



BSI Standards Publication

Industrial communication networks — Fieldbus specifications

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

National foreword

This British Standard is the UK implementation of EN IEC 61158-6-23:2023. It is identical to IEC 61158-6-23:2023. It supersedes BS EN IEC 61158-6-23:2019, which is withdrawn.

The UK participation in its preparation was entrusted to Technical Committee GEL/65, Measurement and control.

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

Contractual and legal considerations

This publication has been prepared in good faith, however no representation, warranty, assurance or undertaking (express or implied) is or will be made, and no responsibility or liability is or will be accepted by BSI in relation to the adequacy, accuracy, completeness or reasonableness of this publication. All and any such responsibility and liability is expressly disclaimed to the full extent permitted by the law.

This publication is provided as is, and is to be used at the recipient's own risk.

The recipient is advised to consider seeking professional guidance with respect to its use of this publication.

This publication is not intended to constitute a contract. Users are responsible for its correct application.

© The British Standards Institution 2023
Published by BSI Standards Limited 2023

ISBN 978 0 539 26665 8

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 31 August 2023.

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 61158-6-23:2023)

Réseaux de communication industriels - Spécifications des
bus de terrain - Partie 6-23: Spécification du protocole de la
couche liaison de données - Eléments de type 23
(IEC 61158-6-23:2023)

Industrielle Kommunikationsnetze - Feldbusse - Teil 6-23:
Protokollspezifikation des Application Layer
(Anwendungsschicht) - Typ 23-Elemente
(IEC 61158-6-23:2023)

This European Standard was approved by CENELEC on 2023-04-28. 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, Türkiye 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/1204/FDIS, future edition 3 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 61158-6-23:2023.

The following dates are fixed:

- latest date by which the document has to be implemented at national (dop) 2024-01-28 level by publication of an identical national standard or by endorsement
- latest date by which the national standards conflicting with the (dow) 2026-04-28 document have to be withdrawn

This document supersedes EN IEC 61158-6-23:2019 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.

Any feedback and questions on this document should be directed to the users' national committee. A complete listing of these bodies can be found on the CENELEC website.

Endorsement notice

The text of the International Standard IEC 61158-6-23:2023 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 standard indicated:

IEC 61158-6 (series)	NOTE Approved as EN 61158-6 (series)
IEC 61784-1 (series)	NOTE Approved as EN IEC 61784-1 (series)
IEC 61784-2 (series)	NOTE Approved as EN IEC 61784-2 (series)

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	2023	Industrial communication networks - Fieldbus specifications - Part 1: Overview and guidance for the IEC 61158 and IEC 61784 series	-	-
IEC 61158-5-23	2023	Industrial communication networks - Fieldbus specifications - Part 5-23: Application layer service definition - Type 23 elements	EN IEC 61158-5-23	2023
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) - Part 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	-	-
IEEE 802.1AS	-	IEEE Standard for Information technology - Telecommunications and information exchange between systems - IEEE standard for Local and metropolitan area networks - Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks	-	-
IEEE Std 1588	-	Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems	-	-
IETF RFC 768	1980	User Datagram Protocol	-	-
IETF RFC 791	1981	Internet Protocol	-	-

