



BSI Standards Publication

Industrial communication networks — Fieldbus specifications

Part 6-24: Application layer protocol specification — Type 24 elements

National foreword

This British Standard is the UK implementation of EN IEC 61158-6-24:2023. It is identical to IEC 61158-6-24:2023. It supersedes BS EN 61158-6-24:2014, which is withdrawn.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 2024
Published by BSI Standards Limited 2024

ISBN 978 0 539 26664 1

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 March 2024.

Amendments/corrigenda issued since publication

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

English Version

Industrial communication networks - Fieldbus specifications -
Part 6-24: Application layer protocol specification - Type 24
elements
(IEC 61158-6-24:2023)

Réseaux de communication industriels - Spécifications des
bus de terrain - Partie 6-24: Spécification du protocole de la
couche liaison de données - Eléments de type 24
(IEC 61158-6-24:2023)

Industrielle Kommunikationsnetze - Feldbusse - Teil 6-24:
Protokollspezifikation des Application Layer
(Anwendungsschicht) - Typ 24-Elemente
(IEC 61158-6-24:2023)

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 2 of IEC 61158-6-24, 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 61158-6-24:2023.

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 61158-6-24:2014 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.

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 61158-6-24:2023 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 61158-1:2023	NOTE Approved as EN IEC 61158-1:2023 (not modified)
IEC 61158-2	NOTE Approved as EN IEC 61158-2
IEC 61158-3-24	NOTE Approved as EN 61158-3-24
IEC 61158-4-24	NOTE Approved as EN IEC 61158-4-24
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 61158-5-24	2023	Industrial communication networks - Fieldbus specifications - Part 5-24: Application layer service definition - Type 24 elements	EN IEC 61158-5-24	2023
ISO/IEC 646	-	Information technology - ISO 7-bit coded character set for information interchange	-	-
ISO/IEC 7498-1	-	Information technology - Open Systems Interconnection - Basic reference model: The basic model	-	-
ISO/IEC 9545	-	Information technology - Open Systems Interconnection - Application layer structure	-	-
ISO/IEC 9899	-	Information technology - Programming languages - C	-	-
ISO/IEC 10731	-	Information technology - Open Systems Interconnection - Basic Reference Model - Conventions for the definition of OSI services	-	-
ISO/IEC 19501	2005	Information technology - Open Distributed Processing - Unified Modeling Language (UML) Version 1.4.2	-	-
ISO/IEC/IEEE 60559	2020	Information technology - Microprocessor Systems - Floating-Point arithmetic	-	-

CONTENTS

FOREWORD.....	6
INTRODUCTION.....	8
1 Scope.....	9
1.1 General.....	9
1.2 Specifications	9
1.3 Conformance	10
2 Normative references	10
3 Terms, definitions, symbols, abbreviated terms, and conventions	10
3.1 Referenced terms and definitions.....	10
3.1.1 Terms and definitions from ISO/IEC 7498-1	11
3.1.2 Terms and definitions from ISO/IEC 9545	11
3.1.3 Terms and definitions from ISO/IEC 8824-1	11
3.1.4 Terms and definitions from ISO/IEC 10731	11
3.1.5 Terms and definitions from ISO/IEC 19501	12
3.2 Additional terms and definitions	12
3.3 Abbreviations and symbols	17
3.4 Conventions.....	19
3.4.1 General conventions.....	19
3.4.2 PDU data type conventions.....	19
3.4.3 State machine conventions	19
4 Abstract syntax.....	22
4.1 Basic Data types.....	22
4.2 FAL PDU types	23
4.2.1 Top of APDU types: _APDU.....	23
4.2.2 PDUs for field device control service	24
4.2.3 PDUs for message service.....	37
4.3 Detailed definitions of _FDCService-PDUs.....	38
4.3.1 Short PDU type.....	38
4.3.2 Long PDU type	45
4.3.3 Enhanced PDU type	48
4.3.4 SubCommand PDU type	58
4.3.5 Short PDU type II.....	59
4.4 Device profile.....	66
5 Transfer syntax.....	66
5.1 Concepts	66
5.2 Encode rules.....	66
5.2.1 INTEGER and its subtypes	66
5.2.2 REAL type and its subtypes	68
5.2.3 BIT STRING type.....	69
5.2.4 OCTET STRING type and IA5String type	70
5.2.5 NULL type	71
5.2.6 Structure type and Array type	71
6 Structure of FAL protocol state machine	71
7 AP-context state machine (APC SM)	73
7.1 Overview.....	73
7.2 State descriptions	74

7.3	Triggering events	75
7.4	Action descriptions at state transitions	75
8	FAL service protocol machines (FSPM)	77
8.1	Overview	77
8.2	Field Device Control Protocol Machine (FDC PM)	77
8.2.1	Protocol overview	77
8.2.2	Cyclic communication mode	78
8.2.3	Event driven communication mode	82
8.2.4	Master Protocol Machine (FDCPM-M)	83
8.2.5	Slave Protocol Machine (FDCPM-S)	92
8.2.6	Monitor Protocol Machine (FDCPM-MN)	102
8.2.7	Error procedure summary	104
8.3	Message Protocol Machine (MSGPM)	106
8.3.1	Protocol overview	106
8.3.2	Requester Protocol Machine (MSGPM-RQ)	108
8.3.3	Responder Protocol Machine (MSGPM-RS)	111
9	Application relationship protocol machine (ARPM)	113
9.1	General	113
9.2	ARPM for FDC ASE	114
9.2.1	Overview	114
9.2.2	ARPM for FDC Master (ARPM-FDCM)	116
9.2.3	ARPM for FDC Slave (ARPM-FDCS)	122
9.2.4	ARPM for FDC Monitor (ARPM-FDCMN)	129
9.3	ARPM for MSG ASE (ARPM-MSG)	131
9.3.1	State descriptions	131
9.3.2	Triggering events	132
9.3.3	Action descriptions at state transitions	133
10	DLL mapping protocol machine (DMPM)	134
	Annex A (informative) Device profile and FDC command sets	135
	Annex B (normative) Virtual memory space and Device Information	137
B.1	Overview	137
B.2	Device Information	137
B.2.1	Device identifier area structure	137
B.2.2	Detail specifications of device IDs	138
	Annex C (informative) Basic message function	144
	Bibliography	145
	Figure 1 – Tree structure of APDU types	24
	Figure 2 – Encode of Integer subtypes	67
	Figure 3 – Example of transfer of INTEGER value	67
	Figure 4 – Encode of Unsigned subtypes	68
	Figure 5 – <i>Float</i> ₃₂ type encode	68
	Figure 6 – <i>Float</i> ₆₄ type encode	69
	Figure 7 – Bit field definition example with named bits	70
	Figure 8 – Bit field definition example with field size	70
	Figure 9 – SEQUENCE type encode	71

Figure 10 – Structure of FAL protocol state machines	73
Figure 11 – Statechart diagram of APCSM	74
Figure 12 – Example communication cycle of FDC master AP	79
Figure 13 – Example communication cycle of FDC slave AP	79
Figure 14 – Synchronous command communication in sync state	80
Figure 15 – Asynchronous command communication in sync state	81
Figure 16 – Asynchronous command communication in async state	82
Figure 17 – Event-driven communication	83
Figure 18 – Statechart diagram of FDCPM-M	84
Figure 19 – Statechart diagram of FDCPM-S	93
Figure 20 – Statechart diagram of FDCPM-MN	102
Figure 21 – PDU transmission flow for user message	107
Figure 22 – PDU transmission flow for one-way message	108
Figure 23 – Statechart diagram of MSGPM-RQ	109
Figure 24 – Statechart diagram of MSGPM-RS	111
Figure 25 – Example of single transfer process	114
Figure 26 – Example of dual transfer process	114
Figure 27 – Example of Synchronous command communication	115
Figure 28 – Timing chart for individual communication cycle setting	116
Figure 29 – Statechart diagram of ARPM-FDCM	117
Figure 30 – Statechart diagram of ARPM-FDCS	123
Figure 31 – Statechart diagram of ARPM-FDCMN	129
Figure 32 – Statechart diagram of ARPM-MSG	132
Figure B.1 – Memory map of virtual memory space	137
Figure B.2 – Memory map of device ID area	138
Table 1 – State transition descriptions	20
Table 2 – Description of state machine elements	20
Table 3 – Conventions used in state machines	21
Table 4 – Mapping for Protocol State Machines	72
Table 5 – State descriptions of APC SM	74
Table 6 – Trigger event descriptions of APC SM	75
Table 7 – Transitions of APC SM	75
Table 8 – FDC protocol mode	77
Table 9 – State descriptions of FDCPM-M	84
Table 10 – Trigger event descriptions of FDCPM-M	86
Table 11 – Transitions of main SM of FDCPM-M	87
Table 12 – Transitions of submachine of FDCPM-M	89
Table 13 – State descriptions of FDCPM-S	93
Table 14 – Trigger event descriptions of FDCPM-S	94
Table 15 – Transitions of main SM of FDCPM-S	95
Table 16 – Transitions of submachine of FDCPM-S	97
Table 17 – State descriptions of FDCPM-MN	102

