

DIN EN IEC 61158-6-3

ICS 35.100.70; 35.160

Ersatz für
DIN EN 61158-6-3:2015-09
Siehe Anwendungsbeginn

**Industrielle Kommunikationsnetze –
Feldbusse –
Teil 6-3: Protokollspezifikation des Application Layer
(Anwendungsschicht) – Typ 3-Elemente
(IEC 61158-6-3:2019);
Englische Fassung EN IEC 61158-6-3:2019, nur auf CD-ROM**

Industrial communication networks –
Fieldbus specifications –
Part 6-3: Application layer protocol specification – Type 3 elements
(IEC 61158-6-3:2019);
English version EN IEC 61158-6-3:2019, only on CD-ROM

Réseaux de communication industriels –
Spécifications des bus de terrain –
Partie 6-3: Spécification du protocole de la couche application – Éléments de type 3
(IEC 61158-6-3:2019);
Version anglaise EN IEC 61158-6-3:2019, seulement en CD-ROM

Gesamtumfang 387 Seiten

DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik in DIN und VDE

Anwendungsbeginn

Anwendungsbeginn für die von CENELEC am 2019-07-25 angenommene Europäische Norm als DIN-Norm ist 2020-08-01.

Für DIN EN 61158-6-3:2015-09 besteht eine Übergangsfrist bis 2022-07-25.

Nationales Vorwort

Vorausgegangener Norm-Entwurf: E DIN EN 61158-600:2017-12.

Dieser Entwurf betrifft auch die Normen DIN EN IEC 61158-6-2:2020-08, DIN EN IEC 61158-6-4:2020-08, DIN EN IEC 61158-6-10:2020-08, DIN EN IEC 61158-6-12:2020-08, DIN EN IEC 61158-6-19:2020-08, DIN EN IEC 61158-6-21:2020-08, DIN EN IEC 61158-6-23:2020-08, DIN EN IEC 61158-6-25:2020-08 und DIN EN IEC 61158-6-26:2020-08.

Für dieses Dokument ist das nationale Arbeitsgremium K 956 „Industrielle Kommunikation“ der DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik in DIN und VDE (www.dke.de) zuständig.

Die enthaltene IEC-Publikation wurde vom SC 65C „Industrial networks“ erarbeitet und unterliegt dem Copyright der IEC .

Das IEC-Komitee hat entschieden, dass der Inhalt dieses Dokuments bis zu dem Datum (stability date) unverändert bleiben soll, das auf der IEC-Website unter „<http://webstore.iec.ch>“ zu diesem Dokument angegeben ist. Zu diesem Zeitpunkt wird entsprechend der Entscheidung des Komitees das Dokument

- bestätigt,
- zurückgezogen,
- durch eine Folgeausgabe ersetzt oder
- geändert.

Für den Fall einer undatierten Verweisung im normativen Text (Verweisung auf ein Dokument ohne Angabe des Ausgabedatums und ohne Hinweis auf eine Abschnittsnummer, eine Tabelle, ein Bild usw.) bezieht sich die Verweisung auf die jeweils aktuellste Ausgabe des in Bezug genommenen Dokuments.

Für den Fall einer datierten Verweisung im normativen Text bezieht sich die Verweisung immer auf die in Bezug genommene Ausgabe des Dokuments.

Der Zusammenhang der zitierten Dokumente mit den entsprechenden deutschen Dokumenten ergibt sich, soweit ein Zusammenhang besteht, grundsätzlich über die Nummer der entsprechenden IEC-Publikation. Beispiel: IEC 60068 ist als EN 60068 als Europäische Norm durch CENELEC übernommen und als DIN EN 60068 ins Deutsche Normenwerk aufgenommen.

Das Präsidium des DIN hat mit Präsidialbeschluss 1/2004 festgelegt, dass DIN-Normen, deren Inhalt sich auf internationale Arbeitsergebnisse der Informationsverarbeitung gründet, unter bestimmten Bedingungen allein in englischer Sprache veröffentlicht werden dürfen. Diese Bedingungen sind für die vorliegende Norm erfüllt.

Da sich die Benutzer der vorliegenden Norm der englischen Sprache als Fachsprache bedienen, wird die Englische Fassung der EN IEC 61158-6-3 veröffentlicht. Zu deren Abschnitt 1, der den Anwendungsbereich festlegt und Abschnitt 3 (außer Unterabschnitte 3.3, 3.4 und 3.5), der die Begriffe festlegt, wurde eine Übersetzung angefertigt und als informativer Nationaler Anhang NA der vorliegenden Norm hinzugefügt. Für die meisten der verwendeten Begriffe existieren keine gebräuchlichen deutschen Benennungen, da sich die deutschen Anwender in der Regel ebenfalls der englischen Benennungen bedienen. Diese Norm steht nicht in unmittelbarem Zusammenhang mit Rechtsvorschriften und ist nicht als Sicherheitsnorm anzusehen.

Ein Feldbus ist ein digitaler serieller Datenbus für die Kommunikation zwischen Geräten der industriellen Automatisierungstechnik wie beispielsweise Messeinrichtungen, Stellgliedern, Reglern und speicherprogrammierbaren Steuerungen. Die Normen der Reihe DIN EN IEC 61158 legen eine Anzahl von Feldbus-Protokolltypen fest. Die vorliegende Norm legt die Dienste fest, die durch den Data-Link Layer der einzelnen Typen zur Verfügung gestellt werden.

Änderungen

Gegenüber DIN EN 61158-6-3:2015-09 wurden folgende Änderungen vorgenommen:

- a) verbesserte Ersetzungen in Tabelle 4;
- b) Verbesserungen in 5.3.14;
- c) Verbesserungen in 5.5.6;
- d) Verbesserungen in 5.17.15;
- e) Verbesserungen in 5.17.16.2;
- f) redaktionelle Verbesserungen.

Frühere Ausgaben

DIN V 19245-1: 1988-01
DIN 19245-1: 1991-04
DIN 19245-2: 1991-04
DIN EN 50170/1: 1997-07
DIN EN 50170/2: 1997-07
DIN EN 50170/3: 1997-07
DIN EN 50254: 1999-07
DIN EN 50170/A2: 2000-08
DIN EN 50170/A1: 2002-11
DIN EN 50170/A3: 2002-11
DIN EN 61158-6: 2005-05
DIN EN 61158-6-3: 2008-09, 2013-03, 2015-09

Nationaler Anhang NA (informativ)

Auszugsweise Übersetzung

Reihenfolge und Inhalt der folgenden Abschnitte sind identisch mit denen in den jeweiligen Abschnitten der Englischen Fassung.

EINLEITUNG

Dieser Teil von IEC 61158 gehört zu einer Normenreihe, die zur Erleichterung der Verbindung von Komponenten von Automatisierungssystemen erstellt wurde. Er ist mit anderen Normen der Reihe verbunden, wie durch das in IEC 61158-1 beschriebene „dreischichtige“ Feldbus-Referenzmodell definiert.

Das Anwendungsprotokoll stellt den Anwendungsdienst bereit, indem es die Dienste nutzt, die über die Datenverbindung oder eine andere unmittelbar darunter liegende Schicht verfügbar sind. Das vorrangige Ziel dieses Dokuments ist es, eine Reihe von Regeln für die Kommunikation zu erstellen, die sich auf die Verfahren beziehen, die von den Peer Application Entities (AE) zum Zeitpunkt der Kommunikation durchzuführen sind. Diese Regeln für die Kommunikation sollen eine stabile Grundlage für die Weiterentwicklung bilden, damit sie einer Vielzahl von Zwecken dienen können:

- als Leitfaden für Implementierer und Konstrukteure;
- zur Verwendung bei der Prüfung und Beschaffung von Ausrüstung;
- als Teil einer Vereinbarung über die Zulassung von Systemen in der offenen Systemumgebung;
- als eine Erweiterung des Verständnisses von zeitkritischer Kommunikation innerhalb der OSI.

Dieses Dokument befasst sich insbesondere mit der Kommunikation und dem Zusammenwirken von Sensoren, Effektoren und anderen Automatisierungsgeräten. Durch die Verwendung dieses Dokuments zusammen mit anderen Normen, die innerhalb der OSI- oder Feldbus-Referenzmodelle vorhanden sind, können andernfalls inkompatible Systeme in beliebiger Kombination zusammenarbeiten.

1 Anwendungsbereich

1.1 Allgemeines

Der Feldbus Application Layer (FAL, en: Fieldbus Application Layer) stellt Anwenderprogramme mit Mitteln zum Zugriff auf die Feldbuskommunikationsumgebung bereit. Insofern kann der FAL als ein „Fenster zwischen entsprechenden Anwendungsprogrammen“ angesehen werden.

Dieser Teil der IEC 61158 legt allgemeine Elemente für grundlegende zeitkritische und nicht zeitkritische Messaging-Kommunikationen zwischen Anwendungsprogrammen in einem Automatisierungsumfeld und für Typ-3-Feldbusse spezifisches Material fest. Der Begriff „zeitkritisch“ wird zum Angeben des Vorhandenseins eines Zeitfensters, in dem eine oder mehrere festgelegte Aktionen mit einem gewissen definierten Sicherheitsgrad abzuschließen sind, verwendet. Das Nichtabschließen festgelegter Aktionen in dem Zeitfenster riskiert den Ausfall der Anwendungen, die die Aktionen anfordern, zusammen mit einem begleitenden Risiko für Betriebsmittel, die Anlage und möglicherweise Menschenleben.

