,  
    dcs                  DCS  
}
```

4.2 FALPDU type F abstract syntax

4.2.1 Basic abstract syntax

The definitions of FALPDU are shown below.

```
FAL-PDU ::= CHOICE {
    f-channelControl-PDU      F-ChannelControl-PDU,
    f-sync-PDU                F-Sync-PDU,
    f-cyclicData-PDU          F-CData-PDU,
    f-transientData-PDU       F-TraData-PDU
}
```

```
F-ChannelControl-PDU ::= CHOICE {
    persuasion-PDU            [16] Persuasion-PDU,
    testData-PDU              [17] TestData-PDU,
    testDataAck-PDU           [18] TestDataAck-PDU,
    setup-PDU                 [19] Setup-PDU,
    setupAck-PDU              [20] SetupAck-PDU,
    token-PDU                 [21] F-Token-PDU,
    myStatus-PDU              [32] F-MyStatus-PDU
}
```

```
F-Sync-PDU ::= CHOICE {
    measure-PDU               [50] F-Measure-PDU,
    measureAck-PDU            [51] F-Measure-PDU,
    offset-PDU                [52] F-Offset-PDU,
    update-PDU                [53] F-Update-PDU
}
```

```
F-CData-PDU ::= CHOICE {
    cyclicDataRWw-PDU         [130] F-CyclicData-PDU,
    cyclicDataRY-PDU          [131] F-CyclicData-PDU,
    cyclicDataRWr-PDU         [132] F-CyclicData-PDU,
    cyclicDataRX-PDU          [133] F-CyclicData-PDU
}
```

```
F-TraData-PDU ::= CHOICE {
    transient1-PDU            [34] Transient1-PDU,
    transientAck-PDU          [35] TransientAck-PDU,
    transient2-PDU            [37] Transient2-PDU,
    paramCheck-PDU            [40] ParamCheck-PDU,
    parameter-PDU             [41] Parameter-PDU,
    timer-PDU                 [44] Timer-PDU
}
```

The following shows the FALARHeader used in each PDU.

```

FALAR-FHeader ::= SEQUENCE {
    arFType                ARFType,
    dataType                DataType,
    varField                CHOICE {
        vField0 [0] SEQUENCE {
            persPriority     PersPriority,
            nodeType        NodeType
        },
        vField1 [1] SEQUENCE {
            reserved1        OCTET STRING (SIZE (4))
        },
        vField2 [2] SEQUENCE {
            nodeId           NodeId,
            reserved2        OCTET STRING (SIZE (2))
        },
        vField3 [3] SEQUENCE {
            nodeId           NodeId,
            syncFlag         SyncFlag,
            nodeType         NodeType
        },
        vField4 [4] SEQUENCE {
            nodeId           NodeId,
            connectionInfo   ConnectionInfo,
            reserved4        OCTET STRING (SIZE (1))
        }
    },
    srcNodeNumber           NodeNumber,
    protocolVerType         ProtocolVerType,
    reserved                OCTET STRING (SIZE (1)),
    hec                    Hec
}

```

4.2.2 Persuasion-PDU

```

Persuasion-PDU ::= SEQUENCE {
    falArHeader             FALAR-FHeader,
    reserved1               OCTET STRING (SIZE (1)),
    myPorts                 Unsigned8,
    vendorCode              VendorCode,
    modelCode               ModelCode,
    reserved2               OCTET STRING (SIZE (20)),
    dcs                    DCS
}

```

4.2.3 TestData-PDU

```

TestData-PDU ::= SEQUENCE {
    falArHeader      FALAR-FHeader,
    tmMacAddr        MACAddress,
    srcPort          PortNumber,
    reserved         OCTET STRING (SIZE (21)),
    dcs              DCS
}

```

4.2.4 TestDataAck-PDU

```

TestDataAck-PDU ::= SEQUENCE {
    falArHeader      FALAR-FHeader,
    tdSrcMacAddr     MACAddress,
    tdSrcPort        PortNumber,
    tdRcvPort        PortNumber,
    reserved1        OCTET STRING (SIZE (1)),
    myPorts          Unsigned8,
    tokenKeepTime    Unsigned16,
    reserved2        OCTET STRING (SIZE (4)),
    myConnectStatus  SEQUENCE {
        port2port1   PortStatus,
        port4port3   PortStatus,
        port6port5   PortStatus,
        port8port7   PortStatus,
        port10port9  PortStatus,
        port12port11 PortStatus,
        port14port13 PortStatus,
        port16port15 PortStatus,
        port18port17 PortStatus,
        port20port19 PortStatus,
        port22port21 PortStatus,
        port24port23 PortStatus
    },
    dcs              DCS
}

```

4.2.5 Setup-PDU