Table 18 – Trigger event descriptions of FDCPM-MN.....	103
Table 19 – Transitions of main SM of FDCPM-MN	103
Table 20 – Transitions of submachine of FDCPM-MN	104
Table 21 – State descriptions of MSGPM-RQ.....	109
Table 22 – Trigger event descriptions of MSGPM-RQ	110
Table 23 – Transitions of MSGPM-RQ	110
Table 24 – State descriptions of MSGPM-RS	112
Table 25 – Trigger event descriptions of MSGPM-RS.....	112
Table 26 – Transitions of MSGPM-RS.....	113
Table 27 – State descriptions of ARPM-FDCM	117
Table 28 – Trigger event descriptions of ARPM-FDCM	119
Table 29 – Transitions of main SM of ARPM-FDCM	120
Table 30 – Transitions of submachine of ARPM-FDCM	121
Table 31 – State descriptions of ARPM-FDCS	123
Table 32 – Trigger event descriptions of ARPM-FDCS	125
Table 33 – Transitions of main SM of ARPM-FDCS.....	126
Table 34 – Transitions of submachine of ARPM-FDCS.....	127
Table 35 – State descriptions of ARPM-FDCMN	129
Table 36 – Trigger event descriptions of ARPM-FDCMN	130
Table 37 – Transitions of main SM of ARPM-FDCMN.....	130
Table 38 – Transitions of submachine of ARPM-FDCMN.....	131
Table 39 – State descriptions of ARPM-MSG	132
Table 40 – Trigger event descriptions of ARPM-MSG.....	133
Table 41 – Transitions of ARPM-MSG.....	133
Table A.1 – Example of registered device profiles	135
Table A.2 – Example command list of the profile '00'H	136
Table B.1 – Specifications of device IDs	138
Table C.1 – Example of message command set.....	144

INTERNATIONAL ELECTROTECHNICAL COMMISSION

**INDUSTRIAL COMMUNICATION NETWORKS –
FIELDBUS SPECIFICATIONS –****Part 6-24: Application layer protocol specification –
Type 24 elements****FOREWORD**

- 1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
- 2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
- 3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
- 4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
- 5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.
- 6) All users should ensure that they have the latest edition of this publication.
- 7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
- 8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
- 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 61158-6-24 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 second edition cancels and replaces the first edition published in 2014. This edition constitutes a technical revision.

The main changes with respect to the previous edition are listed below:

- addition of a new PDU type which called "Short PDU type II" in 4.2;
- update of Table 4;
- addition of examples of Synchronous Command communication in 9.2.1, Figure 27 and Figure 28.

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

The committee has decided that the contents of this 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.

INDUSTRIAL COMMUNICATION NETWORKS – FIELDBUS SPECIFICATIONS –

Part 6-24: Application layer protocol specification – Type 24 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 24 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 Type 24 fieldbus application layer in terms of

- the abstract syntax defining the application layer protocol data units conveyed between communicating application entities,
- the transfer syntax defining the application layer protocol data units conveyed between communicating application entities,
- the application context state machines defining the application service behavior visibly between communicating application entities, and
- the application relationship state machines defining the communication behavior visibly between communicating application entities.

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

- define the representation-on-wire of the service primitives defined in IEC 61158-5-24, and
- define the externally visible behavior associated with their transfer.

This document specifies the protocol of the Type 24 fieldbus application layer, in conformance with the OSI Basic Reference Model (ISO/IEC 7498-1) and the OSI Application Layer Structure (ISO/IEC 9545).

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 61158-5-24.

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 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.

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 the IEC 61784-1 series and the 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 61158-5-24:2023, *Industrial communication networks – Fieldbus specifications – Part 5-24: Application layer service definition – Type 24 elements*

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

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

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

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

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

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

ISO/IEC/IEEE 60559:2020, *Information technology – Microprocessor Systems – Floating-Point arithmetic*

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

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

3.1.1 Terms and definitions from ISO/IEC 7498-1

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

- a) abstract syntax;
- b) application-entity;
- c) application process;
- d) application protocol data unit;
- e) application-process-invocation;
- f) (N)-facility;
- g) (N)-function;
- h) peer-(N)-entities;
- i) presentation context;
- j) real system;
- k) transfer syntax.

3.1.2 Terms and definitions from ISO/IEC 9545

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

- a) application-association;
- b) application-context;
- c) application-entity-invocation;
- d) application-entity-type;
- e) application-service-element.

3.1.3 Terms and definitions from ISO/IEC 8824-1

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

- a) simple type;
- b) component;
- c) component type;
- d) integer type;
- e) bitstring type;
- f) octetstring type;
- g) null type;
- h) sequence type;
- i) sequence of type;
- j) choice type;
- k) IA5String type;
- l) encoding.

3.1.4 Terms and definitions from ISO/IEC 10731

For the purposes of this document, the following terms as defined in ISO/IEC 10731 apply:

- a) OSI-service-primitive; primitive;
- b) OSI-service-provider; provider;
- c) OSI-service-user; user.

3.1.5 Terms and definitions from ISO/IEC 19501

For the purposes of this document, the following terms as defined in ISO/IEC 19501 apply:

- a) event;
- b) state;
- c) state machine;
- d) substate;
- e) submachine;
- f) transition.

3.2 Additional terms and definitions

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

3.2.1

alarm

field device status to tell that the device has detected a fatal problem to be solved and cannot continue normal working, through the field device control (FDC) service of the Type 24 fieldbus

Note 1 to entry: Any alarm statuses are latched and need some operations to be cleared.

Note 2 to entry: Alarms are classified into three groups; communication alarms, illegal-command-related ones, and application specific ones. But concrete definitions are dependent on implementation of each field devices.

3.2.2

application process object

network representation of a specific aspect of an application process (AP), which is modelled as a network accessible object contained within an AP or within another APO

Note 1 to entry: Refer to IEC 61158-1, 9.3.4.

3.2.3

application process context

AP context

shared knowledge or a common set of rules, governing communication of FAL application entities (AEs) and describing the permissible collective communications behavior between the AEs that are party to a specific set of application relationships (ARs)

Note 1 to entry: Data within AP context can be specified by the user in advance, by the option selected while the user uses a field bus management (FSM) service to read out the facility of peer AP, by the automatic negotiation function that the FSM system handles, and so on. The method that is to be adopted depends on the specification of each implementation.

3.2.4

application process type

AP type

description of a classification of application processes (APs) in terms of a set of capabilities for FAL of the Type 24 fieldbus

Note 1 to entry: AP types are classified into three, C1 master AP, C2 master AP and slave AP, by their application roles in the fieldbus network.

3.2.5

async command

type of a command application protocol data unit (APDU) of the FDC service of the Type 24 FAL, which can be issued any time after the previous transaction without consideration of synchronization with the communication cycle

Note 1 to entry: Definitions, which command should be async one or not, are dependent on an application. They can be provided as a registered set of commands and responses or device profiles, see 4.4 and Annex

3.2.6**asynchronous communication**

state or a way of communication for the FDC service of the Type 24 FAL, in which a command can be issued any time after the previous transaction without consideration of synchronization with the communication cycle

Note 1 to entry: In this state, sync commands cannot be issued, but async commands can.

3.2.7**attribute**

information or parameter contained in variable portions of an object

Note 1 to entry: Typically, they provide status information or govern the operation of an object. Attributes also affect the behavior of an object.

3.2.8**C1 master**

AP type that has master facilities for the FDC service of the Type 24 FAL, or the device implementing that AP type

Note 1 to entry: Only one C1 master exists in a network of the Type 24 fieldbus.

3.2.9**C2 master**

AP type that has only monitor facilities for the FDC service but requester facilities for message (MSG) service of the Type 24 FAL, or the device implementing that AP type

Note 1 to entry: Less than two C2 masters can exist in a network of the Type 24 fieldbus.

3.2.10**command**

PDU issued by a requester or a master to make a responder or a slave execute some functions

3.2.11**communication**

process to exchange information in a formal manner between two or more devices, users, APs or entities

3.2.12**transfer**

process to convey a PDU from a sender to a receiver

3.2.13**transmission**

process to send out and propagate electrical signals or encoded data

3.2.14**communication cycle**

period of repetitive activities synchronized with the transmission cycle while the connection establishing for the FDC protocol of the Type 24 FAL

Note 1 to entry: Communication cycle can synchronize with the transmission cycle multiplied by a specified scaling factor.

3.2.15**connection**

context or logical binding under specific conditions for the FDC protocol between a master object and a slave object for the Type 24 FAL

3.2.16

cyclic

repetitive in a regular manner

3.2.17

cyclic communication

transmission mode in which request PDUs and response PDUs are exchanged repetitively in the scheduled time slots synchronized with a transmission cycle for the lower layer protocol of the Type 24

Note 1 to entry: In the AL, the communication cycle arises from the transmission cycle in this mode.

3.2.18

cycle scale counter

counter to generate a communication cycle by means of scaling a primary cycle or a transmission cycle

3.2.19

device ID

part of "Device Information" to identify the device for a specific product type or model of the Type 24 fieldbus

3.2.20

device information

formatted and device-embedded information to characterize a device, which mainly consists of data for device model identification and device-profile specific parameters for the Type 24 fieldbus

3.2.21

device profile

collection of device-model-common information and functionality providing consistency between different device models among the same kind of devices

3.2.22

dual transfer

transfer mode for the FDC protocol of the Type 24 FAL, in which a sender sends a same PDU twice a transaction and a receiver uses them to detect and recover a communication error such as data-corruption or data-loss in cyclic communication mode

3.2.23

event driven communication

transmission mode for the lower layer protocol of the Type 24 fieldbus in which a transaction of command-response-exchanging arises as user's demands

Note 1 to entry: Both the transmission cycle and the communication cycle don't arise in this mode.