This page deliberately left blank

CONTENTS

FOREWORD.....	14
INTRODUCTION.....	16
1 Scope.....	17
1.1 General.....	17
1.2 Specifications	18
1.3 Conformance	18
2 Normative references	18
3 Terms, definitions, symbols, abbreviated terms and conventions	19
3.1 Referenced terms and definitions.....	19
3.1.1 ISO/IEC 7498-1 terms.....	19
3.1.2 ISO/IEC 8822 terms.....	19
3.1.3 IEC 61158-1 terms.....	19
3.2 Additional Type 23 terms and definitions.....	19
3.3 Symbols and abbreviated terms	22
3.4 Conventions.....	23
3.4.1 General concept	23
3.4.2 Convention for the encoding of reserved bits and octets	23
3.4.3 Conventions for abstract syntax description	23
3.4.4 Conventions for bit description in octets	23
3.4.5 Conventions for state machine descriptions	24
4 FAL syntax description	25
4.1 FALPDU type C abstract syntax	25
4.1.1 Basic abstract syntax.....	25
4.1.2 Connect-PDU	26
4.1.3 ConnectAck-PDU.....	26
4.1.4 Scan-PDU	26
4.1.5 Collect-PDU.....	27
4.1.6 Select-PDU.....	27
4.1.7 Launch-PDU	27
4.1.8 Token-PDU.....	27
4.1.9 MyStatus-PDU.....	28
4.1.10 Transient1-PDU	28
4.1.11 Dummy-PDU.....	29
4.1.12 Transient2-PDU	29
4.1.13 NTNTest-PDU.....	29
4.1.14 CyclicDataW-PDU.....	30
4.1.15 CyclicDataB-PDU	30
4.1.16 CyclicDataOut1-PDU	30
4.1.17 CyclicDataOut2-PDU	31
4.1.18 CyclicDataIn1-PDU	31
4.1.19 CyclicDataIn2-PDU.....	31
4.2 FALPDU type F abstract syntax	32
4.2.1 Basic abstract syntax.....	32
4.2.2 Persuasion-PDU	33
4.2.3 TestData-PDU	34
4.2.4 TestDataAck-PDU.....	34
4.2.5 Setup-PDU	35

4.2.6	SetupAck-PDU.....	35
4.2.7	F-Token-PDU	36
4.2.8	F-MyStatus-PDU.....	36
4.2.9	Measure-PDU	36
4.2.10	F-Offset-PDU.....	37
4.2.11	F-Update-PDU	37
4.2.12	F-CyclicData-PDU	37
4.2.13	Transient1-PDU	37
4.2.14	TransientAck-PDU	41
4.2.15	Transient2-PDU	42
4.2.16	ParamCheck-PDU	42
4.2.17	Parameter-PDU	43
4.2.18	Timer-PDU	44
4.3	Data type assignments for type C	45
4.4	Data type assignments for type F	46
4.5	FALPDU type T abstract syntax	47
4.5.1	Basic abstract syntax.....	47
4.5.2	CyclicM-PDU	50
4.5.3	CyclicS-PDU.....	50
4.5.4	CyclicMs-PDU	50
4.5.5	CyclicSs-PDU	51
4.5.6	AcyclicPriority-PDU	52
4.5.7	AcyclicDetection-PDU.....	52
4.5.8	AcyclicDetectionAck-PDU	53
4.5.9	AcyclicTestDataHeader	54
4.5.10	AcyclicTestDataHeader	55
4.5.11	AcyclicData-PDU	55
4.5.12	PtpSync-PDU	55
4.5.13	PtpPdelayReq-PDU	55
4.5.14	PtpPdelayResp-PDU.....	55
4.5.15	PtpFollowUp-PDU.....	56
4.5.16	PtpPdelayRespFollowUp-PDU	56
4.5.17	PtpAnnounce-PDU.....	56
4.5.18	SImpIPAddressSet-PDU	56
4.5.19	SImpNetworkConfigMain-PDU	56
4.5.20	SImpNetworkConfigTslt-PDU	56
4.5.21	SImpNotification-PDU	56
4.5.22	SImpMasterConfig-PDU	57
4.5.23	SImpSlaveConfig-PDU	57
4.5.24	SImpCyclicConfigMain-PDU.....	57
4.5.25	SImpCyclicConfigTrnSubPayload-PDU	57
4.5.26	SImpCyclicConfigRcvSubPayload-PDU.....	57
4.5.27	SImpCyclicConfigRcvSrcInfo-PDU	58
4.5.28	SImpLinkDevicePrmWrite-PDU	58
4.5.29	SImpLinkDevicePrmWriteCheckRequest-PDU.....	58
4.5.30	SImpLinkDevicePrmWriteCheckResponse-PDU	58
4.5.31	SImpNMTStateUpload-PDU	58
4.5.32	SImpNMTStateDownload-PDU.....	59
4.5.33	SImpReadObject-PDU	