Dieses Internationale Norm definiert auf abstrakte Art und Weise das nach außen erkennbare Verhalten, das durch den Typ-3-Feldbus Application Layer bereitgestellt wird, hinsichtlich

- a) der abstrakten Syntax, die die zwischen kommunizierenden Anwendungsinstanzen übermittelten Application Layer-Protokolldateneinheiten definiert,
- b) der Übertragungssyntax, die die zwischen kommunizierenden Anwendungsinstanzen übermittelten Application Layer-Protokolldateneinheiten definiert,
- c) des Anwendungskontextzustandsautomaten, der das zwischen kommunizierenden Anwendungsinstanzen erkennbare Anwendungsdienstverhalten definiert; und

- d) der Anwendungsbeziehungszustandsautomaten, die das zwischen kommunizierenden Anwendungsinstanzen erkennbare Kommunikationsverhalten definieren.

Der Zweck dieses Dokuments ist es, das Protokoll zu definieren, das bereitgestellt wird, um

- a) die Drahtdarstellung der in IEC 61158-5-3 festgelegten Dienstprimitive zu definieren, und
- b) das nach außen erkennbare Verhalten zu definieren, das im Zusammenhang mit der entsprechenden Übertragung steht.

Dieses Dokument legt das Protokoll des Typ-3-Feldbus Application Layer in Übereinstimmung mit dem OSI-Basisreferenzmodell (ISO/IEC 7498-1) und der OSI-Anwendungsschichtstruktur (ISO/IEC 9545) fest.

FAL-Dienste und Protokolle werden durch FAL-Anwendungsinstanzen (AE, en: application entities), die in den Anwendungsprozessen enthalten sind, bereitgestellt. Die FAL-AE besteht aus einem Satz objektorientierter Anwendungsdienstelemente (ASE, en: Application Service Elements) und einer Einheit für Schichtenmanagement (LME, en: Layer Management Entity), die die AE verwaltet. Die ASE stellen Kommunikationsdienste bereit, die auf einem Satz zugehöriger Anwendungsprozessobjekt-(APO)-Klassen operieren. Eines der FAL-ASE ist ein Management-ASE, das einen gemeinsamen Satz von Diensten für die Verwaltung der Instanzen von FAL-Klassen bereitstellt.

Obwohl diese Dienste, aus dem Blickwinkel der Anwendung, beschreiben, wie Anfrage und Antwort ausgegeben und abgegeben werden, beinhalten sie keine Festlegung, was die anfragenden und antwortenden Anwendungen mit diesen vorzunehmen haben. Die Verhaltensaspekte der Anwendungen sind nicht festgelegt, nur eine Definition, welche Anfragen und Antworten diese senden/empfangen können, ist beschrieben. Dies erlaubt den FAL-Benutzern eine größere Flexibilität, um solch ein Objektverhalten zu standardisieren. Zusätzlich zu diesen Diensten sind in dieser Norm ebenfalls einige unterstützende Dienste festgelegt, um den Zugriff zum FAL bereitzustellen und somit bestimmte Aspekte des Betriebs zu steuern.

1.2 Spezifikationen

Das wesentliche Ziel dieses Dokuments ist es, die Syntax und das Verhalten des Application Layer-Protokolls, das die in IEC 61158-5-3 definierten Application Layer-Dienste übermittelt, festzulegen.

Ein Nebenziel ist es, Migrationspfade von zuvor bestehenden industriellen Kommunikationsprotokollen bereitzustellen. Es ist dieses letztgenannte Ziel, das zu der Vielfalt von Protokollen führt, die in Teilen der IEC 61158-6-Unterteile genormt sind.

1.3 Konformität

Dieses Dokument legt keine individuellen Implementierungen oder Produkte fest, noch beschränkt es die Implementierungen von Application Layer-Instanzen in industriellen Automatisierungssystemen.

Es gibt keine Betriebsmittelkonformität betreffend dieser Application Layer-Dienstdefinitionsnorm. Stattdessen wird Konformität durch Implementierung dieser Application Layer-Protokollspezifikation erreicht.

3 Begriffe, Abkürzungen, Symbole und Vereinbarungen

Für die Anwendung dieses Dokuments gelten die folgenden Begriffe, Abkürzungen, Symbole und Vereinbarungen.

ISO und IEC stellen terminologische Datenbanken für die Verwendung in der Normung unter den folgenden Adressen bereit:

- IEC Electropedia: verfügbar unter <http://www.electropedia.org/>
- ISO Online Browsing Platform: verfügbar unter <http://www.iso.org/obp>

3.1 Begriffe mit Verweisungen

3.1.1 Begriffe nach ISO/IEC 7498-1

Für die Anwendung dieses Dokuments gelten die folgenden Begriffe wie in ISO/IEC 7498-1 definiert:

- a) Anwendungsinstantz (en: application entity)
- b) Anwendungsprozess (en: application process)
- c) Anwendungsprotokolldateneinheit (en: application protocol data unit)
- d) Anwendungsdienstelement (en: application service element)
- e) Aufruf einer Anwendungsinstantz (en: application entity invocation)
- f) Aufruf eines Anwendungsprozesses (en: application process invocation)
- g) Anwendungstransaktion (en: application transaction)
- h) reales offenes System (en: real open system)
- i) Übertragungssyntax (en: transfer syntax)

3.1.2 Begriffe nach ISO/IEC 8822

Für die Anwendung dieses Dokuments gelten die folgenden Begriffe wie in ISO/IEC 8822 definiert:

- a) abstrakte Syntax (en: abstract syntax)
- b) Darstellungskontext (en: presentation context)

3.1.3 Begriffe nach ISO/IEC 9545

Für die Anwendung dieses Dokuments gelten die folgenden Begriffe wie in ISO/IEC 9545 definiert:

- a) Anwendungsassoziation (en: application-association)
- b) Anwendungskontext (en: application-context)
- c) Anwendungskontextname (en: application context name)
- d) Anwendungsinstanzaufruf (en: application-entity-invocation)
- e) Anwendungsinstantztyp (en: application-entity-type)
- f) Anwendungsprozessaufruf (en: application-process-invocation)
- g) Anwendungsprozessstyp (en: application-process-type)
- h) Anwendungsdienstelement (en: application-service-element)
- i) Anwendungssteuerdienstelement (en: application control service element)

3.1.4 Begriffe nach ISO/IEC 8824-1

Für die Anwendung dieses Dokuments gelten die folgenden Begriffe wie in ISO/IEC 8824-1 definiert:

- a) Objektbezeichner (en: object identifier)
- b) Typ (en: type)

3.1.5 Begriffe für Feldbus-Data-Link Layer

Für die Anwendung dieses Dokuments gelten die folgenden Begriffe wie in IEC 61158-3-3 und IEC 61158-4-3 definiert.

- a) DL-Zeit (en: DL-Time)
- b) DL-Zeitplanungsstrategie (en: DL-Scheduling-policy)
- c) DLCEP
- d) DLC
- e) DL-verbindungsorientierter Modus (en: DL-connection-oriented mode)

- f) DLPDU
- g) DLSDU
- h) DLSAP
- i) Verbindungszweig (en: link)
- j) Netzwerkadresse (en: network address)
- k) Knotenadresse (en: node address)
- l) Knoten (en: node)
- m) geplant (en: scheduled)

3.2 Zusätzliche Definitionen

3.2.1

Zugriffsschutz

(en: access protection)

Beschränkung der Nutzung eines Anwendungsobjekts auf einen Client

3.2.2

Adresszuweisungstabelle

(en: address-assignment-table)

Zuordnung des internen E/A-Datenobjektspeichers des Clients zu den dezentralen Eingabe- und Ausgabedatenobjekten

3.2.3

Kanal

(en: channel)

einzelner physischer oder logischer Verbindungszweig eines Eingabe- oder Ausgabe-Anwendungsobjekts eines Servers zu dem Prozess

3.2.4

kanalbezogene Diagnose

(en: channel related diagnosis)

Informationen in Bezug auf ein spezifisches Element eines Eingabe- oder Ausgabe-Anwendungsobjektes, die zu Wartungszwecken bereitgestellt werden

BEISPIEL Gültigkeit von Daten

3.2.5

Konfigurationsprüfung

(en: configuration check)

Vergleich der erwarteten Strukturierung des E/A-Datenobjekts des Clients mit der tatsächlichen Strukturierung des E/A-Datenobjekts beim Server in der Anlaufphase

3.2.6

Konfigurationsfehler

(en: configuration fault)

durch den Server erkannter unzulässiger Unterschied zwischen der erwarteten Strukturierung des E/A-Datenobjekts und der tatsächlichen Strukturierung des E/A-Datenobjekts

3.2.7

Konfigurationskennung

(en: configuration identifier)

Repräsentation eines Teils von E/A-Daten eines einzelnen Eingabe- und/oder Ausgabemoduls eines Servers

3.2.8

konfigurationskennungsbezogene Diagnose

(en: configuration identifier related diagnosis)

umfasst alle modulspezifischen Daten, die serverseitig für Wartungszwecke verfügbar sind

3.2.9

Steuerbefehle

(en: control commands)

vom Client zum Server übertragene Aktionsaufrufe, um Ausgänge freizugeben, Eingänge zu blockieren (Freeze) und/oder Ausgänge zu synchronisieren

3.2.10

Datenkonsistenz

(en: data consistency)

Einrichtung für die kohärente Übertragung des Eingabe- oder Ausgabedatenobjekts zwischen Client und Server und für den kohärenten Zugriff auf das Eingabe- oder Ausgabedatenobjekt innerhalb von Client und Server

3.2.11

Data-eXchange-Broadcast

(en: Data-eXchange-Broadcast)

DXB

Dienst zur Übertragung von Informationen zwischen dem DP-Master (Klasse 1) und den zugeordneten DP-Slaves, der die Publisher- und Subscriber-Funktionalität in den DP-Slaves initiiert

3.2.12

standardmäßige DL-Adresse

(en: default DL-address)

Wert 126 als Anfangswert für die Data-Link-Adresse, der vor dem Betrieb mit einem DP-Master (Klasse 1) geändert werden muss (z. B. durch Zuweisung einer Data-Link-Adresse über den Feldbus)

3.2.13

gerätebezogene Diagnose

(en: device related diagnosis)

umfasst alle Daten in Bezug auf den gesamten DP-Slave, die serverseitig für Wartungszwecke verfügbar sind

3.2.14

Diagnose-Informationen

(en: diagnosis information)

alle Daten, die serverseitig für Wartungszwecke verfügbar sind

3.2.15

Sammlung von Diagnose-Informationen

(en: diagnosis information collection)

Systemdiagnoseinformationen, die clientseitig zusammengetragen werden

3.2.16

DP-Master (Klasse 1)

(en: DP-master (class 1))

Steuergerät, das mehrere DP-Slaves (Feldgeräte) steuert

Anmerkung 1 zum Begriff: Hierbei handelt es sich üblicherweise um eine speicherprogrammierbare Steuerung oder ein verteiltes Steuerungssystem.