3.2.24

error

abnormal condition or malfunction for communication or any other activities

3.2.25

field device control

FDC service

time-critical communication service that handles a fixed length command data to control a field device and the corresponding feedback response data in a severe restriction on delay or jitter for the communication timing for the Type 24 FAL

3.2.26**field device protocol****FDC protocol**

time-critical communication protocol that handles a fixed length command data to control a field device and the corresponding feedback response data in a severe restriction on delay or jitter for the communication timing for the Type 24 FAL

3.2.27**master**

class or its instance object of FDC application service element (ASE) who plays a role of a command requester for the Type 24 FAL

3.2.28**message service****MSG service**

communication service that handles the variable length data and not required a severe restriction on response time

3.2.29**monitor**

class or its instance object of FDC ASE who plays a role of a watcher or subscriber of commands and response between other communication nodes for the Type 24 FAL

3.2.30**monitor slave**

variant of slave AP type who has both slave class and monitor class for FDC ASE of the Type 24 FAL

3.2.31**network clock**

synchronized and periodically running counter that each nodes in a same network have, which becomes an oscillation source of the transmission cycle

3.2.32**primary cycle**

period of repetitive activities synchronized with the transmission cycle before the connection establishing for the FDC protocol in Type 24 FAL

3.2.33**protocol machine**

state machine that realizes a protocol as the main function of the entity in each layer

3.2.34**requester**

class or its instance object of MSG ASE who plays a role of a command requester or sender for the Type 24 FAL

3.2.35**responder**

class or its instance object of MSG ASE who plays a role of a command responder or receiver for the Type 24 FAL

3.2.36**response**

PDU issued by a responder or a slave to inform a result or some status for the received command to a requester or a master

**3.2.37
service**

operation or process that an object performs upon request from another object

**3.2.38
single transfer**

normal transfer mode for the FDC protocol of the Type 24 FAL in which a sender sends a PDU once a transaction

**3.2.39
slave AP**

AP type that has slave facilities for the FDC service of the Type 24 FAL, or the device implementing that AP type

**3.2.40
slave**

class or its instance object of FDC ASE who plays a role of a responder for the Type 24 FAL

**3.2.41
sync command**

type of command APDU of the FDC service of the Type 24 FAL, which is issued at the synchronized timing with every communication cycle

Note 1 to entry: The definitions, which command is sync one or not, are dependent on an application. They can be provided as a registered set of commands and responses or device profiles, see 4.4 and Annex A.

**3.2.42
synchronous communication**

state or a way of communication for the FDC service of the Type 24 FAL, in which a command is issued at the synchronized timing with every communication cycle

Note 1 to entry: In this state, both sync commands and async ones can be issued.

Note 2 to entry: In this state, an out-of-synchronization error of APs shall be detected by measures of the watchdog counter.

**3.2.43
transmission cycle**

period of repetitive activities for the lower layers of the Type 24 fieldbus, which of all the slave devices are synchronized with that of a C1 master device by the lower layer protocol

**3.2.44
transmission mode**

state or a way of transmission for the lower layer protocol of the Type 24 fieldbus; cyclic mode, event driven mode

**3.2.45
virtual memory space**

large data block of APOs for the Type 24 FAL that can be read and written with pseudo-memory-addresses to provide consistency between different device models

Note 1 to entry: The virtual memory space includes the device Information and other vendor specific area, see Annex B.

**3.2.46
warning**

field device status to tell that the device has detected a slight or passing problem but is still working normally through the field device control (FDC) service of the Type 24 fieldbus

Note 1 to entry: Any warning statuses are latched and need to be operated to clear them.

Note 2 to entry: Warnings are classified into three groups, communication warnings, illegal-command-related ones, and application specific ones. But concrete definitions are dependent on an implementation of each field devices.

3.2.47

node

device that comply with the protocols described in this specification

3.2.48

output data

data to be sent from a master to a slave

3.2.49

input data

data to be sent from a slave to a master

3.2.50

node address

address to be specified as destination address or source address

3.3 Abbreviations and symbols

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

AE	Application Entity
AL	Application Layer
A-, AL-	Application Layer (as a prefix)
ALME	Application Layer Management Entity
AP	Application Process
APDU	Application Protocol Data Unit
API	Application Process Invocation
APO	Application Process Object
APC	Application Process Context (as prefix of a protocol for Type 24 fieldbus)
APC SM	Application Process Context State Machine (for Type 24 fieldbus)
AR	Application Relationship
AR ASE	Application Relationship Application Service Element
AREP	Application Relationship End Point
ARPM	Application Relationship Protocol Machine (for Type 24 fieldbus)
ARPM-FDCM	ARPM for Field Device Control service Master (for Type 24 fieldbus)
ARPM-FDCMN	ARPM for Field Device Control service Monitor (for Type 24 fieldbus)
ARPM-FDCS	ARPM for Filed Device Control service Slave (for Type 24 fieldbus)
ARPM-MSG	ARPM for Message service (for Type 24 fieldbus)
ASCII	American Standard Code for Information Interchange
ASDU	Application Service Data Unit
ASE	Application Service Element
ASN.1	Abstract Syntax Notation One
CMD	Command PDU for FDC service (for Type 24 fieldbus)
Cnf	confirm primitive
DL-	(as a prefix) Data Link-
DLL	Data Link Layer
DLM	Data Link-management

DLPDU	Data Link-Protocol Data Unit
DLSAP	Data Link Service Access Point
DLSDU	Data Link Service Data Unit
DMPM	Data Link Mapping Protocol Machine
E2PROM	Electrically erasable programmable read only memory
FAL	Fieldbus Application Layer
FCS	Frame check sequence
FDC-	Field Device Control (as prefix of a service or a protocol for Type 24 fieldbus)
FDC ASE	Field Device Control Application Service Element (for Type 24 fieldbus)
FDCPM	Field Device Control Protocol Machine (for Type 24 fieldbus)
FDCPM-M	Field Device Control Protocol Machine for Master (for Type 24 fieldbus)
FDCPM-MN	Field Device Control Protocol Machine for Monitor (for Type 24 fieldbus)
FDCPM-S	Field Device Control Protocol Machine for Slave (for Type 24 fieldbus)
FIFO	First In First Out
FSPM	FAL service protocol machine
FSM-	Fieldbus System Management (as prefix of a service for Type 24 fieldbus)
FSM ASE	Fieldbus System Management Application Service Element for Type 24 fieldbus
HMI	Human-machine Interface
I/O	Input/output
ID	Identifier
Ind	indication primitive
LME	Layer Management Entity
Lsb	least significant bit
MAC	Media Access Control
Msb	most significant bit
MSG	Message (as prefix of a service or a protocol for Type 24 fieldbus)
MSG ASE	Message Application Service Element for Type 24 fieldbus
MSGPM	Message Protocol Machine for Type 24 fieldbus
MSGPM-RQ	MSGPM for Requester for Type 24 fieldbus
MSGPM-RS	MSGPM for Responder for Type 24 fieldbus
OSI	Open Systems Interconnection
PM	Protocol machine
PDU	Protocol Data Unit
PhL	Ph-layer
QoS	Quality of Service
RAM	Random access memory
Req	request primitive
Rsp	response primitive
RSP	Response PDU for FDC service (for Type 24 fieldbus)
SAP	Service Access Point
SDN	Send Data with no Acknowledge
SDU	Service Data Unit

SM	State Machine
SMIB	System Management Information Base
UML	Unified Modelling Language

3.4 Conventions

3.4.1 General conventions

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 61158-5-24. 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. In this document, it is assumed that all attributes are accessible only from the owner's instance object itself directly or through services of the class.

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

3.4.2 PDU data type conventions

The data types of FAL PDUs are defined with the notations of ASN.1.

Character strings that begin with "_" (low line) are used as data type symbols of FAL PDUs or their fields. A same string as a PDU type without "_" means a PDU instance of the corresponding data type.

Example CMD-PDU means an instance of _CMD-PDU and a following statement is omitted.

```
CMD-PDU _CMD-PDU ::= { value }
```

3.4.3 State machine conventions

The protocol sequences are described by means of State Machines.

In statechart diagrams, states are represented as boxes, and state transitions are shown as arrows. Their conventions are given in ISO/IEC 19501 as Universal Modeling Language (UML).

Names of states and transitions of the state diagram correspond to the names in the textual listing of the state transitions.

The textual listing of the state transitions is structured as Table 1.

The first row contains the name of the transition by an index number.

The second row defines the source state or the current state before the transition.

The third row contains an event and actions. The event is followed by optional arguments in parentheses, "(" and ")", and guard conditions in brackets, "[" and "]", as the first line. And the actions with starting character "/" follow the event line.

The last row contains the target state or the next state after the transition.

If the event occurs and the conditions are fulfilled the transition fires, i.e. the actions are executed and the next state is entered.

Table 1 – State transition descriptions

T#	Source State	Event (arguments) [conditions] / Action	Target state

A meaning of each element in Table 1 is shown in Table 2, based on UML "transition" definition (ISO/IEC 19501).

Table 2 – Description of state machine elements

Description element	Meaning
Source state Target state	Names of the originating state and the target state of transition.
T#	Name or number of the transition.
Event	Name or description of the trigger event that fire the transition.
(arguments)	A parameter value, an expression, or a sequence of those divided by "," for the event. The preceding "(" and the succeeding ")" are not part of the argument list.
[conditions]	Boolean expression, which is true for the transition to be fired. The preceding "[" and the succeeding "]" are not part of the condition.
/ Action	List of assignments and service or function invocations. The action should be atomic. The preceding "/" is not part of the action.
NOTE "(arguments)" and "[condition]" can be omitted if not necessary.	