```

Setup-PDU ::= SEQUENCE {
    falArHeader          FALAR-FHeader,
    tokenDstMacAddr      MACAddress,
    reserved1            OCTET STRING (SIZE (2)),
    leaveTimerValue      LeaveTimer,
    portUsage            PortUsage,
    reserved2            OCTET STRING (SIZE (1)),
    netBehaviour         NetworkBehaviour,
    reserved3            OCTET STRING (SIZE (12)),
    dcs                 DCS
}

NetworkBehaviour ::= SEQUENCE {
    multipleTranmit      Unsigned8,
    frameInterval        Unsigned8,
    reserved             OCTET STRING (SIZE (1)),
    multipleTokens       Unsigned8
}

```

4.2.6 SetupAck-PDU

```

SetupAck-PDU ::= SEQUENCE {
    falArHeader          FALAR-FHeader,
    slaveNodeInfo        SlaveNodeInfo,
    fwVersion            Version,
    deviceType           DeviceType,
    reserved1            OCTET STRING (SIZE (2)),
    vendorCode           VendorCode,
    modelCode            ModelCode,
    rySize               Unsigned16,
    rwwSize              Unsigned16,
    rxSize               Unsigned16,
    rwrSize              Unsigned16,
    reserved2            OCTET STRING (SIZE (2)),
    availableFuncs       AvailableFuncs,
    reserved3            OCTET STRING (SIZE (5)),
    dcs                 DCS
}

```

4.2.7 F-Token-PDU

```

F-Token-PDU ::= SEQUENCE {
    falArHeader          FALAR-FHeader,
    tokenDstMacAddr      MACAddress,
    tokenSeqNumber       Unsigned8,
    reserved1            OCTET STRING (SIZE (1)),
    tokenHopCounter      Unsigned16,
    traAvailHopCounter   Unsigned16,
    traLastHopCounter    Unsigned16,
    traAllows            Unsigned8,
    reserved2            OCTET STRING (SIZE (13)),
    dcs                  DCS
}

```

4.2.8 F-MyStatus-PDU

```

F-MyStatus-PDU ::= SEQUENCE {
    falArHeader          FALAR-FHeader,
    seqNumber            SeqNumber,
    netNumber            NetNumber,
    masterCmd            Unsigned16,
    cyclicStatus         Unsigned16,
    nodeStatus           Unsigned16,
    errorCode            ErrorCode,
    portStatus           SEQUENCE {lower PortStatus,
                                upper PortStatus },
    portStatistics       SEQUENCE {lower PortStatistics,
                                upper PortStatistics },
    portIndex            Unsigned8,
    reserved             OCTET STRING (SIZE (3)),
    cyclicSeqNumber      Unsigned8,
    addrTableDistResult  Unsigned8,
    slaveSpfEventInfo1   Unsigned8,
    slaveSpfEventInfo2   Unsigned16,
    vendorSpfNodeInfo    OCTET STRING (SIZE (4)),
    dcs                  DCS
}

```

4.2.9 Measure-PDU

```

F-Measure-PDU ::= SEQUENCE {
    falArHeader          FALAR-FHeader,
    reserved            OCTET STRING (SIZE (28)),
    dcs                  DCS
}

```

4.2.10 F-Offset-PDU

F-Offset-PDU::= SEQUENCE {	
falArHeader	FALAR-FHeader,
reserved	OCTET STRING (SIZE (8)),
syncOffset	SyncOffset,
reserved2	OCTET STRING (SIZE (16)),
dcs	DCS
}	

4.2.11 F-Update-PDU

F-Update-PDU::= SEQUENCE {	
falArHeader	FALAR-FHeader,
reserved	OCTET STRING (SIZE (8)),
syncOffset	SyncOffset,
reserved2	OCTET STRING (SIZE (16)),
dcs	DCS
}	

4.2.12 F-CyclicData-PDU

F-CyclicData-PDU::= SEQUENCE {	
falArHeader	FALAR-FHeader,
seqNumber	SeqNumber,
bothEndsValidity	BothEndsValidity,
cycDataSize	Unsigned16,
offsetAddr	CycOffsetAddr,
reserved	OCTET STRING (SIZE (4)),
cycData	CycData,
dcs	DCS
}	

4.2.13 Transient1-PDU

Transient1-PDU::= SEQUENCE {	
falArHeader	FALAR-FHeader,
traMsgHeader	TraMsgHeader,
data	OCTET STRING (SIZE (12..1466)),
dcs	DCS
}	