3.2.17

DP-Master (Klasse 2)

(en: DP-master (class 1))

Steuergerät, das Konfigurationsdaten (Parametersätze) und Diagnose-Daten eines DP-Masters (Klasse 1) verwaltet und das zusätzlich alle Kommunikationsfunktionen eines DP-Masters (Klasse 1) ausführt

3.2.18

DP-Slave

(en: DP-slave)

Feldgerät, das einem DP-Master (Klasse 1) als Provider für zyklischen E/A-Datenaustausch zugewiesen werden kann und es könnten zusätzlich azyklische Funktionen und Alarmer bereitgestellt werden

3.2.19**DXB-Anforderung**

(en: DXB request)

Anforderung des Dienstes für den zyklischen Datenaustausch mit einem bestimmten DL-Dienst zwischen dem DP-Master (Klasse 1) und den zugeordneten DP-Slaves, mit der die Publisher-Funktionalität in den DP-Slaves initiiert wird

3.2.20**DXB-Antwort**

(en: DXB response)

Antwort des Dienstes für den zyklischen Datenaustausch, die nicht an die Einzeladresse des Anfordernden (DP-Master (Klasse 1)), sondern an die Rundsendeadresse (= 127) gesendet wird

3.2.21**Freeze**

(en: freeze)

Funktion an den DP-Slaves für den gleichzeitigen Datentransfer zwischen dem Eingabedatenobjekt und dem Prozess

3.2.22**Gruppe**

(en: group)

Menge von DP-Slaves, die eine Freeze- oder Sync-Funktion ausführen

3.2.23**E/A-Daten**

(en: I/O data)

Objekt, das zu Verarbeitungszwecken zyklisch übertragen wird

3.2.24**Identnummer**

(en: ident number)

Gerätetyp eines DP-Masters (Klasse 1) oder eines DP-Slaves

3.2.25**Index**

(en: index)

Adresse eines Objekts innerhalb eines Anwendungsprozesses

3.2.26**isochroner Modus**

(en: isochronous mode)

spezieller Betriebsmodus eines DP-Systems, der sowohl einen konstanten DP-Zyklus mit einem festen Plan für die zyklischen und azyklischen DP-Dienste als auch die Synchronisation der Anwendung im DP-Master (Klasse 1) und in den DP-Slaves mit diesem konstanten DP-Zyklus vorsieht

3.2.27**Master-Parametersatz**

(en: master parameter set)

Konfigurations- und Parametrisierungsdaten aller DP-Slaves, die dem entsprechenden DP-Master zugewiesen sind, sowie die Bus-Parameter

3.2.28**Modul**

(en: module)

adressierbare Einheit innerhalb des DP-Slaves

3.2.29

Prozessdaten

(en: process data)

Objekt(e), das/die bereits vorverarbeitet ist/sind und zu Informationszwecken oder zum Zweck der weiteren Verarbeitung azyklisch übertragen wird/werden

3.2.30

Publisher

(en: publisher)

Rolle eines CR-Endpunktes, in der dieser APDU auf den Feldbus für den Konsum durch einen oder mehrere Subscriber überträgt

Anmerkung 1 zum Begriff: Ein Publisher braucht die Identität oder die Anzahl der Subscriber nicht zu kennen und darf seine APDU mithilfe einer zugeordneten CR veröffentlichen.

3.2.31

tatsächliche Konfiguration

(en: real configuration)

Eingabe- und Ausgabedatenstruktur eines DP-Slaves, einschließlich der Festlegung von Datenkonsistenz

3.2.32

Subscriber

(en: subscriber)

Rolle eines CREP, in der dieser von einem Publisher produzierte APDU empfängt

3.2.33

Slot

(en: slot)

Adresse eines Moduls innerhalb eines DP-Slaves

3.2.34

Sync

(en: sync)

Funktion an den DP-Slaves für den gleichzeitigen Datentransfer zwischen dem Ausgabedatenobjekt und dem Prozess

English Version

**Industrial communication networks - Fieldbus specifications -
Part 6-3: Application layer protocol specification -
Type 3 elements
(IEC 61158-6-3:2019)**

Réseaux de communication industriels - Spécifications des
bus de terrain - Partie 6-3: Spécification du protocole de la
couche application - Éléments de type 3
(IEC 61158-6-3:2019)

Industrielle Kommunikationsnetze - Feldbusse -
Teil 6-3: Protokollspezifikation des Application Layer
(Anwendungsschicht) - Typ 3-Elemente
(IEC 61158-6-3:2019)

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

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

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

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



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

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

European foreword

The text of document 65C/948/FDIS, future edition 4 of IEC 61158-6-3, 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 61158-6-3:2019.

The following dates are fixed:

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

This document supersedes EN 61158-6-3:2014 and all of its amendments and corrigenda (if any).

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

Endorsement notice

The text of the International Standard IEC 61158-6-3:2019 was approved by CENELEC as a European Standard without any modification.

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

IEC 61158-1:2019	NOTE	Harmonized as EN IEC 61158-1:2019 (not modified).
IEC 61784-1	NOTE	Harmonized as EN IEC 61784-1.
IEC 61784-2	NOTE	Harmonized as EN IEC 61784-2.

Annex ZA (normative)

Normative references to international publications with their corresponding European publications

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

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

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

<u>Publication</u>	<u>Year</u>	<u>Title</u>	<u>EN/HD</u>	<u>Year</u>
IEC 61158-3-3	2014	Industrial communication networks - Fieldbus specifications - Part 3-3: Data-link layer service definition - Type 3 elements	EN 61158-3-3	2014
IEC 61158-4-3	2019	Industrial communication networks - Fieldbus specifications - Part 4-3: Data-link layer protocol specification - Type 3 elements	EN IEC 61158-4-3	2019
IEC 61158-5-3	2014	Industrial communication networks - Fieldbus specifications - Part 5-3: Application layer service definition - Type 3 elements	EN 61158-5-3	2014
IEC 61158-5-10	2019	Industrial communication networks - Fieldbus specifications - Part 5-10: Application layer service definition - Type 10 elements	EN IEC 61158-5-10	2019
IEC 61158-6-10	2019	Industrial communication networks - Fieldbus specifications - Part 6-10: Application layer protocol specification - Type 10 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): 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 754	-	IEEE Standard for Binary Floating-Point-Arithmetic		-

CONTENTS

European foreword	2
Annex ZA (normative) Normative references to international publications with their corresponding European publications.....	3
INTRODUCTION	16
1 Scope.....	17
1.1 General	17
1.2 Specifications	18
1.3 Conformance	18
2 Normative references	18
3 Terms, definitions, abbreviations, symbols 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 ISO/IEC 9545 terms	19
3.1.4 ISO/IEC 8824-1 terms	20
3.1.5 Fieldbus Data Link Layer terms	20
3.2 Additional definitions	20
3.3 Abbreviations and symbols	23
3.4 Conventions	25
3.4.1 General concept	25
3.4.2 Abstract syntax conventions	25
3.4.3 Convention for the encoding of reserved bits and octets	26
3.4.4 Conventions for the common coding s of specific field octets.....	26
3.5 Conventions used in state machines	27
3.5.1 State machine conventions	27
4 FAL syntax description	29
4.1 APDU abstract syntax.....	29
4.2 Data types	34
4.2.1 Notation for the Boolean type	34
4.2.2 Notation for the Integer type	34
4.2.3 Notation for the Unsigned type	34
4.2.4 Notation for the Floating Point type	35
4.2.5 Notation for the OctetString type	35
4.2.6 Notation for VisibleString type	35
4.2.7 Notation for BinaryDate type	35
4.2.8 Notation for TimeOfDay type	35
4.2.9 Notation for TimeDifference type	35
4.2.10 Notation for Network Time type	35
4.2.11 Notation for Network Time Difference type	35
5 Transfer syntax	35
5.1 Coding of basic data types	35
5.1.1 Encoding of a Boolean value	35
5.1.2 Encoding of an Integer value	36
5.1.3 Encoding of an Unsigned value	36
5.1.4 Encoding of a Floating-Point value	36