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

Table 3 – Conventions used in state machines

Convention	Meaning
<code>/*-- description --*/</code>	Describes and explains conditions and/or procedure by using normal sentence between <code>/*--</code> and <code>--*/</code> instead of using the pseudo code notation. Example: <code>/*-- The PM examines an occurrence of any alarms for the corresponding remote device.</code> <code>-- If an alarm is detected, the PM notifies it to the FSM ASE.</code> <code>-- If not, the PM transfers the MSGService-PDU to the DLL.</code> <code>--*/</code>
<code>=</code>	Value of an item on the left is replaced by value of an item on the right. If an item on the right is a parameter, it comes from the primitive shown as an input event
<code>Axx</code>	Parameter name if 'a' is a letter Example: <code>Identifier = reason;</code> means value of a 'reason' parameter is assigned to a parameter called 'Identifier'
<code>"xxx"</code>	Fixed visible string Example: <code>Identifier = "abc";</code> means value "abc" is assigned to a parameter named 'Identifier'
<code>Nnn</code>	If all elements are digits, the item represents a numerical constant shown in decimal representation.
<code>0xnn</code>	If all elements nn are digits, the item represents a numerical constant shown in hexadecimal representation.
<code>==</code>	Logical condition to indicate an item on the left is equal to an item on the right
<code><</code>	Logical condition to indicate an item on the left is less than the item on the right
<code>></code>	Logical condition to indicate an item on the left is greater than the item on the right
<code>!=</code>	Logical condition to indicate an item on the left is not equal to an item on the right
<code>&&</code>	Logical "AND"
<code> </code>	Logical "OR"
<code>!</code>	Logical "NOT"
<code>+</code> <code>-</code> <code>*</code> <code>/</code>	Arithmetic operators
<code>;</code>	Separator of expressions

Further notations as defined in C language (ISO/IEC 9899) can be used to describe conditions and actions.

4 Abstract syntax

4.1 Basic Data types

The following data types are used in the Type 24 FAL.

- Basic types as simple types of ASN.1:
 - INTEGER;
 - REAL;
 - BIT STRING;
 - OCTET STRING;
 - NULL.
- Specific basic types as subtypes from ASN.1:
 - Integer8 ::= INTEGER (–128..127);
 - Integer16 ::= INTEGER (–32 768 .. 32 767);
 - Integer32 ::= INTEGER (–2 147 483 648 .. 2 147 483 647);
 - Integer64 ::= INTEGER ((–1) × '8000 0000 0000 0000'H..'7FFF FFFF FFFF FFFF'H);
 - Unsigned8 ::= INTEGER (0..'FF'H);
 - Unsigned16 ::= INTEGER (0..'FFFF'H);
 - Unsigned32 ::= INTEGER (0..'FFFF FFFF'H);
 - Unsigned64 ::= INTEGER (0..'FFFF FFFF FFFF FFFF'H);
 - *Float*₃₂ ::= 0 | negative infinity | positive infinity
 | REAL (WITH COMPONENTS { mantissa ((–1) × c_{32}), base (2), exponent (q_{32}) })
 | REAL (WITH COMPONENTS { mantissa (c_{32}), base (2), exponent (q_{32}) })

where

$$c_{32} ::= \text{REAL} \left(1.. \left(\sum_{i=0}^{23} \frac{1}{2^i} \right) \right) = \text{REAL} (1..(2-2^{-23})), \text{ and}$$

$$q_{32} ::= \text{INTEGER} (-127..127).$$

NOTE 1 The range of the real value is covered from negative infinity to positive infinity. But the representable precision conforms to binary32 or the single precision of ISO/IEC/IEEE 60559, see 5.2.2.

- *Float*₆₄ ::= 0 | negative infinity | positive infinity
 | REAL (WITH COMPONENTS { mantissa ((–1) × c_{64}), base (2), exponent (q_{64}) })
 | REAL (WITH COMPONENTS { mantissa (c_{64}), base (2), exponent (q_{64}) })

where

$$c_{64} ::= \text{REAL} \left(1.. \left(\sum_{i=0}^{52} \frac{1}{2^i} \right) \right) = \text{REAL} (1.. (2-2^{-52})), \text{ and}$$

$$q_{64} ::= \text{INTEGER} (-1\ 023..1\ 023).$$

NOTE 2 The range of the real value is covered from negative infinity to positive infinity. But the representable precision conforms to binary64 or the double precision of ISO/IEC/IEEE 60559, see 5.2.2.

- IA5String
- Structure types with component types of ASN.1:

Structure type is defined by a combination of SEQUENCE type, CHOICE type of ASN.1, basic types and subtype elements defined in ASN.1. The FAL PDU described in 4.2 and 4.3 that follows this subclause correspond this Structure type.

 - SEQUENCE
 - CHOICE

- Array types with a component type of ASN.1:

Array types are a list of any data type defined with SEQUENCE OF type of ASN.1. Typical array types are shown with SEQUENCE OF type, below:

- SEQUENCE OF;
- BitArray ::= SEQUENCE OF (BIT STRING SIZE(8));
- Array8 ::= SEQUENCE OF Integer8;
- Array16 ::= SEQUENCE OF Integer16;
- Array32 ::= SEQUENCE OF Integer32;
- Array64 ::= SEQUENCE OF Integer64.

4.2 FAL PDU types

4.2.1 Top of APDU types: _APDU

The APDU type: _APDU shall consist of two groups; _FDCServicePDU and _MSGServicePDU.

And _FDCServicePDU shall consist of a pair of PDUs; _CMD-PDU and _RSP-PDU, used for FDC commands and its responses respectively.

Furthermore, _MSGServicePDU shall consist of a pair of PDUs; _MSGREQ-PDU and _MSGRSP-PDU, used for MSG requests and its responses respectively.

```

_APDU                ::= _FDCServicePDU
                        | _MSGServicePDU

_FDCServicePDU       ::= _CMD-PDU                -- FDC command PDU type
                        | _RSP-PDU                -- FDC response PDU type

_MSGServicePDU        ::= _MSGREQ-PDU            -- MSG request PDU type
                        | _MSGRSP-PDU            -- MSG response PDU type

```

Figure 1 shows the overview of APDU type definitions by the tree structure chart.

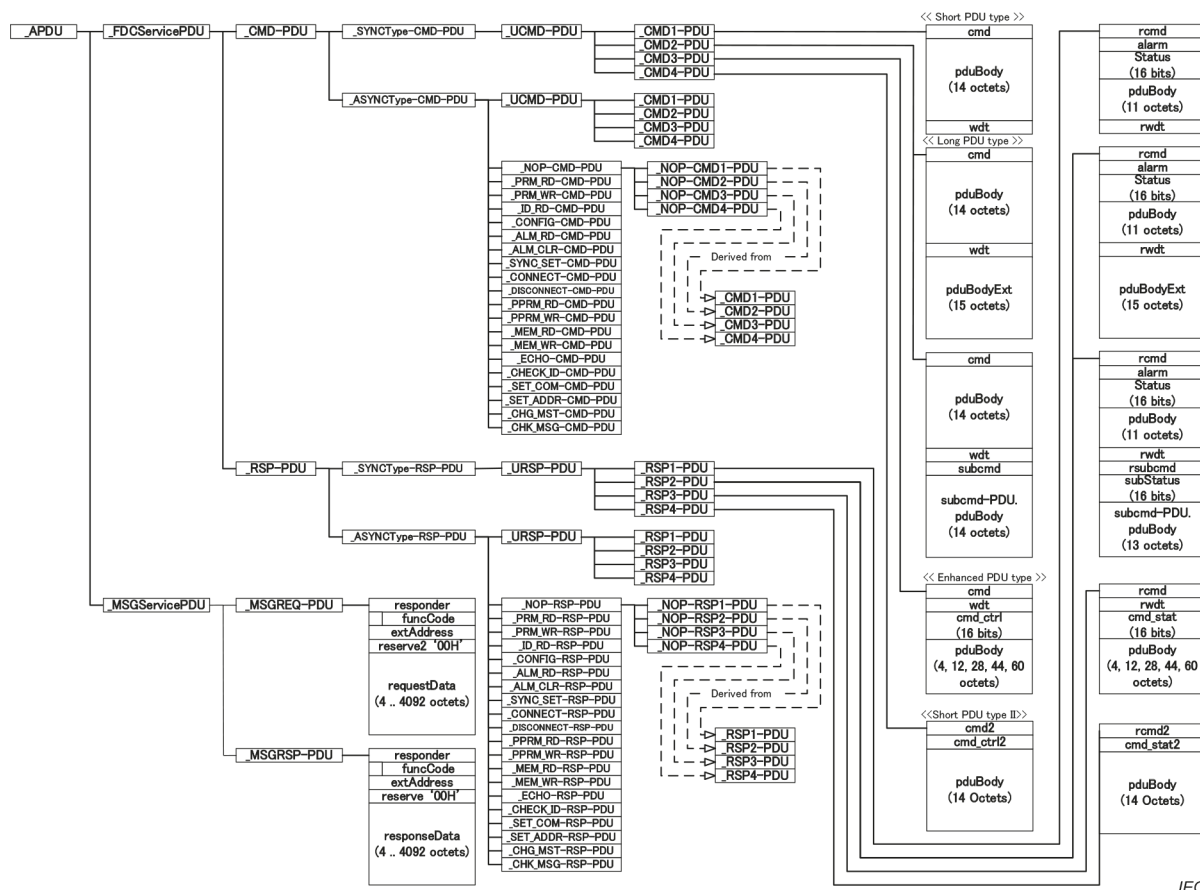


Figure 1 – Tree structure of APDU types

4.2.2 PDUs for field device control service

4.2.2.1 Overview

FDCServicePDU shall be edited with a SDU in the FDC service primitive and be sent via AR. And a FDC ASE receives the PDU from the peer ASE via AR.