```

TraMsgHeader ::= SEQUENCE {
    reserved                OCTET STRING (SIZE (4)),
    seqNumber               SeqNumber,
    dataId                  TraDataId,
    wholeDataSize           TraWholeDataSize,
    offsetAddr              TraOffsetAddr,
    dataSize                TraDataSize,
    dataSubType             TraDataSubType
}

FieldSpecificTransient ::= SEQUENCE {
    opHeader                TraMsgCmdExHeader,
    fSTraData               CHOICE {
        nodeInfoDist        [721] TraSysNodeInfoDist,
        statisticsGet        [723] TraSysStatisticsGet,
        nodeInfoDetailGet    [724] TraSysNodeInfoDetailGet,
        ...
    }
}

TraMsgCmdExHeader ::= SEQUENCE {
    command                 TraCommand,
    subCommand              TraSubCommand,
    rtn                     CHOICE {
        reserved            [0] OCTET STRING (SIZE (2)),
        value                [1] Unsigned16
    },
    reserved1               OCTET STRING (SIZE (1)),
    destNetNumber           NetNumber,
    destNodeNumber          NodeNumber,
    reserved2               OCTET STRING (SIZE (5)),
    srcNetNumber            NetNumber,
    srcNodeNumber           NodeNumber,
    reserved3               OCTET STRING (SIZE (4))
}

TraSysNodeInfoDist ::= SEQUENCE {
    seqNumber               SeqNumber,
    masterNetNumber         NetNumber,
    masterDeviceType        DeviceType,
    masterModelCode         ModelCode,
    masterVendorCode        VendorCode,
    masterNodeType          NodeType,
    Reserved1               OCTET STRING (SIZE (1)),
    masterMacAddress        MACAddress
    Reserved2               OCTET STRING (SIZE (2)),

```

dataNum	Unsigned32,
messages	SEQUENCE OF NodeInfoMessage
}	
NodeInfoMessage ::= SEQUENCE {	
nodeNumber	NodeNumber,
reserved1	OCTET STRING (SIZE (1)),
availableFuncs	AvailableFuncs,
reserved2	OCTET STRING (SIZE (1)),
netNumber	NetNumber,
deviceType	DeviceType,
modelCode	ModelCode,
vendorCode	VendorCode,
nodeType	NodeType,
reserved3	OCTET STRING (SIZE (1)),
macAddress	MACAddress,
reserved4	OCTET STRING (SIZE (2))
}	
TraSysStatisticsGet ::= CHOICE {	
statGetRequest	[0] SEQUENCE {
},	
statGetResponse	[1] SEQUENCE {
port1Mib1	Unsigned32,
port1Mib2	Unsigned32,
port1Mib3	Unsigned32,
port1Mib4	Unsigned32,
port1Mib5	Unsigned32,
port1Mib6	Unsigned32,
port1Mib7	Unsigned32,
reserved	OCTET STRING (SIZE (4)),
port2Mib1	Unsigned32,
port2Mib2	Unsigned32,
port2Mib3	Unsigned32,
port2Mib4	Unsigned32,
port2Mib5	Unsigned32,
port2Mib6	Unsigned32,
port2Mib7	Unsigned32,
healthStatusNum	Unsigned32,
healthStatus	SEQUENCE SIZE (0..128) OF Unsigned32
}	
}	