5.1.5	Encoding of a Visible String value.....	36
5.1.6	Encoding of an Octet String value.....	36
5.1.7	Encoding of a BinaryDate value.....	36
5.1.8	Encoding of a TimeOfDay with and without date indication value	36
5.1.9	Encoding of a Time Difference with and without date indication value	36
5.1.10	Encoding of a Network Time value	37
5.1.11	Encoding of a Network Time Difference value	37
5.1.12	Encoding of a Null value.....	37
5.2	Coding section related to data exchange PDUs	37
5.2.1	General.....	37
5.2.2	Coding of the field Outp_Data	37
5.2.3	Coding of the field Inp_Data.....	37
5.3	Coding section related to slave diagnosis PDUs	37
5.3.1	Coding of the field Station_status_1	37
5.3.2	Coding of the field Station_status_2	38
5.3.3	Coding of the field Station_status_3	39
5.3.4	Coding of the field Diag_Master_Add	39
5.3.5	Coding of the field Ident_Number	39
5.3.6	Coding of the field Header_Octet.....	39
5.3.7	Coding of the field Alarm_Type.....	40
5.3.8	Coding of the field Status_Type	41
5.3.9	Coding of the field Slot_Number	41
5.3.10	Coding of the field Alarm_Specifier	41
5.3.11	Coding of the field Status_Specifier	42
5.3.12	Coding of the field Diagnosis_User_Data	43
5.3.13	Coding of the field Modul_Status_Array.....	43
5.3.14	Coding of the field Identifier_Diagnosis_Data_Array	44
5.3.15	Coding of the field Identifier_Number.....	45
5.3.16	Coding of the field Channel_Number	45
5.3.17	Coding of the field Type_of_Diagnosis	46
5.3.18	Coding of the field Revision_Number.....	46
5.3.19	Coding of the field Publisher_Address	46
5.3.20	Coding of the field Publisher_Status	47
5.3.21	Coding of the field RedSpecifier.....	47
5.3.22	Coding of the field Function.....	47
5.3.23	Coding of the field Red_Status1	48
5.3.24	Coding of the field Red_Status2	48
5.3.25	Coding of the field Red_Status3	49
5.4	Coding section related to parameterization PDU	49
5.4.1	Coding of the field Station_status	49
5.4.2	Coding of the field WD_Fact_1	50
5.4.3	Coding of the field WD_Fact_2	50
5.4.4	Coding of the field min_TSDR.....	50
5.4.5	Coding of the field Group_Ident	50
5.4.6	Coding of the field User_Prm_Data_Element	51
5.4.7	Coding of the field DPV1_Status_1.....	51
5.4.8	Coding of the field DPV1_Status_2.....	51
5.4.9	Coding of the field DPV1_Status_3.....	52

5.4.10	Coding of the field Structure_Length	53
5.4.11	Coding of the field Structure_Type	53
5.4.12	Coding of the field Version	53
5.4.13	Coding of the field Publisher_Addr	53
5.4.14	Coding of the field Publisher_Length	53
5.4.15	Coding of the field Sample_Offset	54
5.4.16	Coding of the field Sample_Length	54
5.4.17	Coding of the field Dest_Slot_Number	54
5.4.18	Coding of the field Offset_Data_Area	54
5.4.19	Coding of the field TBASE_DP	54
5.4.20	Coding of the field TDP	54
5.4.21	Coding of the field TMAPC	54
5.4.22	Coding of the field TBASE_IO	54
5.4.23	Coding of the field TI	54
5.4.24	Coding of the field TO	55
5.4.25	Coding of the field TDX	55
5.4.26	Coding of the field TPLL_W	55
5.4.27	Coding of the field TPLL_D	55
5.4.28	Coding of the field Specifier	55
5.4.29	Coding of the field Function	55
5.4.30	Coding of the field Properties	56
5.4.31	Coding of the field Output Hold Time	56
5.4.32	Coding of the field Clock Sync Interval	56
5.4.33	Coding of the field CS Delay Time	56
5.5	Coding section related to configuration PDUs	56
5.5.1	Coding of the field Cfg_Identifier	56
5.5.2	Coding of the field Special_Cfg_Identifier	57
5.5.3	Coding of the fields Length_Octet	58
5.5.4	Coding of the field Manufacturer_Specific_Data	58
5.5.5	Coding of the field Extended_Length_Octet	58
5.5.6	Coding of the field Data_Type	59
5.6	Coding section related to global control PDUs	59
5.6.1	Coding of the field Control_Command	59
5.6.2	Coding of the field Group_Select	60
5.7	Coding section related to clock-value-PDUs	60
5.7.1	Coding of the field Clock_value_time_event	60
5.7.2	Clock_value_previous_TE	60
5.7.3	Coding of the field Clock_value_status1	60
5.7.4	Coding of the field Clock_value_status2	61
5.8	Coding section related to function identification and errors	62
5.8.1	Coding of the field Function_Num	62
5.8.2	Coding of the field Error Decode	63
5.8.3	Coding of the field Error_Code_1	64
5.8.4	Coding of the field Error_Code_2	65
5.9	Coding section related to master diagnosis PDU	65
5.9.1	Coding of the field MDiag_Identifier	65
5.9.2	Coding of the field System_Diagnosis	65
5.9.3	Coding of the field USIF_State	66

5.9.4	Coding of the field Hardware_Release_DP	66
5.9.5	Coding of the field Firmware_Release_DP	66
5.9.6	Coding of the field Hardware_Release_User	66
5.9.7	Coding of the field Firmware_Release_User	66
5.9.8	Coding of the field Data_Transfer_List	66
5.10	Coding section related to upload/download/act para PDUs	67
5.10.1	Coding of the field Area_Code_UpDownload	67
5.10.2	Coding of the field Timeout	67
5.10.3	Coding of the field Max_Len_Data_Unit	67
5.10.4	Coding of the field Add_Offset	67
5.10.5	Coding of the field Data	67
5.10.6	Coding of the field Data_Len	67
5.10.7	Coding of the field Area_CodeActBrct	67
5.10.8	Coding of the field Area_CodeAct	68
5.10.9	Coding of the field Activate	68
5.11	Coding section related to the bus parameter set	69
5.11.1	Coding of the field Bus_Para_Len	69
5.11.2	Coding of the field DL_Add	69
5.11.3	Coding of the field Data_rate	69
5.11.4	Coding of the fields TSL, min TSDR, max TSDR	69
5.11.5	Coding of the fields TQUI, TSET, G, HSA, max_retry_limit	69
5.11.6	Coding of the field TTR (Target Token Rotation time)	69
5.11.7	Coding of the field Bp_Flag (Busparameter flag)	69
5.11.8	Coding of the field Min_Slave_Interval	70
5.11.9	Coding of the field Poll_Timeout	70
5.11.10	Coding of the field Data_Control_Time	70
5.11.11	Coding of the field Alarm_Max	70
5.11.12	Coding of the field Max_User_Global_Control	70
5.11.13	Coding of the field Master_User_Data_Len	70
5.11.14	Coding of the field Master_Class2_Name	70
5.11.15	Coding of the field Master_User_Data	71
5.11.16	Coding of the field TCT	71
5.11.17	Coding of the field maxTSH	71
5.12	Coding section related to the slave parameter set	71
5.12.1	Coding of the field Slave_Para_Len	71
5.12.2	Coding of the field SI_Flag (slave flag)	71
5.12.3	Coding of the field Slave_Type	72
5.12.4	Coding of the field Max_Diag_Data_Len	72
5.12.5	Coding of the field Max_Alarm_Len	72
5.12.6	Coding of the field Max_Channel_Data_Length	72
5.12.7	Coding of the field Diag_Upd_Delay	72
5.12.8	Coding of the field Alarm_Mode	72
5.12.9	Coding of the field Add_SI_Flag	73
5.12.10	Coding of the field MS1_Timeout	73
5.12.11	Coding of the field Prm_Data_Len	73
5.12.12	Coding of the field Prm_Data	73
5.12.13	Coding of the field Cfg_Data_Len	73
5.12.14	Coding of the field Cfg_Data	73

5.12.15	Coding of the field Add_Tab_Len	74
5.12.16	Coding of the field Number_of_Entries	74
5.12.17	Coding of the field Add_Tab_Entry_Header	74
5.12.18	Coding of the field I/O_Data_Length	74
5.12.19	Coding of the field I/O_Config_Address	74
5.12.20	Coding of the field Host_Address	74
5.12.21	Coding of the field Slave_User_Data_Len	74
5.12.22	Coding of the field Slave_User_Data	75
5.12.23	Coding of the field Ext_Prm_Data_Len	75
5.12.24	Coding of the field Ext_Prm_Data	75
5.13	Coding section related to statistic counters	75
5.13.1	Coding of the field DLPDU_sent_count and SD_count	75
5.13.2	Coding of the field Error_count and SD_error_count	75
5.14	Coding section related to set slave address PDU	75
5.14.1	Coding of the field New_Slave_Add	75
5.14.2	Coding of the field No_Add_Change	75
5.14.3	Coding of the field Rem_Slave_Data	75
5.15	Coding section related to initiate/abort PDUs	75
5.15.1	Coding of the field Features_Supported_1	75
5.15.2	Coding of the field Features_Supported_2	76
5.15.3	Coding of the field Profile_Features_Supported_1	76
5.15.4	Coding of the field Profile_Features_Supported_2	76
5.15.5	Coding of the field Profile_Ident_Number	76
5.15.6	Coding of the field S_Type (source type)	76
5.15.7	Coding of the field D_Type (destination type)	76
5.15.8	Coding of the field S_Len (source length)	76
5.15.9	Coding of the field D_Len (destination length)	76
5.15.10	Coding of the field S_API (source application identifier)	76
5.15.11	Coding of the field D_API (destination application identifier)	76
5.15.12	Coding of the field S_SCL (source security level)	77
5.15.13	Coding of the field D_SCL (destination security level)	77
5.15.14	Coding of the field S_Network_Address	77
5.15.15	Coding of the field D_Network_Address	77
5.15.16	Coding of the field S_MAC_Address	77
5.15.17	Coding of the field D_MAC_Address	77
5.15.18	Coding of the field Send_Timeout	77
5.15.19	Coding of the field Server_SAP	77
5.15.20	Coding of the field Subnet	77
5.15.21	Coding of the field Instance_Reason_Code	78
5.16	Coding section related to read/write/data transport PDUs	79
5.16.1	Coding of the field Index	79
5.16.2	Coding of the field Length	79
5.17	Coding section related to load region and function invocation PDUs	79
5.17.1	Coding of the field Extended_Function_Num	79
5.17.2	Coding of the field Options	79
5.17.3	Coding of the field Sequence_Number	80
5.17.4	Coding of the field LR_Data	80
5.17.5	Coding of the field Max_Segment_Length	80