The FDC command PDU type: `_CMD_PDU` shall be classified into sync type command (sync command): `_SYNCType-CMD-PDU` and async type command (async command): `_ASYNCType-CMD-PDU`. Both types are the identical PDU formats: `_UCMD-PDU`. Therefore, the FDC PM cannot distinguish their type's PDUs. The user of FDC ASE should know which commands belong to the sync commands or not, and can distinguish the types by using `cmd` field as a key code in the PDU header part.

The async command PDU: `_ASYNCType-CMD_PDU` shall include `_UCMD-PDU` and other twenty PDU types as common commands, which are also variants of `_UCMD-PDU`.

Derived from the `_UCMD-PDU` and the `_URSP-PDU` explained later, a set of commands and responses can be defined for a specific application, such as motor-drive control, servo control, etc., by FAL users, and the set can be registered as a device profile (see 4.4 and Annex A). In addition, a FAL user can select any device profiles supported by a target field device, when a connection is established or when it is transited to the cyclic communication mode for connection-less type by FDC-Connect service (see 8.2.4.3 and IEC 61158-5-24, 6.4.1.2.3.4).

_CMD-PDU	::= _SYNCType-CMD-PDU _ASYNCType-CMD-PDU
_SYNCType-CMD-PDU	::= _UCMD-PDU
_ASYNCType-CMD-PDU	::= _UCMD-PDU _NOP-CMD-PDU _PRM_RD-CMD-PDU _PRM_WR-CMD-PDU _ID_RD-CMD-PDU _CONFIG-CMD-PDU _ALM_RD-CMD-PDU _ALM_CLR-CMD-PDU _SYNC_SET-CMD-PDU _CONNECT-CMD-PDU _DISCONNECT-CMD-PDU _PPRM_RD-CMD-PDU _PPRM_WR-CMD-PDU _MEM_RD-CMD-PDU _MEM_WR-CMD-PDU _ECHO-CMD-PDU _CHECK_ID-CMD-PDU _SET_COM-CMD-PDU _SET_ADDR-CMD-PDU _CHG_MST-CMD-PDU _CHK_MSG-CMD-PDU
_UCMD-PDU	::= _CMD1-PDU _CMD2-PDU _CMD3-PDU _CMD4-PDU
_NOP-CMD-PDU	::= _NOP-CMD1-PDU _NOP-CMD2-PDU _NOP-CMD3-PDU _NOP-CMD4-PDU
_PRM_RD-CMD-PDU	::= _PRM_RD-CMD1-PDU _PRM_RD-CMD2-PDU _PRM_RD-CMD3-PDU
_PRM_WR-CMD-PDU	::= _PRM_WR-CMD1-PDU _PRM_WR-CMD2-PDU _PRM_WR-CMD3-PDU
_ID_RD-CMD-PDU	::= _ID_RD-CMD1-PDU _ID_RD-CMD2-PDU _ID_RD-CMD3-PDU
_CONFIG-CMD-PDU	::= _CONFIG-CMD1-PDU _CONFIG-CMD2-PDU _CONFIG-CMD3-PDU
_ALM_RD-CMD-PDU	::= _ALM_RD-CMD1-PDU _ALM_RD-CMD2-PDU _ALM_RD-CMD3-PDU
_ALM_CLR-CMD-PDU	::= _ALM_CLR-CMD1-PDU _ALM_CLR-CMD2-PDU _ALM_CLR-CMD3-PDU
_SYNC_SET-CMD-PDU	::= _SYNC_SET-CMD1-PDU _SYNC_SET-CMD2-PDU _SYNC_SET-CMD3-PDU
_CONNECT-CMD-PDU	::= _CONNECT-CMD1-PDU _CONNECT-CMD2-PDU _CONNECT-CMD3-PDU
_DISCONNECT-CMD-PDU	::= _DISCONNECT-CMD1-PDU _DISCONNECT-CMD2-PDU _DISCONNECT-CMD3-PDU
_PPRM_RD-CMD-PDU	::= _PPRM_RD-CMD1-PDU _PPRM_RD-CMD2-PDU _PPRM_RD-CMD3-PDU
_PPRM_WR-CMD-PDU	::= _PPRM_WR-CMD1-PDU _PPRM_WR-CMD2-PDU _PPRM_WR-CMD3-PDU

```

_MEM_RD-CMD-PDU      ::= _MEM_RD-CMD3-PDU
_MEM_WR-CMD-PDU      ::= _MEM_WR-CMD3-PDU
_ECHO-CMD-PDU        ::= _ECHO_ID-CMD4-PDU
_CHK_ID-CMD-PDU      ::= _CHK_ID-CMD4-PDU
_SET_COM-CMD-PDU     ::= _SET_COM-CMD4-PDU
_SET_ADDR_ID-CMD-PDU ::= _SET_ADDR-CMD4-PDU
_CHG_MST-CMD-PDU     ::= _CHG_MST-CMD4-PDU

```

The FDC response PDU type: `_RSP_PDU` shall be also classified into sync type response (sync response): `_SYNCType-RSP-PDU` and async type response (async response): `_ASYNCType-RSP-PDU`. Both types shall be derived from the identical PDU formats: `_URSP-PDU`. Therefore, the FDC PM cannot distinguish their type's PDUs. The user of FDC ASE should know which responses belong to the sync responses or not, and can distinguish the types by using `rcmd` field as a key code in the PDU header part. Moreover, this prefix code of a response PDU corresponds to one of a command PDU, since a response shall be exchanged for a command.

The async type response PDU: `_ASYNCType-RSP_PDU` shall include `_URSP-PDU` and other twenty PDU types as common responses to any application, which are also variants of `_URSP-PDU`.

```

_RSP-PDU              ::= _SYNCType-RSP-PDU
                        | _ASYNCType-RSP-PDU
_SYNCType-RSP-PDU     ::= _URSP-PDU
_ASYNCType-RSP-PDU    ::= _URSP-PDU
                        | _NOP-RSP-PDU
                        | _PRM_RD-RSP-PDU
                        | _PRM_WR-RSP-PDU
                        | _ID_RD-RSP-PDU
                        | _CONFIG-RSP-PDU
                        | _ALM_RD-RSP-PDU
                        | _ALM_CLR-RSP-PDU
                        | _SYNC_SET-RSP-PDU
                        | _CONNECT-RSP-PDU
                        | _DISCONNECT-RSP-PDU
                        | _PPRM_RD-RSP-PDU
                        | _PPRM_WR-RSP-PDU
                        | _MEM_RD-RSP-PDU
                        | _MEM_WR-RSP-PDU
                        | _ECHO-RSP-PDU
                        | _CHECK_ID- RSP-PDU
                        | _SET_COM- RSP-PDU
                        | _SET_ADDR- RSP-PDU
                        | _CHG_MST- RSP-PDU
                        | _CHK_MSG- RSP -PDU
_URSP-PDU              ::= _RSP1-PDU | _RSP2-PDU | _RSP3-PDU | _RSP4-PDU
_NOP-RSP-PDU           ::= _NOP-RSP1-PDU
                        | _NOP-RSP2-PDU
                        | _NOP-RSP3-PDU
                        | _NOP-RSP4-PDU

```

<code>_PRM_RD-RSP-PDU</code>	::= <code>_PRM_RD-RSP1-PDU</code> <code>_PRM_RD-RSP2-PDU</code> <code>_PRM_RD-RSP3-PDU</code>
<code>_PRM_WR-RSP-PDU</code>	::= <code>_PRM_WR-RSP1-PDU</code> <code>_PRM_WR-RSP2-PDU</code> <code>_PRM_WR-RSP3-PDU</code>
<code>_ID_RD-RSP-PDU</code>	::= <code>_ID_RD-RSP1-PDU</code> <code>_ID_RD-RSP2-PDU</code> <code>_ID_RD-RSP3-PDU</code>
<code>_CONFIG-RSP-PDU</code>	::= <code>_CONFIG-RSP1-PDU</code> <code>_CONFIG-RSP2-PDU</code> <code>_CONFIG-RSP3-PDU</code>
<code>_ALM_RD-RSP-PDU</code>	::= <code>_ALM_RD-RSP1-PDU</code> <code>_ALM_RD-RSP2-PDU</code> <code>_ALM_RD-RSP3-PDU</code>
<code>_ALM_CLR-RSP-PDU</code>	::= <code>_ALM_CLR-RSP1-PDU</code> <code>_ALM_CLR-RSP2-PDU</code> <code>_ALM_CLR-RSP3-PDU</code>
<code>_SYNC_SET-RSP-PDU</code>	::= <code>_SYNC_SET-RSP1-PDU</code> <code>_SYNC_SET-RSP2-PDU</code> <code>_SYNC_SET-RSP3-PDU</code>
<code>_CONNECT-RSP-PDU</code>	::= <code>_CONNECT-RSP1-PDU</code> <code>_CONNECT-RSP2-PDU</code> <code>_CONNECT-RSP3-PDU</code>
<code>_DISCONNECT-RSP-PDU</code>	::= <code>_DISCONNECT-RSP1-PDU</code> <code>_DISCONNECT-RSP2-PDU</code> <code>_DISCONNECT-RSP3-PDU</code>
<code>_PPRM_RD-RSP-PDU</code>	::= <code>_PPRM_RD-RSP1-PDU</code> <code>_PPRM_RD-RSP2-PDU</code> <code>_PPRM_RD-RSP3-PDU</code>
<code>_PPRM_WR-RSP-PDU</code>	::= <code>_PPRM_WR-RSP1-PDU</code> <code>_PPRM_WR-RSP2-PDU</code> <code>_PPRM_WR-RSP3-PDU</code>
<code>_MEM_RD-RSP-PDU</code>	::= <code>_MEM_RD-RSP3-PDU</code>
<code>_MEM_WR-RSP-PDU</code>	::= <code>_MEM_WR-RSP3-PDU</code>
<code>_ECHO-RSP-PDU</code>	::= <code>_ECHO-RSP4-PDU</code>
<code>_CHK_ID-RSP-PDU</code>	::= <code>_CHK_ID-RSP4-PDU</code>
<code>_SET_COM- RSP-PDU</code>	::= <code>_SET_COM-RSP4-PDU</code>
<code>_SET_ADDR- RSP-PDU</code>	::= <code>_SET_ADDR-RSP4-PDU</code>
<code>_CHG_MST- RSP-PDU</code>	::= <code>_CHG_MST-RSP4-PDU</code>
<code>_MSG_REQ- RSP-PDU</code>	::= <code>_MSG_REQ-RSP4-PDU</code>