```

TraSysNodeInfoDetailGet ::= CHOICE {
    nodeInfoDetailGetRequest      [0] SEQUENCE {
    },
    nodeInfoDetailGetResponse     [1] SEQUENCE {
        rySize                    Unsigned16,
        rwwSize                   Unsigned16,
        rxSize                    Unsigned16,
        rwrSize                   Unsigned16,
        reserved1                 OCTET STRING (SIZE (1)),
        ports                    Unsigned8,
        tokenKeepTime             Unsigned16,
        netBehaviour              NetworkBehaviour,
        nodeInfo                  SlaveNodeInfo,
        fwVersion                 Version,
        deviceType                DeviceType,
        modelCode                 ModelCode,
        vendorCode                VendorCode,
        reserved2                 OCTET STRING (SIZE (2)),
        modelName                 OCTET STRING (SIZE (20)),
        vendorName                OCTET STRING (SIZE (32)),
        contInfo                  Unsigned8,
        contFwVersion             Version,
        contDeviceType            DeviceType,
        contModelCode             ModelCode,
        contVendorCode            VendorCode,
        reserved3                 OCTET STRING (SIZE (2)),
        contModelName             OCTET STRING (SIZE (20)),
        contVendorName            OCTET STRING (SIZE (32)),
        contVendorSpecificInfo    OCTET STRING (SIZE (4))
    }
}

```

4.2.14 TransientAck-PDU

```

TransientAck-PDU ::= SEQUENCE {
    falArHeader                  FALAR-FHeader,
    acks                        Unsigned32,
    ackData                     SEQUENCE OF TraAckData,
    dcs                         DCS
}

```

4.2.15 Transient2-PDU

```

Transient2-PDU ::= SEQUENCE {
    falArHeader          FALAR-FHeader,
    l                    TraLength,
    reserved              OCTET STRING (SIZE (1)),
    tp                   TraType,
    fno                  TraFrameSequence,
    dt                   TraDataFrameType,
    da                   TraDstAddr,
    sa                   TraSrcAddr,
    dat                  TraDstAppType,
    sat                  TraSrcAppType,
    dmf                  TraDstModuleFlag,
    smf                  TraSrcModuleFlag,
    dna                  TraDstNetAddr,
    ds                   TraDstStaNo,
    did                  TraDstID,
    sna                  TraSrcNetAddr,
    ss                   TraSrcStaNo,
    sid                  TraSrcID,
    l1                   TraCmdLen,
    ct                   TraCmdType,
    dno                  TraDataNo,
    aps                  TraAppSeq,
    rst                  TraReturnStatus,
    data                 Tra2Data,
    dcs                  DCS
}

```

4.2.16 ParamCheck-PDU

```

ParamCheck-PDU ::= SEQUENCE {
    falArHeader          FALAR-FHeader,
    reserved1             OCTET STRING (SIZE (4)),
    paramId              CommonParamId,
    reserved2             OCTET STRING (SIZE (12)),
    dcs                  DCS
}

CommonParamId ::= SEQUENCE {
    date                 ParamDate,
    timeNodeId           ParamTime,
    checksum              ParamChecksum
}

```

4.2.17 Parameter-PDU

```

Parameter-PDU ::= SEQUENCE {
    falArHeader          FALAR-FHeader,
    paramSetFlag         ParamFlag,
    addressOrder         AddressOrder,
    cmdOrder             CmdOrder,
    cyclicParameter      CyclicParameter,
    dcs                  DCS
}

AddressOrder ::= SEQUENCE {
    assignedNetNumber    NetNumber,
    assignedNodeNumber   NodeNumber
}

CmdOrder ::= SEQUENCE {
    cmd                  Unsigned24,
    nodeType             NodeType
}

CyclicParameter ::= SEQUENCE {
    paramId              CommonParamId,
    reserved1            OCTET STRING (SIZE (2)),
    masterStatus         Unsigned16,
    rySeqNumber          SeqNumber,
    ryBothEndsValidity   BothEndsValidity,
    ryDataSize           Unsigned16,
    ryOffset             Unsigned16,
    reserved2            OCTET STRING (SIZE (2)),
    rwwSeqNumber         SeqNumber,
    reserved3            OCTET STRING (SIZE (1)),
    rwwDataSize          Unsigned16,
    rwwOffset            Unsigned16,
    reserved4            OCTET STRING (SIZE (3)),
    rxBothEndsValidity   BothEndsValidity,
    rxDataSize           Unsigned16,
    rxOffset             Unsigned32,
    reserved5            OCTET STRING (SIZE (2)),
    rwrDataSize          Unsigned16,
    rwrOffset            Unsigned32,
    reserved6            OCTET STRING (SIZE (4)),
    masterWatchTimer     Unsigned16,
    reserved7            OCTET STRING (SIZE (3)),
    cmRyBothEndsValidity BothEndsValidity,
    cmRyDataSize         Unsigned16,
    cmRyOffset           Unsigned32,

```

reserved8	OCTET STRING (SIZE (2)),
cmRwwDataSize	Unsigned16,
cmRwwOffset	Unsigned32,
reserved9	OCTET STRING (SIZE (1)),
cmRxBothEndsValidity	BothEndsValidity,
cmRxDataSize	Unsigned16,
cmRxOffset	Unsigned32,
reserved10	OCTET STRING (SIZE (2)),
cmRwrDataSize	Unsigned16,
cmRwrOffset	Unsigned32
}	

4.2.18 Timer-PDU

Timer-PDU::= SEQUENCE {	
falArHeader	FALAR-FHeader,
time	Timer,
reserved	OCTET STRING (SIZE (22)),
dcs	DCS
}	

4.3 Data type assignments for type C

Data types used in FALPDU type C abstract syntax are shown as follows.