4.5.34	SImpWriteObject-PDU	59
4.5.35	SImpObjectSubIDReadBlock-PDU	59
4.5.36	SImpObjectSubIDWriteBlock-PDU	59
4.5.37	SImpGetODList-PDU	59
4.5.38	SImpGetObjectDescription-PDU	60
4.5.39	SImpGetEntryDescription-PDU	60
4.5.40	SImpStopOwnStationCyclic-PDU	60
4.5.41	SImpStartOwnStationCyclic-PDU	60
4.5.42	SImpStopOtherStationsCyclic-PDU	60
4.5.43	SImpStartOtherStationsCyclic-PDU	60
4.5.44	SImpAllParameterGet-PDU	61
4.5.45	SImpParameterGet-PDU	61
4.5.46	SImpAllParameterSizeGet-PDU	61
4.5.47	SImpParameterSizeGet-PDU	61
4.5.48	SImpStationSubIDListGet-PDU	61
4.5.49	SImpDeviceIdentificationInfoGet-PDU	61
4.5.50	SImpDataMonitoring-PDU	62
4.5.51	SImpAllParameterSet-PDU	62
4.5.52	SImpParameterSet-PDU	62
4.5.53	SImpParameterVersionCheck-PDU	62
4.5.54	SImpDeviceIdentificationInfoCompare-PDU	62
4.5.55	SImpNodeSearch-PDU	62
4.5.56	SImpIPAddressSet-PDU	63
4.5.57	SImpDeviceInfoCompare-PDU	63
4.5.58	SImpParameterGet-PDU	63
4.5.59	SImpParameterSet-PDU	63
4.5.60	SImpParameterSetStart-PDU	63
4.5.61	SImpParameterSetEnd-PDU	63
4.5.62	SImpVerifyCheckCode-PDU	64
4.5.63	SImpOutputMapFileNameGet-PDU	64
4.5.64	SImpNewFile-PDU	64
4.5.65	SImpParameterSetCancel-PDU	64
4.5.66	SImpOpenFile-PDU	64
4.5.67	SImpCloseFile-PDU	64
4.5.68	SImpReadFile-PDU	65
4.5.69	SImpWriteFile-PDU	65
4.5.70	SImpStatusRead-PDU	65
4.5.71	SImpCommunicationSettingGet-PDU	65
4.5.72	SImpGetDeviceInfo-PDU	65
4.5.73	SImpGetBackupListFileName-PDU	65
4.5.74	SImpStartBackup-PDU	66
4.5.75	SImpEndBackup-PDU	66
4.5.76	SImpCheckRestoreVersion-PDU	66
4.5.77	SImpStartRestore-PDU	66
4.5.78	SImpEndRestore-PDU	66
4.5.79	SImpStatusRead2-PDU	66
4.5.80	SImpReqSearchNode-PDU	67
4.5.81	SImpGetSearchNodeState-PDU	67
4.5.82	SImpGetNodeList-PDU	67