5.17.6	Coding of the field LR_Index	80
5.17.7	Coding of the field LR_Length	80
5.17.8	Coding of the field Max_Response_Delay	80
5.17.9	Coding of the field Intersegment_Request_Timeout	80
5.17.10	Coding of the field User_Specific	80
5.17.11	Coding of the field FI_Index	80
5.17.12	Coding of the field Entity Number	80
5.17.13	Coding of the field Execution_Argument	81
5.17.14	Coding of the field Result_Argument	81
5.17.15	Coding of the field FI_State	81
5.17.16	Coding of the field IMData_Execution_Argument	81
5.17.17	Coding of the field IMData_Result_Argument	82
5.18	Examples of Diagnosis-RES-PDUs	83
5.19	Example of Chk_Cfg-REQ-PDU	85
5.20	Examples of Chk_Cfg-REQ-PDUs with DPV1 data types	85
5.21	Example structure of the Data_Unit for Data_Exchange	87
6	FAL protocol state machines	88
6.1	Overall structure	88
6.1.1	Fieldbus Service Protocol Machines (FSPM)	88
6.1.2	Master to Slave cyclic (MS0)	88
6.1.3	Master (class 1) to Slave acyclic (MS1)	88
6.1.4	Master (class 2) to Slave acyclic (MS2)	89
6.1.5	Master to Slave clock synchronisation (MS3)	89
6.1.6	Master Master acyclic (MM1/MM2)	89
6.1.7	DLL Mapping Protocol Machines (DMPM)	89
6.2	Assignment of state machines to devices	90
6.3	Overview DP-slave	90
6.4	Overview DP-master (class 1)	92
6.5	Overview DP-master (class 2)	93
6.6	Cyclic communication between DP-master (class 1) and DP-slave	94
6.7	Acyclic communication between DP-master (class 2) and DP-master (class 1)	95
6.8	Acyclic communication between DP-master (class 1) and DP-slave	97
6.9	Application relationship monitoring	99
6.9.1	Monitoring of the MS0 – AR	99
6.9.2	Monitoring of the MS2 – AR	99
7	AP-context state machine	103
8	FAL service protocol machines (FSPMs)	104
8.1	FSPMS	104
8.1.1	Primitive definitions	104
8.1.2	State machine description	110
8.1.3	FSPMS state table	112
8.1.4	Functions	138
8.2	FSPMM1	139
8.2.1	Primitive definitions	139
8.2.2	State machine description	145
8.2.3	FSPMM1 state table	148
8.2.4	Functions	174

8.3	FSPMM2	174
8.3.1	Primitive definitions.....	174
8.3.2	State machine description	179
8.3.3	FSPMM2 state table	179
8.3.4	Functions	191
9	Application relationship protocol machines (ARPMs).....	192
9.1	MSCY1S	192
9.1.1	Primitive definitions.....	192
9.1.2	State machine description	193
9.1.3	MSCY1S state table	199
9.1.4	Functions	220
9.2	MSAC1S	223
9.2.1	Primitive definitions.....	223
9.2.2	State machine description	225
9.2.3	MSAC1S state table	226
9.2.4	Functions	236
9.3	SSCY1S.....	236
9.3.1	Primitive definitions.....	236
9.3.2	State machine description	237
9.3.3	SSCY1S state table.....	237
9.3.4	Functions	239
9.4	MSRM2S.....	239
9.4.1	Primitive definitions.....	239
9.4.2	State machine description	240
9.4.3	MSRM2S state table.....	243
9.5	MSAC2S	245
9.5.1	Primitive definitions.....	245
9.5.2	State machine description	248
9.5.3	MSAC2S state table	250
9.6	MSCS1S	262
9.6.1	Primitive definitions.....	262
9.6.2	State machine description	262
9.6.3	MSCS1S state table	263
9.7	MSCY1M.....	264
9.7.1	Primitive definitions.....	264
9.7.2	State machine description	266
9.7.3	MSCY1M state table.....	268
9.8	MSAL1M	283
9.8.1	Primitive definitions.....	283
9.8.2	State machine description	284
9.8.3	MSAL1M state table	287
9.9	MSAC1M.....	292
9.9.1	Primitive definitions.....	292
9.9.2	State machine description	293
9.9.3	MSAC1M state table.....	299
9.10	MMAC1	305
9.10.1	Primitive definitions.....	305
9.10.2	State machine description	307

9.10.3	MMAC1 state table	307
9.11	MSCS1M	312
9.11.1	Primitive definitions	312
9.11.2	State machine description	313
9.11.3	MSCS1M state table	314
9.12	MSAC2M	317
9.12.1	Primitive definitions	317
9.12.2	State machine description	319
9.12.3	MSAC2M state table	322
9.13	MMAC2	332
9.13.1	Primitive definitions	332
9.13.2	State machine description	334
9.13.3	MMAC2 state table	334
10	DLL mapping protocol machines (DMPMs)	339
10.1	DMPMS	339
10.1.1	Primitive definitions	339
10.1.2	State machine description	345
10.1.3	DMPMS state table	345
10.1.4	Functions	351
10.2	DMPMM1	352
10.2.1	Primitive definitions	352
10.2.2	State machine description	359
10.2.3	DMPMM1 state table	360
10.2.4	Functions	367
10.3	DMPMM2	368
10.3.1	Primitive definitions	368
10.3.2	State machine description	372
10.3.3	DMPMM2 state table	372
10.3.4	Functions	375
11	Parameters for a DP-slave	376
	Bibliography	377
	Figure 1 – Common structure of specific fields	26
	Figure 2 – Example Modul_Status_Array	44
	Figure 3 – Example of Ext_Diag_Data in case of DPV1 diagnosis format with alarm and status PDU	83
	Figure 4 – Example of Ext_Diag_Data in case of the basic diagnosis format	84
	Figure 5 – Example of a special identifier format	85
	Figure 6 – Example of a special identifier format with data types	85
	Figure 7 – Example of a special identifier format with data types	86
	Figure 8 – Example of an empty slot with data types	86
	Figure 9 – Example for multi-variable device with AI and DO function blocks	87
	Figure 10 – Identifiers (ID)	87
	Figure 11 – Identifier list	87
	Figure 12 – Structure of the Data_Unit for the request- and response-DLPDU	88
	Figure 13 – Structuring of the protocol machines and adjacent layers in a DP-slave	91
	Figure 14 – Structuring of the protocol machines and adjacent layers in a DP-master (class 1)	92

Figure 15 – Structuring of the protocol machines and adjacent layers in a DP-master (class 2)	93
Figure 16 – Sequence of the communication between DP-master and DP-slave	95
Figure 17 – Sequence of communication between DP-master (class 2) and DP-master (class 1)	97
Figure 18 – Sequence of acyclic communication between DP-master (class 1) and DP-slave	98
Figure 19 – Example for connection establishment on MS2	101
Figure 20 – Idle at master-side on MS2	102
Figure 21 – Idle at slave-side on MS2	103
Figure 22 – Example for connection establishment on MS2(server-side).....	241
Figure 23 – Structure of RM entries in the RM_Registry.....	242
Table 1 – State machine description elements.....	27
Table 2 – Description of state machine elements.....	27
Table 3 – Conventions used in state machines	28
Table 4 – APDU syntax	30
Table 5 – Substitutions	32
Table 6 – Block_Length for Selection:= 0	39
Table 7 – Block_Length for Selection:= 1	39
Table 8 – Block_Length for Selection:= 2	40
Table 9 – Block_Length for Selection:= 3	40
Table 10 – Selection range	40
Table 11 – Alarm_Type range	41
Table 12 – Status_Type value range	41
Table 13 – Alarm_Specifier	42
Table 14 – Additional_Acknowledge.....	42
Table 15 – Status_Specifier.....	42
Table 16 – Range of Modul_Status_Entry (1-4).....	44
Table 17 – Input_Output_Selection	46
Table 18 – Error type	46
Table 19 – Channel_Type	46
Table 20 – Specification of the bits Lock_Req and Unlock_Req.....	50
Table 21 – Range of Length_of_Manufacturer_Specific_Data if used in Chk_Cfg-REQ-PDU	57
Table 22 – Range of Length_of_Manufacturer_Specific_Data if used in Get_Cfg-RES-PDU	57
Table 23 – Input_Output_Selection	58
Table 24 – Data types.....	59
Table 25 – Specification of the bits for Un-/Freeze	59
Table 26 – Specification of the bits for Un-/Sync.....	60
Table 27 – Coding of the Function_Code/ Function_Num	62
Table 28 – Coding of the Error_Code / Function_Num	63
Table 29 – Values of Error_Decode	63
Table 30 – Coding of Error_Code_1 at DPV1	64
Table 31 – Values of MDiag_Identifier	65
Table 32 – Values for Area_Code_UpDownload.....	67

Table 33 – Values for Area_CodeActBrct	68
Table 34 – Values for Area_CodeAct.....	68
Table 35 – Values for Activate	68
Table 36 – Values for Data_rate	69
Table 37 – DPV1_Data_Types.....	71
Table 38 – Values for Slave_Type	72
Table 39 – Values for Alarm_Mode.....	73
Table 40 – Values for Subnet.....	77
Table 41 – Values of reason code if instance is DLL	78
Table 42 – Values of reason code if instance is MS2	78
Table 43 – Values of Extended_Function_Num.....	79
Table 44 – Values of FI_Index	80
Table 45 – Values of FI_State.....	81
Table 46 – IMData_Execution_Argument	81
Table 47 – IMData_Result_Argument.....	82
Table 48 – Assignment of state machines	90
Table 49 – Primitives issued by AP-Context to FSPMS.....	104
Table 50 – Primitives issued by FSPMS to AP-Context.....	106
Table 51 – FSPMS state table	113
Table 52 – Functions used by the FSPMS.....	138
Table 53 – Primitives issued by AP-Context to FSPMM1	139
Table 54 – Primitives issued by FSPMM1 to AP-Context.....	142
Table 55 – FSPMM1 state table.....	148
Table 56 – Functions used by the FSPMM1	174
Table 57 – Primitives issued by AP-Context to FSPMM2.....	174
Table 58 – Primitives issued by FSPMM2 to AP-Context.....	176
Table 59 – FSPMM2 state table.....	179
Table 60 – Functions used by the FSPMM2	191
Table 61 – Primitives issued by FSPMS to MSCY1S	192
Table 62 – Primitives issued by MSCY1S to FSPMS	192
Table 63 – Rules for DPV1_Status_1, DPV1_Status_2 and DPV1_Status_3 check	194
Table 64 – MSCY1S state table	199
Table 65 – Functions used by the MSCY1S	220
Table 66 – Primitives issued by FSPMS to MSAC1S	223
Table 67 – Primitives issued by MSAC1S to FSPMS	224
Table 68 – Primitives issued by MSCY1S to MSAC1S.....	224
Table 69 – Primitives issued by MSAC1S to MSCY1S.....	224
Table 70 – Parameter used with primitives exchanged between MSAC1S and MSCY1S.....	224
Table 71 – MSAC1S state table	226
Table 72 – Functions used by the MSAC1S	236
Table 73 – Primitives issued by FSPMS to SSCY1S.....	236
Table 74 – Primitives issued by SSCY1S to FSPMS.....	236