```
ARFType ::= Unsigned8
DCS ::= Unsigned8
Priority ::= Unsigned8
ScanNumber ::= Unsigned24
NodeNumber ::= Unsigned16
Hec ::= Unsigned32
PortChoice ::= Unsigned32
PortCheckResult ::= Unsigned32
DestPortInfo ::= Unsigned32
ScanState ::= Unsigned32
SendTime ::= Unsigned16
VendorCode ::= Unsigned16
NodeType ::= Unsigned8
NetNumber ::= Unsigned8
LoopState ::= Unsigned8
ParmTypeCyclicStatus ::= Unsigned8
ParamDate ::= Unsigned32
ParamTime ::= Unsigned32
ParamChecksum ::= Unsigned32
OpState ::= Unsigned16
ErrorState ::= Unsigned16
ErrorCode ::= Unsigned16
DeviceType ::= Unsigned16
UnitTypeName ::= VisibleString SIZE(20)
UnitTypeCode ::= Unsigned16
NodeInfo ::= OctetString SIZE(96)
DestinationGroup ::= Unsigned32
SeqNumber ::= Unsigned8
TraDataId ::= Unsigned8
TraWholeDataSize ::= Unsigned16
TraOffsetAddr ::= Unsigned32
TraDataSize ::= Unsigned16
TraDataType ::= Unsigned16
```

```

TraData ::= LOctetString SIZE(12..1466)
Length ::= Unsigned16
GateCount ::= Unsigned8
TypeSeqF ::= Unsigned8
FrameSequence ::= Unsigned8
DataFrameType ::= Unsigned8
TraDstAddr ::= Unsigned8
TraSrcAddr ::= Unsigned8
TraDstAppType ::= Unsigned8
TraSrcAppType ::= Unsigned8
TraDstModuleFlag ::= Unsigned8
TraSrcModuleFlag ::= Unsigned8
TraDstNetAddr ::= Unsigned8
TraDstStaNo ::= Unsigned8
TraDstID ::= Unsigned16
TraSrcNetAddr ::= Unsigned8
TraSrcStaNo ::= Unsigned8
TraSrcID ::= Unsigned16
TraCmdLen ::= Unsigned16
TraCmdType ::= Unsigned8
TraAppSeq ::= Unsigned16
Tra2Data ::= LOctetString SIZE(12..1466)
NTNTestData ::= OctetString SIZE(28..1480)
ByteValidity ::= Unsigned8
CycDataSize ::= Unsigned16
CycOffsetAddr ::= Unsigned32
CycExSeqNumber ::= Unsigned16
CycWData ::= LOctetString SIZE(16..1468)
CycBData ::= LOctetString SIZE(16..1468)
CycOut1Data ::= LOctetString SIZE(16..1468)
CycOut2Data ::= LOctetString SIZE(16..1468)
CycIn1Data ::= LOctetString SIZE(16..1024)
CycIn2Data ::= LOctetString SIZE(16..1024)

```

4.4 Data type assignments for type F

Data types used in FALPDU type F abstract syntax are shown as follows.