4.5.83	SImpReqSetIPAddress-PDU	67
4.5.84	SImpSearchPrmControlStation-PDU	67
4.5.85	SImpRequestRestore-PDU	68
4.5.86	SImpCheckPrmDelivery-PDU	68
4.5.87	SImpRsvStationConfigTemporaryRelease-PDU	68
4.5.88	SImpRsvStationConfig-PDU	68
4.5.89	SImpGetEventNum-PDU	68
4.5.90	SImpGetEventHistory-PDU	69
4.5.91	SImpClearEventHistory-PDU	69
4.5.92	SImpClockOffsetDataSend-PDU	69
4.5.93	SImpSetWatchdogCounterInfo-PDU	69
4.5.94	SImpWatchdogCounterOffsetConfig-PDU	69
4.5.95	SImpRemoteReset-PDU	69
4.5.96	SImpGetCommunicationSet-PDU	70
4.5.97	SImpGetStationSubIDList-PDU	70
4.5.98	SImpGetDeviceInfo-PDU	70
4.5.99	SImpStartBackup-PDU	70
4.5.100	SImpEndBackup-PDU	70
4.5.101	SImpRequestBackup-PDU	70
4.5.102	SImpGetBackupPrm-PDU	71
4.5.103	SImpCheckRestore-PDU	71
4.5.104	SImpStartRestore-PDU	71
4.5.105	SImpEndRestore-PDU	71
4.5.106	SImpSetBackupPrm-PDU	71
4.5.107	SImpLinkupSpeed-PDU	71
4.5.108	SImpNodeIndication-PDU	72
4.6	Data type assignments for type T	72
5	FAL transfer syntax	72
5.1	Encoding rules	72
5.1.1	Unsigned encoding	72
5.1.2	Octet string encoding	72
5.1.3	SEQUENCE encoding	72
5.1.4	LOctetString encoding	72
5.2	FALPDU type C elements encoding	73
5.2.1	FALARHeader	73
5.2.2	Connect-PDU	75
5.2.3	ConnectAck-PDU	76
5.2.4	Scan-PDU	76
5.2.5	Collect-PDU	77
5.2.6	Select-PDU	79
5.2.7	Launch-PDU	80
5.2.8	Token-PDU	80
5.2.9	MyStatus-PDU	80
5.2.10	Transient1-PDU	82
5.2.11	Dummy-PDU	86
5.2.12	Transient2-PDU	87
5.2.13	NTNTest-PDU	98
5.2.14	CyclicDataW-PDU	98
5.2.15	CyclicDataB-PDU	99

5.2.16	CyclicDataOut1-PDU	100
5.2.17	CyclicDataOut2-PDU	100
5.2.18	CyclicDataIn1-PDU	101
5.2.19	CyclicDataIn2-PDU	102
5.3	FALPDU type F elements encoding	103
5.3.1	FALARHeader	103
5.3.2	Persuasion-PDU	108
5.3.3	TestData-PDU	109
5.3.4	TestDataAck-PDU	109
5.3.5	Setup-PDU	111
5.3.6	SetupAck-PDU	113
5.3.7	F-Token-PDU	114
5.3.8	F-Measure-PDU	115
5.3.9	F-Offset-PDU	116
5.3.10	F-Update-PDU	116
5.3.11	F-MyStatus-PDU	116
5.3.12	F-CyclicData-PDU	122
5.3.13	Transient1-PDU	123
5.3.14	TransientAck-PDU	128
5.3.15	Transient2-PDU	129
5.3.16	ParamCheck-PDU	132
5.3.17	Parameter-PDU	133
5.3.18	Timer-PDU	140
5.4	FALPDU type T elements encoding	141
5.4.1	CyclicM-PDU	141
5.4.2	CyclicS-PDU	144
5.4.3	CyclicMs-PDU	146
5.4.4	CyclicSs-PDU	147
5.4.5	AcyclicPriority-PDU	148
5.4.6	AcyclicDetection-PDU	150
5.4.7	AcyclicDetectionAck-PDU	151
5.4.8	AcyclicTestData-PDU	157
5.4.9	AcyclicTestDataAck-PDU	160
5.4.10	AcyclicData-PDU	162
5.4.11	Ptp-PDU	162
5.4.12	IpData-PDU	164
6	Structure of the FAL protocol state machine	164
7	FAL service protocol machine (FSPM)	165
7.1	Overview	165
7.2	FSPM type C	165
7.2.1	Overview	165
7.2.2	FSPM	166
7.3	FSPM type F	169
7.3.1	Overview	169
7.3.2	FSPM	171
7.4	FSPM type T	176
7.4.1	Overview	176
7.4.2	FSPM State Machine	177
8	Application relationship protocol machine (ARPM)	181