Table 75 – SSCY1S state table	238
Table 76 – Functions used by the SSCY1S	239
Table 77 – Primitives issued by FSPMS to MSRM2S	239
Table 78 – Primitives issued by MSRM2S to FSPMS	240
Table 79 – MSRM2S state table	243
Table 80 – Primitives issued by FSPMS to MSAC2S	246
Table 81 – Primitives issued by MSAC2S to FSPMS	247
Table 82 – Primitives issued by MSRM2S to MSAC2S	247
Table 83 – Primitives issued by MSAC2S to MSRM2S	248
Table 84 – Parameter used with primitives exchanged with MSAC2S	248
Table 85 – MSAC2S state table	251
Table 86 – Primitives issued by MSCS1S to FSPMS	262
Table 87 – MSCS1S state table	263
Table 88 – Primitives issued by FSPMM1 to MSCY1M	264
Table 89 – Primitives issued by MSCY1M to FSPMM1	265
Table 90 – Parameters used with primitives exchanged between FSPMM1 and MSCY1M	265
Table 91 – MSCY1M state table	268
Table 92 – Primitives issued by FSPMM1 to MSAL1M	283
Table 93 – Primitives issued by MSAL1M to FSPMM1	283
Table 94 – Primitives issued by MSCY1M to MSAL1M	284
Table 95 – Primitives issued by MSAL1M to MSCY1M	284
Table 96 – Parameter used with primitives exchanged between MSAL1M and MSCY1M	284
Table 97 – Possible values in the Alarm_State_Table	285
Table 98 – MSAL1M state table	287
Table 99 – Primitives issued by FSPMM1 to MSAC1M	292
Table 100 – Primitives issued by MSAC1M to FSPMM1	292
Table 101 – Primitives issued by MSAL1M to MSAC1M	293
Table 102 – Primitives issued by MSAC1M to MSAL1M	293
Table 103 – Parameter used with primitives exchanged between MSAL1M and MSCY1M	293
Table 104 – MSAC1M state table	299
Table 105 – Primitives issued by FSPMM1 to MMAC1	306
Table 106 – Primitives issued by MMAC1 to FSPMM1	306
Table 107 – MMAC1 state table	308
Table 108 – Primitives issued by FSPMM1 to MSCS1M	313
Table 109 – Primitives issued by MSCS1M to FSPMM1	313
Table 110 – MSCS1M state table	315
Table 111 – Primitives issued by FSPMM2 to MSAC2M	317
Table 112 – Primitives issued by MSAC2M to FSPMM2	318
Table 113 – Parameters used with primitives exchanged with MSAC2M	318
Table 114 – MSAC2M state table	322
Table 115 – Primitives issued by FSPMM2 to MMAC2	332
Table 116 – Primitives issued by MMAC2 to FSPMM2	333

Table 117 – Parameters used with primitives exchanged with MMAC2	334
Table 118 – MMAC2 state table	335
Table 119 – Primitives issued by FSPMS to DMPMS	340
Table 120 – Primitives issued by DMPMS to FSPMS	340
Table 121 – Primitives issued by MSCY1S to DMPMS	340
Table 122 – Primitives issued by DMPMS to MSCY1S	341
Table 123 – Primitives issued by DMPMS to SSCY1S	341
Table 124 – Primitives issued by MSAC1S, MSRM2S, MSAC2S to DMPMS	342
Table 125 – Primitives issued by DMPMS to MSAC1S, MSRM2S, MSAC2S	342
Table 126 – Primitives issued by DMPMS to MSCS1S	342
Table 127 – Primitives issued by DMPMS to DL	343
Table 128 – Primitives issued by DL to DMPMS	343
Table 129 – Parameters used with primitives exchanged with DMPMS	344
Table 130 – DMPMS state table	346
Table 131 – Functions used by the DMPMS	351
Table 132 – Primitives issued by FSPMM1 to DMPMM1	353
Table 133 – Primitives issued by DMPMM1 to FSPMM1	353
Table 134 – Primitives issued by MSCY1M to DMPMM1	354
Table 135 – Primitives issued by DMPMM1 to MSCY1M	354
Table 136 – Primitives issued by MSAL1M, MSAC1M to DMPMM1	355
Table 137 – Primitives issued by DMPMM1 to MSAL1M, MSAC1M	355
Table 138 – Primitives issued by MMAC1 to DMPMM1	355
Table 139 – Primitives issued by DMPMM1 to MMAC1	355
Table 140 – Primitives issued by MSCS1M to DMPMM1	356
Table 141 – Primitives issued by DMPMM1 to MSCS1M	356
Table 142 – Primitives issued by DMPMM1 to DL	356
Table 143 – Primitives issued by DL to DMPMM1	357
Table 144 – Parameters used with primitives exchanged with DMPMM1	358
Table 145 – Possible values of status	359
Table 146 – DMPMM1 state table	360
Table 147 – Functions used by the DMPMM1	368
Table 148 – Primitives issued by FSPMM2 to DMPMM2	369
Table 149 – Primitives issued by DMPMM2 to FSPMM2	369
Table 150 – Primitives issued by MSAC2M to DMPMM2	370
Table 151 – Primitives issued by DMPMM2 to MSAC2M	370
Table 152 – Primitives issued by MMAC2 to DMPMM2	370
Table 153 – Primitives issued by DMPMM2 to MMAC2	370
Table 154 – Primitives issued by DMPMM2 to DL	371
Table 155 – Primitives issued by DL to DMPMM2	371
Table 156 – Parameters used with primitives exchanged with DMPMM2	372
Table 157 – DMPMM2 state Table	372
Table 158 – Functions used by DMPMM2	376
Table 159 – Bus parameter/reaction times for a DP-slave	376

INTRODUCTION

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

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

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

INDUSTRIAL COMMUNICATION NETWORKS – FIELDBUS SPECIFICATIONS –

Part 6-3: Application layer protocol specification – Type 3 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 3 fieldbus. The term “time-critical” is used to represent the presence of a time-window, within which one or more specified actions are required to be completed with some defined level of certainty. Failure to complete specified actions within the time window risks failure of the applications requesting the actions, with attendant risk to equipment, plant and possibly human life.

This International Standard defines in an abstract way the externally visible behavior provided by the Type 3 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 specified in IEC 61158-5-3, and
- b) define the externally visible behavior associated with their transfer.

This document specifies the protocol of the Type 3 fieldbus application layer, in conformance with the OSI Basic Reference Model (ISO/IEC 7498-1) 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 61158-5-3.

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 parts of the IEC 61158-6 subparts.

1.3 Conformance

This 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 standard. Instead, conformance is achieved through implementation of this application layer protocol specification.

2 Normative references

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

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

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

IEC 61158-4-3:2019, *Industrial communication networks – Fieldbus specifications – Part 4-3: Data-link layer protocol specification – Type 3 elements*

IEC 61158-5-3:2014, *Industrial communication networks – Fieldbus specifications – Part 5-3: Application layer service definition – Type 3 elements*

IEC 61158-5-10:2019, *Industrial communication networks – Fieldbus specifications – Part 5-10: Application layer service definition – Type 10 elements*

IEC 61158-6-10:2019, *Industrial communication networks – Fieldbus specifications – Part 6-10: Application layer protocol specification – Type 10 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): 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 754, *IEEE Standard for Floating-Point Arithmetic*, available at <<http://www.ieee.org>> [viewed 2018-09-10]

3 Terms, definitions, abbreviations, symbols and conventions

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

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

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

3.1 Referenced terms and definitions

3.1.1 ISO/IEC 7498-1 terms

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

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

3.1.2 ISO/IEC 8822 terms

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

- a) abstract syntax
- b) presentation context

3.1.3 ISO/IEC 9545 terms

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

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

3.1.4 ISO/IEC 8824-1 terms

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

- a) object identifier
- b) type

3.1.5 Fieldbus Data Link Layer terms

For the purposes of this document, the following terms as defined in IEC 61158-3-3 and IEC 61158-4-3 apply.

- a) DL-Time
- b) DL-Scheduling-policy
- c) DLCEP
- d) DLC
- e) DL-connection-oriented mode
- f) DLPDU
- g) DLSDU
- h) DLSAP
- i) link
- j) network address
- k) node address
- l) node
- m) scheduled

3.2 Additional definitions

3.2.1

access protection

limitation of the usage of an application object to one client

3.2.2

address-assignment-table

mapping of the client's internal I/O-Data object storage to the decentralized input and output data objects

3.2.3

channel

single physical or logical link of an input or output application object of a server to the process

3.2.4

channel related diagnosis

information concerning a specific element of an input or output application object, provided for maintenance purposes

EXAMPLE: validity of data.