```

DCS ::= Unsigned32
ARFType ::= Unsigned8
AvailableFuncs ::= Unsigned16
DataType ::= Unsigned8
NodeNumber ::= Unsigned16
ProtocolVerType ::= Unsigned8
Hec ::= Unsigned32
PersPriority ::= Unsigned24
NodeType ::= Unsigned8
NodeId ::= Unsigned16
ConnectionInfo ::= Unsigned8
VendorCode ::= Unsigned16
ModelCode ::= Unsigned32
PortNumber ::= Unsigned8
TraControl ::= Unsigned8
PortStatus ::= Unsigned8
LeaveTimer ::= Unsigned16
PortUsage ::= Unsigned8
SlaveNodeInfo ::= Unsigned8
Version ::= Unsigned8
DeviceType ::= Unsigned16
AvailableFuncs ::= Unsigned8
SeqNumber ::= Unsigned8
NetNumber ::= Unsigned8
PortStatistics ::= Unsigned8
ErrorCode ::= Unsigned32
ParamFlag ::= Unsigned8
ParamDate ::= Unsigned32
ParamTime ::= Unsigned32

```

```

ParamChecksum ::= Unsigned32
Timer ::= Unsigned48
TraDataSubType ::= Unsigned16
TraDataId ::= Unsigned8
TraReturnValue ::= Unsigned16
TraWholeDataSize ::= Unsigned16
TraOffsetAddr ::= Unsigned32
TraDataSize ::= Unsigned16
TraCommand ::= Unsigned8
TraSubCommand ::= Unsigned8
TraLength ::= Unsigned16
TraType ::= Unsigned8
TraFrameSequence ::= Unsigned8
TraDataFrameType ::= Unsigned8
TraDstAddr ::= Unsigned8
TraSrcAddr ::= Unsigned8
TraDstAppType ::= Unsigned8
TraSrcAppType ::= Unsigned8
TraDstModuleFlag ::= Unsigned8
TraSrcModuleFlag ::= Unsigned8
TraDstNetAddr ::= Unsigned8
TraDstStaNo ::= Unsigned8
TraDstID ::= Unsigned16
TraSrcNetAddr ::= Unsigned8
TraSrcStaNo ::= Unsigned8
TraSrcID ::= Unsigned16
TraCmdLen ::= Unsigned16
TraCmdType ::= Unsigned8
TraDataNo ::= Unsigned8
TraAppSeq ::= Unsigned16
TraReturnStatus ::= Unsigned16
Tra2Data ::= LOctetString (SIZE(0..960))
BothEndsValidity ::= Unsigned8
CycOffsetAddr ::= Unsigned32
CycData ::= LOctetString
OctetString ::= OCTET STRING
BitString8 ::= OCTET STRING (SIZE (1))
BitString16 ::= OCTET STRING (SIZE (2))
BitString32 ::= OCTET STRING (SIZE (3))
Unsigned8 ::= OCTET STRING (SIZE (1))
Unsigned16 ::= OCTET STRING (SIZE (2))
Unsigned24 ::= OCTET STRING (SIZE (3))
Unsigned32 ::= OCTET STRING (SIZE (4))
Unsigned48 ::= OCTET STRING (SIZE (6))
MACAddress ::= OCTET STRING (SIZE (6))
LOctetString ::= OCTET STRING
SyncOffset ::= Unsigned32
SyncFlag ::= Unsigned8

```

5 FAL transfer syntax

5.1 Encoding rules

5.1.1 Unsigned encoding

The fixed-length, unsigned values Unsigned8, Unsigned16, Unsigned24, and Unsigned32 are encoded as unsigned integers of one octet, two octets, three octets, and four octets in length, respectively. Unsigned16, Unsigned24, and Unsigned32 are encoded as big endians, where the most significant octet is regarded as the first octet, the next octet is regarded as the second octet, and the least significant octet is regarded as the last octet.

5.1.2 Octet string encoding

OctetString which has variable length of octets is encoded octet by octet, in sequential order.

5.1.3 SEQUENCE encoding

SEQUENCE (and SEQUENCE OF) encoding is performed in sequence, starting from the initial element. The identifiers and length used in ASN.1 are not used.

5.1.4 LOctetString encoding

LOctetString has variable length and is encoded as little endian, where the lowest, or least significant, octet is ordered as the first octet, and follow in sequential order until the highest, or most significant, octet is ordered last.

5.2 FALPDU type C elements encoding

5.2.1 FALARHeader

5.2.1.1 arFType

This field shows the PDU types described in Table 4.

Table 4 – afFType

Value	Description
0x00	Connect-PDU
0x01	ConnectAck-PDU
0x02	Scan-PDU
0x03	Collect-PDU
0x04	Select-PDU
0x05	Launch-PDU
0x06	Token-PDU
0x07..0x09	Reserved
0x10	Persuasion-PDU
0x11	TestData-PDU
0x12	TestDataAck-PDU
0x13	Setup-PDU
0x14	SetupAck-PDU
0x15	Token-M-PDU
0x16..0x1F	Reserved
0x20	MyStatus-PDU
0x22	Transient1-PDU
0x23	Reserved
0x24	Dummy-PDU
0x25	Transient2-PDU
0x26..0x2E	Reserved
0x2F	NTNTest-PDU
0x30..0x7F	Reserved
0x80	CyclicDataW-PDU
0x81	CyclicDataB-PDU
0x82..0x8B	Reserved
0x8C	CyclicDataOut1-PDU
0x8D	CyclicDataOut2-PDU

Value	Description
0x8E	CyclicDataIn1-PDU
0x8F	CyclicDataIn2-PDU

5.2.1.2 priority

This field shows priorities described in Table 5.

Table 5 – priority

Value	Description
0x00	Select-PDU, Launch-PDU, Token-PDU, Dummy-PDU, CyclicDataW-PDU, CyclicDataB-PDU, CyclicDataOut1-PDU, CyclicDataOut2-PDU, CyclicDataIn1-PDU, CyclicDataIn2-PDU
0x01	Collect-PDU, Connect-PDU, ConnectAck-PDU, Scan-PDU
0x02	MyStatus-PDU
0x03	Transient1-PDU
0x04	Transient2-PDU, NTNTest-PDU
0x05	Reserved
0x06	Transient1-PDU

5.2.1.3 scanNumber

This field contains a frame identifier used in Scan-PDU. The number is incremented each time when sending Scan-PDU. The default value of 0x000000 is used in all other PDUs.

5.2.1.4 reserved1

This field is reserved for future use. The value is 0x00.

5.2.1.5 srcNodeNumber

This field contains the identifier number of the source node.

5.2.1.6 reserved2

This field is reserved for future use. The value is 0x0000.

5.2.1.7 hec

This field is an error checking code from DestAddr of DLPDU to reserved2 of FALARHeader. DLPDU definitions are as follows.