8.1	ARPM type C	181
8.1.1	Overview	181
8.1.2	Acyclic transmission	182
8.1.3	Cyclic transmission	183
8.1.4	Connection control	188
8.1.5	Common parameter dist.	227
8.2	ARPM type F	232
8.2.1	Overview	232
8.2.2	Acyclic transmission	233
8.2.3	Cyclic transmission	235
8.2.4	Channel control	238
8.2.5	Parameter dist	278
8.2.6	Synchronous trigger	282
8.2.7	Timer	283
8.2.8	Measure transmission	284
8.3	ARPM type T	289
8.3.1	Overview	289
8.3.2	Cyclic Transmission	289
8.3.3	Acyclic Transmission	291
8.3.4	Channel Control	294
8.3.5	TimeSync Control	297
8.3.6	IPTrans Control	299
8.3.7	Handler	300
9	DLL mapping protocol machine (DMPM)	305
9.1	DMPM type C	305
9.2	DMPM type F	306
9.3	DMPM type T	307
	Bibliography	309
	Figure 1 – Bit description in octets	24
	Figure 2 – Structure for memory access information retrieve response	90
	Figure 3 – Attribute definitions	90
	Figure 4 – Access code definitions	91
	Figure 5 – Structure for RUN request	92
	Figure 6 – Structure for RUN response	93
	Figure 7 – Structure for STOP request	93
	Figure 8 – Structure for STOP response	93
	Figure 9 – Structure for batch memory read request	94
	Figure 10 – Structure for batch memory read response	94
	Figure 11 – Structure for random memory read request	95
	Figure 12 – Structure for random memory read response	95
	Figure 13 – Structure for batch memory write request	96
	Figure 14 – Structure for batch memory write response	96
	Figure 15 – Structure for random memory write request	97
	Figure 16 – Structure for random memory write response	97
	Figure 17 – Relationships between protocol machines	165

Figure 18 – Structure of FSPM C	166
Figure 19 – Structure of FSPM F	169
Figure 20 – Structure of FSPM T	176
Figure 21 – Structure of ARPM C	181
Figure 22 – Structure of ARPM F	232
Figure 23 – Structure of ARPM T	289
Figure 24 – Structure of type C DMPM	305
Figure 25 – Structure of type F DMPM	306
Figure 26 – Structure of type T DMPM	307
Table 1 – State machine description elements	24
Table 2 – Description of state machine elements	24
Table 3 – Conventions used in state machines	25
Table 4 – afFType	73
Table 5 – priority	74
Table 6 – portChoice	75
Table 7 – portCheckResult	76
Table 8 – dstPortInfo	76
Table 9 – scanState	76
Table 10 – nodeType	77
Table 11 – loopState	78
Table 12 – Cyclic status	78
Table 13 – Parameter setting mode	78
Table 14 – opState	81
Table 15 – errorState	81
Table 16 – Data type	83
Table 17 – CPW	83
Table 18 – CPWC	84
Table 19 – CPWCR	84
Table 20 – cmParam	84
Table 21 – Details of param area	85
Table 22 – Details of application parameters	85
Table 23 – Details of LB/LW CM area and LB/LW CM additional area	86
Table 24 – Details of LX/LY CM 1 area and LX/LY CM 2 area	86
Table 25 – Destination module flag	88
Table 26 – Command types	89
Table 27 – Access codes of network module memory	91
Table 28 – Access codes of controller memory	92
Table 29 – byteValidity	98
Table 30 – afFType	104
Table 31 – dataType	105
Table 32 – varField	106
Table 33 – nodeType	107