4.2.2.2 Common component of PDU

4.2.2.2.1 FDC command PDU

`_UCMD-PDU` has four types, short type: `_CMD1-PDU`, long type: `_CMD2-PDU`, and enhanced type: `_CMD3-PDU`, and short type II: `_CMD4-PDU`. Both `_CMD1-PDU` and `_CMD2-PDU` shall have same header fields, but `_CMD3-PDU` shall have another enhanced one.

The pduBody length of `_CMD1-PDU` shall be 14 octets when it encoded, and so as for `_CMD2-PDU`. This part is called main-command, and `_CMD2-PDU` shall have another extended pduBodyExt field too, and can piggyback an additional command PDU called sub-command: `_SUBCMD-PDU` on that field.

The pduBody length of `_CMD3-PDU` shall be selected from 4, 12, 28, 44, and 60 octets. It can be selected for each station; however, it shall not be changed after selecting.

The pduBody length of `_CMD4-PDU` shall be within the range of 1 octet to 14 octets. It can be changed for each `_CMD4-PDU`.

```

_CMD1-PDU      ::= SEQUENCE{                                     -- short PDU type
    cmd         _CMDCode,
    pduBody     OCTET STRING SIZE (14) ,
    wdt         _WDT }

_CMD2-PDU      ::= SEQUENCE{                                     -- long PDU type
    maincmd-PDU COMPONENTS OF _CMD1-PDU,                       -- main-command
    CHOICE { subcmd-PDU _SUBCMD-PDU,                           -- sub-command
        pduBodyExt OCTET STRING SIZE (15) } }

_SUBCMD-PDU    ::= _USUBCMD-PDU SEQUENCE{                       -- sub command
    subcmd      _SubCMD,                                       PDU type
    pduBody     OCTET STRING SIZE
(14)}
    | _NOP-SUBCMD-PDU
    | _PRM_RD-SUBCMD-PDU
    | _PRM_WR-SUBCMD-PDU
    | _ALM_RD-SUBCMD-PDU
    | _PPRM_RD-SUBCMD-PDU
    | _PPRM_WR-SUBCMD-PDU

_CMD3-PDU      ::= SEQUENCE{                                     -- enhanced PDU
    cmd         _CMDCode,                                     type
    wdt         _WDT,
    cmd_ctrl    _CMD_CTRL,
    pduBody     OCTET STRING SIZE (4|12|28|44|60) }

_CMD4-PDU      ::= SEQUENCE{                                     -- short PDU type
    cmd2        _CMDCode2,                                    ||
    cmd_ctrl2    _CMD_CTRL2,
    pduBody     OCTET STRING SIZE (1..14) }

```

- cmd field

cmd

data field which contains the code to identify contents of each command

This field shall be coded as _CMDCode, subtype of Unsigned8, with the following value range. The predefined common commands are shown as _PrimaryCMD below.

- '00'H to '1F'H: used or reserved for common commands;
- '20'H to 'BF'H: reserved for application specific commands;
- 'C0'H to 'FF'H: reserved for vendors.

```

_CMDCode          ::= Unsigned8
_PrimaryCMD       ::= _CMDCode {
    nop ('00'H),
    prm_rd ('01'H), prm_wr ('02'H),
    id_rd ('03'H),
    config ('04'H),
    alm_rd ('05'H), alm_clr ('06'H),
    sync_set ('0D'H),
    connect ('0E'H),
    disconnect ('0F'H),
    pprm_rd ('1B'H), pprm_wr ('1C'H),
    mem_rd ('1D'H), mem_wr ('1E'H) }

```

- wdt field

wdt

watchdog counter field

This field shall be used for the slave to detect out-of synchronous activity or the WDT error of the corresponding master. The master shall count the wdt field of sending PDUs every communication cycle in the case of the synchronous communication state. In addition, the slave shall examine the field value of receiving PDUs every communication cycle to detect the wdt error.

The Data type is `_WDT`. It shall have two sub fields, mn and sn.

```

_WDT              ::= SEQUENCE { mn BIT STRING SIZE (4),
                                   sn BIT STRING SIZE (4)}

```

mn (Master count value)

The value range shall be from 0 to 15.

The mn field in a next CMD-PDU to be sent shall be counted up by one. An FDC master shall do that every communication cycle. On the other hand, when an FDC slave receives the PDU, it shall examine the counter every cycle. And, if the counter isn't matching the last mn value plus one, it shall alert the watchdog counter error.

sn (Slave count value)

The value range shall be from 0 to 15.

The sn field in a next CMD-PDU to be sent shall be set the same value of the rsn field in the last received RSP-PDU. An FDC master shall copy it from the last RSP-PDU every communication cycle.

- subcmd field

subcmd

data field which contains the code to identify contents of each sub-command

The Data type is `_SubCMD`. The code definition for the sub-commands shall be the same as main cmd field in principle, but some codes are restricted to use, if they have any difficulties in processing parallel commands. The predefined common sub-commands are shown as `_PrimarySubCMD` below.

```

_SubCMD ::= Unsigned8
_PrimarySubCMD ::= _SubCMD {
    nop ('00'H),
    prm_rd ('01'H), prm_wr ('02'H),
    alm_rd ('05'H),
    pprm_rd ('1B'H), pprm_wr ('1C'H) }

```

- cmd_ctrl field

cmd_ctrl

set of control bits that is independent of any command transactions, from FDC master to FDC slave

The Data type is `_CMD_CTRL`. It shall have two meaningful sub-fields, `alm_clr` and `cmd_id`.

```

_CMD_CTRL ::= SEQUENCE { reserve1 BIT STRING SIZE (3),
    alm_clr BIT STRING SIZE (1),
    reserve2 BIT STRING SIZE (2),
    cmd_id BIT STRING SIZE (2),
    reserve3 BIT STRING SIZE (8) }

```

alm_clr

This field shall be used to command to clear alarm/warning status of the slave device.

value 0: to execute nothing

value changes to 1: to execute the status clear

cmd_id

When the master device continuously issues identical contents of a CMD-PDU to the slave, it can be used to make the slave device recognizing the command as another command.

The value range shall be from 0 to 3.

The FDC master can use the `cmd_id` to have a FDC slave acknowledge that the command is a new command when the master sends the same command repeatedly to the slave.

Since the slave shall respond the echo of the `cmd_id` of every command through `RSP-PDU.cmd_stat.rcmd_id` field explained later, the master can also judge the command for which the slave station sent the response.

Use of the `cmd_id` is not mandatory for the master.

While `RSP-PDU.cmd_stat.cmdRdy = 0` in response PDUs, the slave shall disregard commands that have a different `cmd_id` and continues to execute the command which has been already accepted.

- cmd2 field

cmd2

data field which contains the code to identify contents of each command

This field shall be coded as `_CMDCode2`, subtype of `Unsigned8`, with the following value range. The predefined common commands are shown as `_PrimaryCMD2` below.

```

_CMDCode2          ::= Unsigned8
_PrimaryCMD2       ::= _CMDCode2 {
    nop ('00'H),
    echo ('FF'H),
    check_id ('E2'H),
    set_com ('FA'H),
    set_addr ('F3'H),
    chg_mst ('0E'H),
    chk_msg_req ('0F'H) }

```

- cmd_ctrl2 field

cmd_ctrl2

set of control data that is independent of any command transactions, from FDC master to FDC slave. The Data type is _CMD_CTRL2 and cmd_ctrl2 depends on cmd2.

```

_CMD_CTRL2         ::= Unsigned8

```

4.2.2.2.2 FDC response PDU

_URSP-PDU has four types, short type: _RSP1-PDU, long type: _RSP2-PDU, and enhanced type: _RSP3-PDU, and short type II: _RSP4-PDU. Both _RSP1-PDU and _RSP2-PDU shall have common header fields, but _RSP3-PDU shall have another enhanced one.

The pduBody length of _RSP-PDU shall be 11 octets, and so as for _RSP2-PDU. This is called a main response or a response of the main command, and _RSP2-PDU shall have another extended pduBodyExt field too, and can piggyback an additional command PDU called sub-response or a response of a sub command: _SUBRS-PDU on that field.