```

DLPDU ::= SEQUENCE {
    preamble          Preamble,
    sfd               SFD,
    destaddr          DestAddr,
    srcaddr           SrcAddr,
    lt                LT,
    dlsdu             FAL-PDU,
    fcs               FCS
}
    
```

```

Preamble ::= OctetString SIZE(7)
SFD ::= OctetString SIZE(1)
DestAddr ::= MACAddress
SrcAddr ::= MACAddress
LT ::= Unsigned16
FCS ::= OctetString SIZE(4)
    
```

The generating polynomial is $x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x^1+1$.

5.2.2 Connect-PDU

5.2.2.1 falArHeader

Refer to 5.2.1.

5.2.2.2 portChoice

This field shows the port types as described in Table 6.

Table 6 – portChoice

Value	Description
0x0	"In" side
0x1	"Out" side
0xF	"Out" side, during Test between nodes

5.2.2.3 padding

This field is padding. The value is 0x00.

5.2.2.4 dcs

This field is an error checking code from DestAddr of DLPDU. For DLPDU definitions, refer to 5.2.1.7.

The generating polynomial is $x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x^1+1$.

5.2.3 ConnectAck-PDU

5.2.3.1 falArHeader

Refer to 5.2.1.

5.2.3.2 portCheckResult

This field shows the check results of In-Out connection. The values are specified in Table 7.

Table 7 – portCheckResult

Value	Description
0x0	OK
0x1	NG
0xE	Node-to-node test OK
0xF	Node-to-node test NG

5.2.3.3 dstPortInfo

This field shows the destination port types. The values are specified in Table 8.

Table 8 – dstPortInfo

Value	Description
0	"In" side
1	"Out" side

5.2.3.4 padding

This field is padding. The value is 0x00.

5.2.3.5 dcs

Refer to 5.2.2.4.

5.2.4 Scan-PDU

5.2.4.1 falArHeader

Refer to 5.2.1.

5.2.4.2 scanState

This field shows the status of transmission path. The values are specified in Table 9.

Table 9 – scanState

Value	Description
0	Through status
1	Loopback status of "In" side
2	Loopback status of "Out" side