Table 34 – ProtocolVerType.....	108
Table 35 – Link status.....	111
Table 36 – Port enable/disable specification	112
Table 37 – Cyclic transmission parameter hold status.....	118
Table 38 – Detailed application operation status	119
Table 39 – Error detection status	119
Table 40 – Slave-specific event reception status.....	121
Table 41 – dataSupType of dataType (0x07).....	124
Table 42 – FieldSpecificTransient opHeader.....	124
Table 43 – command (dataType: 0x07, dataSubType: 0x0002)	125
Table 44 – subCommand type for each command type	125
Table 45 – Structure of Deliver node information	125
Table 46 – Structure of Deliver node information – message.....	126
Table 47 – Structure of Get statistical information response.....	127
Table 48 – Structure of Acquisition of node details response	128
Table 49 – Execution module specification.....	130
Table 50 – Command type	131
Table 51 – frameType	141
Table 52 – cycleNo	141
Table 53 – sa.....	141
Table 54 – da.....	142
Table 55 – commInfo	142
Table 56 – txAsynInfo	142
Table 57 – seqNo	143
Table 58 – Upper one octet of diagnosisData.....	143
Table 59 – Lower three octets of diagnosisData.....	144
Table 60 – Upper one octet of diagnosisData.....	145
Table 61 – Lower three octets of diagnosisData.....	145
Table 62 – mngPriority.....	149
Table 63 – KindFlag.....	149
Table 64 – previousNodePort.....	151
Table 65 – optionFlag	151
Table 66 – sendInfo	151
Table 67 – nodeType	152
Table 68 – IP address 4th octet	152
Table 69 – detectionRcvPort.....	153
Table 70 – myPort	153
Table 71 – Four bits of myPortLinkStatus.....	153
Table 72 – Four bits of myPortFilterStatus	154
Table 73 – performance	154
Table 74 – gmPriority.....	155
Table 75 – syncType.....	155
Table 76 – cyclicSize	156

Table 77 – function	157
Table 78 – optionInfo	157
Table 79 – Protocol version	158
Table 80 – Protocol type	159
Table 81 – Source information	159
Table 82 – Link status	162
Table 83 – Cyclic data state table	167
Table 84 – Acyclic data state table	167
Table 85 – Management state table	169
Table 86 – Cyclic data state table	172
Table 87 – Acyclic data state table	172
Table 88 – Management state table	175
Table 89 – Synchronization state table	175
Table 90 – Measurement state table	176
Table 91 – Primitives provided by FSPM	177
Table 92 – Cyclic Data state	177
Table 93 – Cyclic Data state table	178
Table 94 – Function used for Cyclic Data	178
Table 95 – Acyclic Data state	178
Table 96 – Acyclic Data state table	178
Table 97 – Management state	179
Table 98 – Management state table	180
Table 99 – TimeSync Data state	180
Table 100 – TimeSync Data state table	180
Table 101 – SLMP Data state	180
Table 102 – SLMP Data state table	181
Table 103 – Acyclic transmission state table	182
Table 104 – Acyclic transmission functions	183
Table 105 – Cyclic transmission state table	183
Table 106 – Cyclic transmission functions	188
Table 107 – Connection control state machine – Initial	189
Table 108 – Connection control state machine – Connect	189
Table 109 – Connection control state machine – Scan	191
Table 110 – Connection control state machine – ScanWait	194
Table 111 – Connection control state machine – Collect	197
Table 112 – Connection control state machine – CollectWait	200
Table 113 – Connection control state machine – Select	203
Table 114 – Connection control state machine – TokenStartWait	206
Table 115 – Connection control state machine – LaunchWait	209
Table 116 – Connection control state machine – TokenReleaseWait	212
Table 117 – Connection control state machine – TokenReleased	215
Table 118 – Connection control state machine – TokenWait	221
Table 119 – Connection control state machine – NTNTestMaster	226