3.2.5

configuration check

comparison of the expected I/O-Data object structuring of the client with the real I/O-Data object structuring to the server in the start-up phase

3.2.6

configuration fault

an unacceptable difference between the expected I/O-Data object structuring and the real I/O-Data object structuring, as detected by the server

3.2.7

configuration identifier

representation of a portion of I/O Data of a single input- and/or output-module of a server.

3.2.8

configuration identifier related diagnosis

compromises all module specific data available at the server for maintenance purposes

3.2.9

control commands

action invocations transferred from client to server to clear outputs, freeze inputs and/or synchronize outputs

3.2.10

data consistency

means for coherent transmission and access of the input- or output-data object between and within client and server

3.2.11

Data-eXchange-Broadcast DXB

service for conveyance of information between the DP-master (Class 1) and the assigned DP-slaves initiating the Publisher and Subscriber functionality in the DP-slaves

3.2.12

default DL-address

value 126 as an initial value for DL-address, which has to be changed (e.g. by assignment of an DL-address via the fieldbus) before operation with a DP-master (class 1)

3.2.13

device related diagnosis

compromises all data related to the whole DP-slave available at the server for maintenance purposes

3.2.14

diagnosis information

all data available at the server for maintenance purposes

3.2.15

diagnosis information collection

system diagnosis information that is assembled at the client side

3.2.16

DP-master (class 1)

a controlling device which controls several DP-slaves (field devices)

Note 1 to entry: This is usually a programmable controller or a distributed control system.

3.2.17

DP-master (class 2)

controlling device which manages configuration data (parameter sets) and diagnosis data of a DP-master (Class 1), and that additionally performs all communication capabilities of a DP-master (Class 1)

3.2.18

DP-slave

field device that can be assigned to one DP-master (Class 1) as a provider for cyclic I/O data exchange, in addition acyclic functions and alarms could be provided

3.2.19

DXB request

cyclic data exchange service request with a specific DL service between the DP-master (Class 1) and the assigned DP-slaves initiating the Publisher functionality in the DP-slaves

3.2.20

DXB response

cyclic data exchange service response not to the individual address of the requester (DP-master (Class 1)) but to the broadcast address (=127)

3.2.21

freeze

function at the DP-slaves for simultaneous data transfer between the input data object and the process

3.2.22

group

set of DP-slaves which perform a Freeze or Sync function

3.2.23

I/O data

object designated to be transferred cyclically for the purpose of processing.

3.2.24

ident number

DP-master (Class 1) or DP-slave device type

3.2.25

index

address of an object within an application process

3.2.26

isochronous mode

special operation mode of a DP system that implies both a constant DP cycle with a fixed schedule of the cyclic and acyclic DP services, and the synchronisation of the application in the DP-master (Class 1) and the DP-slaves with this constant DP cycle

3.2.27

master parameter set

the configuration and parameterization data of all DP-slaves that are assigned to the corresponding DP-master and the bus parameters

3.2.28

module

addressable unit inside the DP-slave

3.2.29

process data

object(s) which are already pre-processed and transferred acyclically for the purpose of information or further processing

3.2.30

publisher

role of a CR endpoint that transmits APDUs onto the Fieldbus for consumption by one or more subscribers

Note 1 to entry: A publisher may not be aware of the identity or the number of subscribers and it may publish its APDUs using a dedicated CR.

3.2.31

real configuration

input and output data structure of a DP-slave, including definition of data consistency

3.2.32

subscriber

role of a CREP in which it receives APDUs produced by a publisher

3.2.33

slot

address of a module within a DP-slave

3.2.34

sync

function at the DP-slaves for simultaneous data transfer between the output data object and the process

3.3 Abbreviations and symbols

Ack	Acknowledgement
Add	Address
AP	Application Process
APDU	Application Protocol Data Unit
API	Application Process Identifier
AR	Application Relationship
AREP	Application Relationship End Point
ARL	Application Relationship List
ASE	Application Service Element
Cfg	Configuration
cnf	confirmation primitive
CREP	Communication Relationship End Point
CRL	Communication Relationship List
D_	Destination
DLL	Data Link Layer
DLM	Data Link-management
DLSAP	Data Link Service Access Point
DLSDU	Data Link Service Data Unit
DMPM	Data Link Mapping Protocol Machine
DMPMM1	Data Link Mapping Protocol Machine DP-master (Class 1)
DMPMM2	Data Link Mapping Protocol Machine DP-master (Class 2)
DMPMS	Data Link Mapping Protocol Machine DP-slave
DSAP	Destination Service Access Point

DXB	Data eXchange Broadcast
FAL	Fieldbus Application Layer
FIFO	First In First Out
FSPMM1	FAL Service Protocol Machine DP-master (Class 1)
FSPMM2	FAL Service Protocol Machine DP-master (Class 2)
FSPMS	FAL Service Protocol Machine DP-slave
HSA	Highest Station Address
I&M	Identification and Maintenance Profile
ID	Identifier
IEC	International Electrotechnical Commission
ind	indication primitive
Inp	Input
ISO	International Organization For Standardization
IsoM	Isochronous Mode
L_sdu	Link Service Data Unit
lsb	least significant bit
msb	most significant bit
OSI	Open Systems Interconnection
Outp	Output
Param	Parameter
PDU	Protocol Data Unit
RD_	Read
Rem	Remote
Req	Request (used to indicate an APDU type)
req	request primitive
REQ	Request to indicate a Request PDU
RES	Response to indicate a Response PDU
RM	Resource Manager
RPL_UPD	Reply Update
Rsp	Response (used to indicate an APDU type)
rsp	response primitive
S_	Source
SAP	Service Access Point
SCL	Security Level
SD	Start Delimiter
SDN	Send Data with no Acknowledge
SDR	Station Delay Responder
Seq	Sequence
Src	Source

SRD	Send and Request Data with Reply
SSAP	Source Service Access Point
TR	Technical Report
USIF	User Interface
WD	Watchdog

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 61158-5-3. 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 61158-5-3. 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 Abstract syntax conventions

PDUs are described as octets or groups of octets.

- a) Groups of octets separated by a comma appear in the order they are transferred
If optional octets are not present the following octets appear without a gap
- b) If octets or groups of octets are grouped within "{" the order is arbitrary
- c) If octets or groups of octets are marked with "*" they may appear more than once.
If it is used within a "{" section they may appear mixed with other octets or group of octets of this section.
- d) Octets can be grouped or values can be assigned within "(")"
- e) If octets or groups of octets are grouped within "["]" the group can be omitted
- f) Complex APDUs may be built by means of substitutions (sub-structures)
- g) Exclusive selections of octets or groups of octets are separated by "^"

NOTE 1 The formal PDU example

AP_PDU=Octet1,OctetGroup1,[Octet2],[Octet3],[OctGroup2*],OctetGroup3^Octet4}
According to this the following variants are valid on the wire (non exhaustive):

Variant 1: Octet1, OctetGroup1, Octet2, Octet3, OctetGroup2,OctetGroup3

Variant 2: Octet1, OctetGroup1, Octet2, Octet3, OctetGroup2, OctetGroup2, OctetGroup2,OctetGroup3

Variant 3: Octet1, OctetGroup1, OctetGroup2, OctetGroup2, OctetGroup2,OctetGroup3, OctetGroup2

Variant4: Octet1, OctetGroup1, OctetGroup2, OctetGroup3, OctetGroup2, OctetGroup2, OctetGroup2, OctetGroup2

Variant 5: Octet1, OctetGroup1, Octet3, Octet4

NOTE 2 The arbitrary order implies that groups of octets are characterized by a special header that is described within the coding rules.

NOTE 3 The APDU syntax implies that according to the maximum DLSDU a APDU shall not exceed 244 octets in total.

3.4.3 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 check by a state machine.

3.4.4 Conventions for the common coding s of specific field octets

APDUs may contain specific fields that carry information in a primitive and condensed way. These fields shall be coded in the order according to Figure 1.

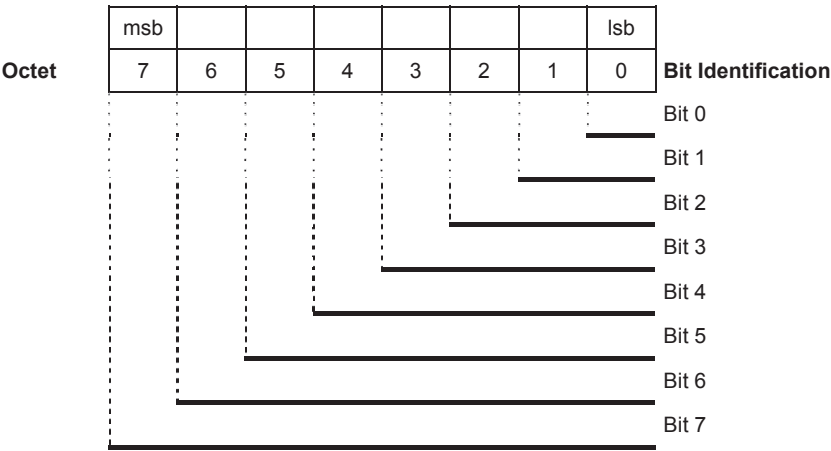


Figure 1 – Common structure of specific fields

- Bits may be grouped as group of bits. Each bit or group of bits shall be addressed by its Bit Identification (e.g. Bit 0, Bit 1 to 4). The position within the octet shall be according to the figure above. Alias names may be used for each bit or group of bits or they may be marked as reserved. The grouping of individual bits shall be in ascending order without gaps. The values for a group of bits may be represented as binary, decimal or hexadecimal values. This value shall only be valid for the grouped bits and can only represent the whole octet if all 8 bits are grouped. Decimal or hexadecimal values shall be transferred in binary values so that the bit with the highest number of the group represents the msb concerning the grouped bits.

EXAMPLE 1: Description and relation for the specific field octet

Bit 0: reserved.

Bit 1-3: Reason_Code The decimal value 2 for the Reason_Code means general error.

Bit 4-7: shall always set to one.