The pduBody length of _RSP3-PDU shall be able to be selected from 4, 12, 28, 44, and 60 octets. It can be selected for each station; however, it shall not be changed after selecting.

The pduBody length of _RSP4-PDU shall be within the range of 1 octet to 14 octets. It can be changed for each _RSP4-PDU.

```

_RSP1-PDU          ::= SEQUENCE{                                     -- short PDU type
    rcmd            _CMDCode,
    alarm           _ALARM,
    status          _STATUS,
    pduBody         OCTET STRING SIZE (11),
    rwdt            _RWDT}

_RSP2-PDU          ::= SEQUENCE{                                     -- long PDU type
    rsp1-PDU        COMPONENTS OF _RSP1-PDU,                       -- main response
    CHOICE { subrsp-PDU _SUBRS-PDU,                                -- sub response
        pduBodyExt  OCTET STRING SIZE (15) } }

```

```

_SUBRSP-PDU ::= _USUBRSP-PDU SEQUENCE{
                                rsubcmd  _SubCMD,
                                subStatus _SUBSTATUS,
                                pduBody   OCTET STRING SIZE
(13)}
    | _NOP-SUBRSP-PDU
    | _PRM_RD-SUBRSP-PDU
    | _PRM_WR-SUBRSP-PDU
    | _ALM_RD-SUBRSP-PDU
    | _PPRM_RD-SUBRSP-PDU
    | _PPRM_WR-SUBRSP-PDU

_RSP3-PDU ::= SEQUENCE{
            rcmd      _CMDCode,
            rwdt      _RWDt,
            cmd_stat  _CMD_STAT,
            pduBody   OCTET STRING SIZE (4|12|28|44|60) }

_RSP4-PDU ::= SEQUENCE{
            rcmd2      _CMDCode2,
            cmd_stat2  _CMD_STAT2,
            pduBody   OCTET STRING SIZE (1..14) }

```

-- sub response PDU type

-- enhanced PDU type

-- short PDU type

- rcmd field

rcmd

data field which contains the same code as a received and processing command

So, this field shall be coded as _CMDCode, see 4.2.2.2.1.

- alarm field

alarm

This field shall contain the alarm code.

The Data type is _ALARM. The value range shall be from 0 to 255.

```
_ALARM ::= Unsigned8
```

- status field

status

This field shall indicate the status of the slave device.

The Data type is _STATUS. It shall have three meaningful sub-fields as follows.

```

_STATUS ::= SEQUENCE { alarm  BIT STRING SIZE (1),
                        warning BIT STRING SIZE (1),
                        cmdRdy BIT STRING SIZE (1),
                        reserve BIT STRING SIZE (13) }

```

alarm

This bit field shall indicate the alarm status in the slave device.

- 0: No alarm,
- 1: any alarms occur.

warning

This bit field shall indicate the warning status in the slave device.

- 0: No warning,
- 1: any warnings occur.

cmdRdy

This bit field shall indicate the command progress status of the FDC slave.

The slave device shall keep the bit cmdRdy 0, while it is processing the command. The slave device shall change the bit from 0 to 1, when it has completed the processing.

To notice the completion of the specific command, the FDC master shall not only examine the bit, but also collate some other RSP-PDU fields with the corresponding CMD-PDU fields to discriminate from the preceding transaction.

When the retention time of cmdRdy = 0 is expired, the master shall raise a command time out error. The timer value depends on each product specification for slave devices.

A change of this bit status shall be independent of the alarm or warning status.

- 0: busy with command execution in progress
- 1: ready for new command

- rwdt field

rwdt

watchdog counter field

This field shall be used for the master to detect out-of synchronous activity or the WDT error of the corresponding slave. The slave shall count and update the rwdt field of sending PDUs every communication cycle in the case of the synchronous communication state. In addition, the master shall examine the field value of receiving PDUs every communication cycle to detect the rwdt error.

The Data type is `_RWDT`. It shall have two sub fields, `rmn` and `rsn`.

```
_RWDT ::= SEQUENCE {
    rmn BIT STRING SIZE (4),
    rsn BIT STRING SIZE (4)}
```

rmn (Master count value)

The value range shall be from 0 to 15.

The rmn field in a next RSP-PDU to be sent shall be set the same value of the mn field in the last received CMD-PDU. An FDC slave shall copy it from the CMD-PDU every communication cycle.

rsn (Slave count value)

The value range shall be from 0 to 15.

The rsn field in a next RSP-PDU to be sent shall be counted up by one. An FDC slave shall do that every communication cycle. On the other hand, when an FDC master receives the PDU, it shall examine the counter every cycle. And, if the counter isn't matching the last rsn value plus one, it shall alert the watchdog counter error.

- rsubcmd

rsubcmd

data field which contains the same code as the received and processing sub-command

This field shall be coded as _SubCMD, see 4.2.2.2.1.

- subStatus

subStatus

This field shall indicate the status of the slave station in the sub response PDU part.

The Data type is _SubSTATUS. This field shall be coded same as status field.

_SUBSTATUS ::= _STATUS

- cmd_stat field

cmd_stat

This field shall indicate the status of the slave device.

The Data type is _CMD_STAT. It shall have seven sub fields as follows.

_CMD_STAT ::= SEQUENCE { d_alm BIT STRING SIZE (1),
d_war BIT STRING SIZE (1),
cmdRdy BIT STRING SIZE (1),
alm_clr_cmp BIT STRING SIZE (1),
reserve BIT STRING SIZE (2),
rcmd_id BIT STRING SIZE (2),
cmd_alm BIT STRING SIZE (4),
comm_alm BIT STRING SIZE (4) }

d_alm

This bit field shall indicate occurrence status of device-specific alarm in the slave device.

0: the slave is not in device specific alarm status;

1: the slave is in device specific alarm status except comm_alm or cmd_alm.

Specification of device alarm status depends on specified product implementation, but a factor of the alarms should be classified separately from a communication problem and a command issue.

After the slave has been recovered to the alarm status, through the execution of the command ALM_CLR-CMD-PDU or cmd_ctrl.alm_clr bit in any CMD-PDU, this bit shall be cleared to 0.(d_alm = 0)

d_war

This bit field shall indicate occurrence status of device-specific warning in the slave station.

0: the slave is not in device specific alarm status;

1: the slave is in device specific alarm status except comm_alm or cmd_alm.

Specification of device alarm status depends on specified product implementation, but a factor of the alarms should be classified separately from a communication problem and a command issue.

After the slave has been recovered to the warning status, through the execution of the command ALM_CLR-CMD-PDU or cmd_ctrl.alm_clr bit in any CMD-PDU, this bit shall be cleared to 0.(d_war = 0)

cmdRdy

This bit field shall indicate the command progress status of the FDC slave.

Meaning of this bit is the same as RSP1_PDU.status.cmdRdy.

0: busy with command execution in progress;

1: ready for new command.

alm_clr_cmp

This bit field shall indicate the execution status of alarm clear process.

0: Alarm/warning clear not completed;

1: Alarm/warning clear completed.

rcmd_id

This field shall be a key code to indicate which command the response PDU corresponds to, by echoing back cmd_id of the corresponding command PDU.

The value range is from 0 to 3.

cmd_alm

This field shall notify an alarm code for command abnormality.

Each code shall mean the following.

- '00'H: Normal;
- '01'H: Warning on out-of-range data;
- '02'H to '07'H: reserved for warning codes;
- '08'H: Alarm on out-of-support;
- '09'H: Alarm on out-of-range data;
- '0A'H: Alarm on abnormal command execution condition;
- '0B'H: Alarm on abnormal subcommand combination;
- '0C'H: Alarm on abnormal phase;
- '0D'H to '0F'H: reserved for alarm codes.

comm_alm

This field shall notify an alarm code for communication error status.

Each code shall mean the following.

- '00'H: Normal;
- '01'H: Warning on abnormal FCS;
- '02'H: Warning on abnormal reception;
- '03'H to '07'H: reserved for warning;
- '08'H: Alarm on abnormal FCS;
- '09'H: Alarm on abnormal reception;
- '0A'H to '0F'H: reserved for alarm.

- rcmd2 field

rcmd2

data field which contains the same code as a received and processing command.

So, this field shall be coded as _CMDCode2, see 4.2.2.2.1.

- cmd_stat2 field

cmd_stat2

This field shall indicate the status of the slave device.

The Data type is _CMD_STAT2. It shall have seven sub fields as follows.

```
_CMD_STAT2 ::= SEQUENCE { busy BIT STRING SIZE (1),
                                com_err BIT STRING SIZE (1),
                                reserve BIT STRING SIZE (4),
                                wng_flg BIT STRING SIZE (1),
                                alm_flg BIT STRING SIZE (1) }
```

busy

This bit field shall indicate the command progress status of the FDC slave.

Meaning of this bit is the same as RSP1_PDU.status.cmdRdy.

0: busy with command execution in progress;

1: ready for new command.

comm_err

This bit field shall indicate the occurrence of command processing error or receiving of the unsupported command in the slave station.

wng_flg

This bit field shall indicate occurrence status of device-specific warning in the slave station.

0: the slave is not in device specific warning status;

1: the slave is in device specific warning status.

alm_flg

This bit field shall indicate occurrence status of device-specific alarm in the slave station.

0: the slave is not in device specific alarm status;

1: the slave is in device specific alarm status.