Table 120 – Connection control state machine – NTNTestSlave	226
Table 121 – Function list of connection control	227
Table 122 – Common parameter dist state table	227
Table 123 – Function list of connection control	231
Table 124 – Mapping of internal service and acyclic transmission service	232
Table 125 – Acyclic transmission states	233
Table 126 – Acyclic transmission state table	233
Table 127 – Acyclic transmission functions	235
Table 128 – Acyclic transmission variables	235
Table 129 – Cyclic transmission states	236
Table 130 – Cyclic transmission state table	236
Table 131 – Cyclic transmission functions	238
Table 132 – Cyclic transmission variables	238
Table 133 – Master station channel control states	239
Table 134 – Slave station channel control states	239
Table 135 – Master station state table – MasterDown	239
Table 136 – Master station state table – Listen	240
Table 137 – Master station state table – MasterArbitration	241
Table 138 – Master station state table – PrimaryMasterScatterTD	242
Table 139 – Master station state table – PrimaryMasterSettingUp	245
Table 140 – Master station state table – PrimaryMasterHoldToken	248
Table 141 – Master station state table – PrimaryMasterSolicitToken	251
Table 142 – Master station state table – PrimaryMasterInviting	254
Table 143 – Master station state table – MasterWaitTD	256
Table 144 – Master station state table – MasterWaitSetup	257
Table 145 – Master station state table – MasterSolicitToken (without Transmission path delay measurement)	258
Table 146 – Master station state table – MasterSolicitToken (with Transmission path delay measurement)	260
Table 147 – Master station state table – MasterHoldToken	263
Table 148 – Master station state table – MasterMeasurement (without Transmission path delay measurement function)	266
Table 149 – Master station state table – MasterMeasurement (with Transmission path delay measurement function)	266
Table 150 – Slave station state table – SlaveDown	266
Table 151 – Slave station state table – SlaveWaitTD	267
Table 152 – Slave station state table – SlaveWaitSetup	268
Table 153 – Slave station state table – SlaveSolicitToken (without Transmission path delay measurement)	269
Table 154 – Slave station state table – SlaveSolicitToken (with Transmission path delay measurement)	271
Table 155 – Slave station state table – SlaveHoldToken	273
Table 156 – Master station channel control functions	276
Table 157 – Slave station channel control functions	277
Table 158 – Master station channel control variables	

Table 159 – Slave station channel control variables	278
Table 160 – Master station channel control timers	278
Table 161 – Slave station channel control timers	278
Table 162 – Master station parameter dist states	279
Table 163 – Slave station parameter dist states	279
Table 164 – Master station parameter dist state table	279
Table 165 – Slave station parameter dist state table	280
Table 166 – Master station parameter dist functions	281
Table 167 – Slave station parameter dist functions	282
Table 168 – Master station synchronous trigger states.....	282
Table 169 – Slave station synchronous trigger states.....	282
Table 170 – Master station synchronous trigger state table.....	282
Table 171 – Slave station synchronous trigger state table.....	283
Table 172 – Synchronous trigger functions	283
Table 173 – Timer states – Best effort type.....	283
Table 174 – Timer states – Fixed cycle type	283
Table 175 – Timer state table – Best effort type	283
Table 176 – Timer state table – Fixed cycle type	284
Table 177 – Timer variables.....	284
Table 178 – Fixed cycle timer	284
Table 179 – Master station measure transmission states	285
Table 180 – Slave station measure transmission states	285
Table 181 – Master station measure transmission state table	285
Table 182 – Slave station measure transmission state table	287
Table 183 – Master station measure transmission functions.....	287
Table 184 – Slave station measure transmission functions.....	288
Table 185 – Master station measure transmission variables.....	289
Table 186 – Primitives provided by Cyclic Transmission	289
Table 187 – Cyclic Transmission state	290
Table 188 – Cyclic Transmission state table	290
Table 189 – Functions used for Cyclic Transmission.....	290
Table 190 – Variables used in Cyclic Transmission.....	291
Table 191 – Primitives provided by AcyclicTransmission.....	291
Table 192 – Acyclic Transmission state	291
Table 193 – Acyclic Transmission state table.....	291
Table 194 – Functions used for Acyclic Transmission	294
Table 195 – Variables used in Acyclic Transmission	294
Table 196 – Primitives provided by Channel Control	295
Table 197 – Channel Control state.....	295
Table 198 – Channel Control state machine.....	295
Table 199 – Functions used for Channel Control.....	297
Table 200 – Variables used in Channel Control.....	297
Table 201 – Primitives provided by TimeSync Control.....	297