4.2.3 PDUs for message service

```

_MSGREQ-PDU ::= SEQUENCE { responder Unsigned8,
                                funcCode BIT STRING SIZE (7),
                                reserve1 BIT STRING SIZE (1),
                                extAddress Unsigned8,
                                reserve2 OCTET STRING ('00'H),
                                requestData OCTET STRING SIZE (4..4092)}
| SEQUENCE { requestUData OCTET STRING SIZE (8..4096)}
_MSGRSP-PDU ::= SEQUENCE { responder Unsigned8,
                                funcCode BIT STRING SIZE (7),
                                errorFlag BIT STRING SIZE (1),
                                extAddress Unsigned8,
                                reserve OCTET STRING ('00'H),
                                responseData OCTET STRING SIZE (4..4092)}
| SEQUENCE { responseUData OCTET STRING SIZE (8..4096)}

```

- responder field

responder

This field shall contain the destination node address or responder of the message.

This field shall be coded as data type Unsigned8 with the following value range.

- '00'H: reserved;
- '01'H: C1 master device;
- '02'H: C2 master device;
- '03'H to 'EF'h: Slave device;
- 'F0'H to 'FF'H: reserved.

- funcCode field

funcCode

This field shall contain the code indicating the function of the message.

The code range shall be from '00'H to '7F'H.

- errorFlag field

errorFlag

This field shall indicate the response state of the message. It shall be set to 0 on a normal response, and to 1 on an abnormal response.

- 0: a normal response;
- 1: an error response.

- extAddress field

extAddress

This field shall contain the sub address representing a sub device when the destination node is an integrated device with two or more sub devices.

This field shall be coded as data type Unsigned8 with the value range '00'H to 'FE'H. The code 'FF'H is reserved.

- requestData field and responseData field

requestData and responseData

These fields shall contain the SDUs that is defined by user for each function code. An example of a user message command set for function code: '42'H is shown in Annex C.

4.3 Detailed definitions of _FDCService-PDUs

4.3.1 Short PDU type

4.3.1.1 NOP command and response

NOP symbolizes a "no operation" command. Receiving this command, the slave shall do nothing but respond with a NOP-RSP1-PDU showing the latest alarm and status.

```
_NOP-CMD1-PDU      ::= _CMD1-PDU ( WITH COMPONENTS { ..., cmd (nop) } )
_NOP-RSP1-PDU      ::= _RSP1-PDU ( WITH COMPONENTS { ..., rcmd (nop) } )
```

4.3.1.2 PRM_RD command and response

PRM_RD symbolizes a "parameter-read" command. Receiving this command, the slave shall read a specified parameter value into PRM_RD-RSP1-PDU.parameter field and shall respond with it

```
_PRM_RD-CMD1-PDU    ::= _CMD1-PDU ( WITH COMPONENTS
                                { ..., cmd (prm_rd),
                                pduBody (cmdBody _PRM_RD-CMD1Body) } )
_PRM_RD-CMD1Body    ::= SEQUENCE{ reserve1  OCTET STRING SIZE (3),
                                pNo  Unsigned16,
                                pSize Unsigned8,
                                reserve2  OCTET STRING SIZE (8) }
_PRM_RD-RSP1-PDU    ::= _RSP1-PDU ( WITH COMPONENTS
                                { ..., rcmd (prm_rd),
                                pduBody (rspBody _PRM_RD-RSP1Body) } )
```

```

_PRm_RD-RSP1Body ::= SEQUENCE { pNo  Unsigned16,
                                   pSize  Unsigned8,
                                   parameter OCTET STRING SIZE (8) }

```

- pNo field

pNo

This field shall contain the parameter number to read out.

The value range shall be from 0 to 65 535.

- pSize field

pSize

This field shall contain an octet size of the parameter.

The value range shall be from 0 to 255.

- parameter field

parameter

This field shall contain the read data corresponding to pNo. The data size, data structure and its meaning depend on the pNo.

4.3.1.3 PRM_WR command and response

PRM_WR symbolizes a "parameter-write" command. Receiving this command, the slave shall write a specified parameter value from PRM_WR-CMD1-PDU.parameter field and shall respond with the echo of it to tell completion of the command.

```

_PRm_WR-CMD1-PDU ::= _CMD1-PDU ( WITH COMPONENTS
                                   { ..., cmd (prm_wr),
                                   pduBody (cmdBody _PRm_WR-CMD1Body) } )
_PRm_WR-CMD1Body ::= SEQUENCE { reserve  OCTET STRING SIZE (3),
                                   pNo  Unsigned16,
                                   pSize Unsigned8,
                                   parameter OCTET STRING SIZE (8) }
_PRm_WR-RSP1-PDU ::= _RSP1-PDU (WITH COMPONENTS
                                   { ..., rcmd (prm_wr),
                                   pduBody (rspBody _PRm_WR-RSP1Body) } )
_PRm_WR-RSP1Body ::= SEQUENCE { pNo  Unsigned16,
                                   pSize Unsigned8,
                                   parameter OCTET STRING SIZE (8) }

```

- pNo field

pNo

This field shall contain the parameter number to write in.

The value range shall be from 0 to 65 535.

- pSize field

pSize

This field shall contain the octet size of the parameter.

The value range shall be from 0 to 255.

- parameter field

parameter

This field shall contain the data to write corresponding to pNo. The data size, data structure and its meaning depend on the pNo.

4.3.1.4 ID_RD command and response

ID_RD symbolizes a "device-ID-read" command. Receiving this command, the slave shall read a part of a specified item in the Device Information into ID_RD-RSP1-PDU.idData field and shall respond with it. The semantics and data types of the Device Information can be dependent on each device model, except idCode: 0, which shall specify the device model information.

```

_ID_RD-CMD1-PDU      ::= _CMD1-PDU (WITH COMPONENTS
                        { ..., cmd (id_rd),
                          pduBody (cmdBody _ID_RD-CMDBody) } )

_ID_RD-CMD1Body      ::= SEQUENCE { reserve1  OCTET STRING SIZE (3),
                                      idCode  Unsigned8,
                                      idOffset Unsigned8,
                                      idSize  Unsigned8,
                                      reserve2  OCTET STRING SIZE (8) }

_ID_RD-RSP1-PDU      ::= _RSP1-PDU (WITH COMPONENTS
                        { ..., rcmd (id_rd),
                          pduBody (rspBody _ID_RD-RSPBody) } )

_ID_RD-RSP1Body      ::= SEQUENCE { idCode  Unsigned8,
                                      idOffset Unsigned8,
                                      idSize  Unsigned8,
                                      idData  OCTET STRING SIZE (8) }
    
```

- idCode field

idCode

This field shall contain the device-ID code to read out.

The value range shall be from 0 to 255.

- '00'H: device model;
- '01'H to 'FF'H: reserved.

- idOffset field

idOffset

This field shall contain the offset address of the device-ID to read out.

The value range shall be from 0 to 255.

- idSize field

idSize

This field shall contain the read octet size of the device-ID.

The value range shall be from 0 to 255.

- idData field

idData

This field shall contain the read data corresponding to the idCode. The data size, data structure and its meaning depend on the idCode.

4.3.1.5 CONFIG command and response

CONFIG symbolizes a "configure-device" command. Receiving this command, the slave shall activate a set of parameters on RAM updated with a PAR_WR command and shall respond to tell completion of the command.

```

_CONFIG-CMD1-PDU      ::= _CMD1-PDU (WITH COMPONENTS
                                { ..., cmd (config),
                                  pduBody (cmdBody _CONFIG-CMD1Body) } )
_CONFIG-CMD1Body      ::= SEQUENCE { reserve1  OCTET STRING SIZE (3),
                                config_mode Unsigned8 { pActive (0),
                                reserve2  OCTET STRING SIZE (10) }
_CONFIG-RSP1-PDU      ::= _RSP1-PDU (WITH COMPONENTS { ..., rcmd (config) } )

```

- config_mode field

config_mode

This field shall contain the mode of device configuration.

This field shall be coded as data type Unsigned8 with the following value range.

- pActive ('00'H): to recalculate with the parameters and set up;
- '01'H to 'FF'H: reserved.

4.3.1.6 ALM_RD command and response

ALM_RD symbolizes an "alarm-read" command. Receiving this command, the slave shall read a set of alarm data that includes details of status of alarm and warning into ALM_RD-RSP1-PDU.alm_data field and shall respond with it. The semantics and data types of the alm_data are dependent on each device model.

```

_ALM_RD-CMD1-PDU      ::= _CMD1-PDU (WITH COMPONENTS
                                { ..., cmd (alm_rd),
                                  pduBody (cmdBody _ALM_RD-CMD1Body) } )
_ALM_RD-CMD1Body      ::= SEQUENCE { reserve1  OCTET STRING SIZE (3),
                                alm_rd_mode Unsigned8 { currentAlm (0)},
                                reserve2  OCTET STRING SIZE (10) }
_ALM_RD-RSP1-PDU      ::= _RSP1-PDU (WITH COMPONENTS
                                { ..., rcmd (alm_rd),
                                  pduBody (rspBody _ALM_RD-RSP1Body) } )
_ALM_RD-RSP1Body      ::= SEQUENCE { alm_data Unsigned8 { currentAlm (0)},
                                alm_data  OCTET STRING SIZE (10) }

```

- alm_rd_mode field

alm_rd_mode

This field shall contain the mode for the alarm reading.

This field shall be coded as data type Unsigned8 with the following value range.

- currentAlm ('00'H): to read out the current alarm/warning state;
- 01'H to 'FF'H: reserved.

- alm_data field

alm_data

This field shall contain the read alarm status corresponding to alm_rd_mode