Table 202 – TimeSync Control state	298
Table 203 – TimeSync Control state table	298
Table 204 – Functions used for TimeSync Control	298
Table 205 – Variables used in TimeSync Control	299
Table 206 – Primitives provided by IPTrans Control	299
Table 207 – IPTrans Control state in master station	299
Table 208 – IPTrans Control state table	299
Table 209 – Functions used for IPTrans Control	300
Table 210 – Variables used in IPTrans Control	300
Table 211 – Handler state	300
Table 212 – DOWN state machine	301
Table 213 – INIT state machine	301
Table 214 – RUN state machine	303
Table 215 – Functions used for Handler	304
Table 216 – Variables used for Handler	304
Table 217 – Mapping of type C DMPM service and DL service	305
Table 218 – Destination address for each type C PDU	306
Table 219 – Mapping of type F DMPM service and DL service	307
Table 220 – Primitives provided by DMPM	307
Table 221 – Cyclic Data state	307
Table 222 – DMPM state table	308

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 the IEC 61784-1 series and the IEC 61784-2 series.

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

This third edition cancels and replaces the second edition published in 2019. This edition constitutes a technical revision.

This edition includes the following significant technical changes with respect to the previous edition:

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

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

Draft	Report on voting
65C/1204/FDIS	65C/1245/RVD

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

The language used for the development of this International Standard is English.

This document was drafted in accordance with ISO/IEC Directives, Part 2, and developed in accordance with ISO/IEC Directives, Part 1 and ISO/IEC Directives, IEC Supplement, available at www.iec.ch/members_experts/refdocs. The main document types developed by IEC are described in greater detail at www.iec.ch/publications.

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 document will remain unchanged until the stability date indicated on the IEC website under webstore.iec.ch in the data related to the specific document. At this date, the document will be

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

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

The holder of this patent right has assured IEC that s/he is willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of this patent right is registered with IEC. Information may be obtained from the patent database available at <http://patents.iec.ch>.

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

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 document 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 the IEC 61158-6 series.

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 series and IEC 61784-2 series 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 61158-1:2023, *Industrial communication networks – Fieldbus specifications – Part 1: Overview and guidance for the IEC 61158 and IEC 61784 series*

IEC 61158-5-23:2023, *Industrial communication networks – Fieldbus specifications – Part 5-23: Application layer service definition – Type 23 elements*

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) – Part 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*

IEEE Std 802.1AS, *Standard for Local and Metropolitan Area Networks – Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks*

IEEE Std 1588, *Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*

IETF RFC 768, J. Postel, *User Datagram Protocol*, August 1980, available at <https://www.rfc-editor.org/info/rfc768> [viewed 2022-02-18]

IETF RFC 791, J. Postel, *Internet Protocol*, September 1981, available at <https://www.rfc-editor.org/info/rfc791> [viewed 2022-02-18]

3 Terms, definitions, symbols, abbreviated terms and conventions

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

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

- IEC Electropedia: available at <https://www.electropedia.org/>
- ISO Online browsing platform: available at <https://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) 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

3.1.3 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



Documents

Document Sharing Hub

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.

MSB							LSB	
7	6	5	4	3	2	1	0	Bit identification number

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..1 482),
    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 ..1 466)),
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)),
    cmRxBBothEndsValidity  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)
```