The octet that is constructed according to the description above looks as follows:

(msb) Bit 7 = 1,

Bit 6 = 1,

Bit 5 = 1,

Bit 4 = 1,

Bit 3 = 0,

Bit 2 = 1,

Bit 1 = 0,

(lsb) Bit 0 = 0.

The bit combination "0-1-0" for Bit 1-3 equals the decimal value 2.

3.5 Conventions used in state machines

3.5.1 State machine conventions

The protocol sequences are described by means of State Machines.

In state diagrams states are represented as boxes state transitions are shown as arrows. 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 follows, see also Table 1.

- The first row contains the name of the transition.
- The second row in define the current state.
- The third row contains an optional event followed by Conditions starting with a "/" as first line character and finally followed by the Actions starting with a "=>" as first line character.
- The last row contains the next state.
- If the event occurs and the conditions are fulfilled the transition fires, i.e. the actions are executed and the next state is entered.

The layout of a Machine description is shown in Table 1. The meaning of the elements of a State Machine Description are shown in Table 2.

Table 1 – State machine description elements

#	Current state	Event or condition => action	Next state

Table 2 – Description of state machine elements

Description element	Meaning
Current state	Name of the given states.
Next state	
#	Name or number of the state transition.
Event	Name or description of the event.
/Condition	Boolean expression. The preceding "/" is not part of the condition.
=> Action	List of assignments and service or function invocations. The preceding "=>" is not part of the action.

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

Table 3 – Conventions used in state machines

Convention	Meaning
:=	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.
xxx	A parameter name. Example: Identifier:= reason means value of a 'reason' parameter is assigned to a parameter called 'Identifier.'
"xxx"	Indicates fixed value. Example: Identifier:= "abc" means value "abc" is assigned to a parameter named 'Identifier.'
=	A logical condition to indicate an item on the left is equal to an item on the right.
<	A logical condition to indicate an item on the left is less than the item on the right.
>	A logical condition to indicate an item on the left is greater than the item on the right.
<>	A logical condition to indicate an item on the left is not equal to an item on the right.
&&	Logical "AND"
	Logical "OR"
for (Identifier:= start_value to end_value) actions endfor	This construct allows the execution of a sequence of actions in a loop within one transition. The loop is executed for all values from start_value to end_value.
If (condition) actions else actions	This construct allows the execution of alternative actions depending on some condition (which might be the value of some identifier or the outcome of a previous action) within one transition. The parts beginning with "else" can be omitted if there is no action if the condition is not fulfilled.

Readers are strongly recommended to refer to the subclauses for the AREP and CREP attribute definitions, the local functions, and the FAL-PDU definitions to understand protocol machines. It is assumed that readers have sufficient knowledge of these definitions and they are used without further explanations.

In addition the following description elements are used:

Wildcard in names

name_XXX: "XXX" is used as wildcard string for all names beginning with "name".

The typical use of a wildcard is an Event. In this context there are as many state transitions as possible events for this wildcard exists.

Conditional macro

<CONDITIONAL -MACRO-NAME>

<

CONDITION1: macrobody1

CONDITION2: macrobody2

...

>

CONDITION1, CONDITION2, etc. define all possible cases for the conditional macro. The “CONDITIONAL-MACRO-NAME” acts as place holder for the “macrocode” depending on the result of the condition.

Replacement macro

<REPLACEMENT-MACRO-NAME>

<

XXX= Name1: macrobody1

XXX= Name2: macrobody2

...

>

Name1, Name2, etc. define all possible cases for the replacement macro. The “REPLACEMENT-MACRO-NAME” acts as place holder for the “macrocode” depending on the value of the current use of the wildcard XXX.

Example:

<SERVICE_REQ_PARA>

<

XXX=Read: Para1, Para2

XXX=Write: Para1, Para2, Para3

>

when called in

MSAB_XXX.req(<SERVICE_REQ_PARA>)

will result in

MSAB_Read.req(Para1, Para2) or MSAB_Write.req(Para1, Para2, Para3)

4 FAL syntax description

4.1 APDU abstract syntax

Table 4 defines the abstract syntax of the DP Application Layer PDUs referred to as APDUs. The defined order of octets shall be used to convey the APDUs.

Table 4 – APDU syntax

APDU name	APDU structure
DataExchange-REQ-PDU	[Outp_Data*]
DataExchange-RES-PDU	[Inp_Data*]
RD_Input-RES-PDU	[Inp_Data*]
RD_Output-RES-PDU	[Outp_Data*]
Chk_Cfg-REQ-PDU	{ [Identifier_Format*], [Special_Identifier_Format*] ^ [Extended_Special_Identifier_Format*] } The field SI_Flag.DPV1_Data_Types defines whether Special_Identifier_Format or Extended_Special_Identifier_Format applies. The sequence of identifiers shall follow the ascending number of slots/modules.
Get_Cfg-RES-PDU	{ [Identifier_Format*], [Special_Identifier_Format*] ^ [Extended_Special_Identifier_Format*] }
Set_Prm-REQ-PDU	Station_status, WD_Fact_1, WD_Fact_2, min_TSDR, Ident_Number, Group_Ident, User_Prm_Data
Set_Ext_Prm-REQ-PDU	Structured_Prm_Data*
Diagnosis-RES-PDU	Station_status_1, Station_status_2, Station_status_3, Diag_Master_Add, Ident_Number, { [Device_Related_Diagnosis*] ^ ([Alarm], [Status*], [Modul_Status], [DXB_Link_Status], { [Red_Status] ^ [PrmCmdAck] }) }, [Identifier_Related_Diagnosis], [Channel_Related_Diagnosis*], [Revision_Number] } The field Device_Related_Diagnosis may be present if DPV1=FALSE, otherwise if DPV1=TRUE it shall not be present. In this case the fields Alarm, Status, Modul_Status may be present.
Set_Slave_Add-REQ-PDU	New_Slave_Add, Ident_Number, No_Add_Chg, [Rem_Slave_Data*]
Global_Control-REQ-PDU	Control_Command, Group_Select
Clock-Value-PDU	Clock_value_time_event, Clock_value_previous_TE, Clock_value_status1, Clock_value_status2
Initiate-REQ-PDU	Function_Num(0x57), reserved (3 Octets), Send_Timeout, Features_Supported, Profile_Features_Supported, Profile_Ident_Number, Add_Addr_Param
Initiate-RES-PDU	Function_Num(0x57), Max_Len_Data_Unit, Features_Supported, Profile_Features_Supported, Profile_Ident_Number, Add_Addr_Param
Initiate-NRS-PDU	Function_Num(0xD7), Error_Decode, Error_Code_1, Error_Code_2
Abort-REQ-PDU	Function_Num(0x58), Subnet, Instance_Reason_Code
Read-REQ-PDU	Function_Num(0x5E), Slot_Number, Index (0 .. 254), Length
Read-RES-PDU	Function_Num(0x5E), Slot_Number, Index (0 .. 254), Length, [Data*]
Read-NRS-PDU Pull-NRS-PDU	Function_Num(0xDE), Error_Decode, Error_Code_1, Error_Code_2
Write-REQ-PDU	Function_Num(0x5F), Slot_Number, Index (0 .. 254), Length, [Data*]
Write-RES-PDU	Function_Num(0x5F), Slot_Number, Index (0 .. 254), Length
Write-NRS-PDU Initiate_Load_NRS-PDU Push-NRS-PDU Terminate_Load-NRS-PDU Start-NRS-PDU Stop-NRS-PDU Resume-NRS-PDU Reset-NRS-PDU Call-NRS-PDU Get_FI_State-NRS-PDU	Function_Num(0xDF), Error_Decode, Error_Code_1, Error_Code_2
Alarm_Ack-REQ-PDU	Function_Num(0x5C), Slot_Number, Alarm_Type, Alarm_Specifier
Alarm_Ack-RES-PDU	Function_Num(0x5C), Slot_Number, Alarm_Type, Alarm_Specifier
Alarm_Ack-NRS-PDU	Function_Num(0xDC), Error_Decode, Error_Code_1, Error_Code_2
Idle-REQ-PDU	Function_Num(0x48)
Idle-RES-PDU	Function_Num(0x48)

APDU name	APDU structure
Data_Transport-REQ-PDU	Function_Num(0x51), Slot_Number, Index, Length, [Data*]
Data_Transport-RES-PDU	Function_Num(0x51), Slot_Number, Index, Length, [Data*]
Data_Transport-NRS-PDU	Function_Num(0xD1), Error_Decode, Error_Code_1, Error_Code_2
Initiate_Load-REQ-PDU	Function_Num(0x5F), Slot_Number, Index (255), Length, Extended_Function_Num (0x00), Options, LR_Index, LR_Length, Intersegment_Request_Timeout, [User_Specific]
Initiate_Load-RES-PDU	Function_Num(0x5E), Slot_Number, Index (255), Length, Extended_Function_Num (0x00), Max_Segment_Length, LR_Index, LR_Length, Max_Response_Delay
Push-REQ-PDU	Function_Num(0x5F), Slot_Number, Index (255), Length, Extended_Function_Num (0x01), Options, Sequence_Number, LR_Data
Pull-REQ-PDU	Function_Num(0x5E), Slot_Number, Index (255), Length
Pull-RES-PDU	Function_Num(0x5E), Slot_Number, Index (255), Length, Extended_Function_Num (0x02), Options, Sequence_Number, LR_Data
Terminate_Load-REQ-PDU	Function_Num(0x5F), Slot_Number, Index (255), Length, Extended_Function_Num (0x03), reserved (1 octet), reserved (2 Octets),
Start-REQ-PDU	Function_Num(0x5F), Slot_Number, Index (255), Length, Extended_Function_Num (0x04), reserved (1 octet), FI_Index, [Execution_Argument]
Stop-REQ-PDU	Function_Num(0x5F), Slot_Number, Index (255), Length, Extended_Function_Num (0x05), reserved (1 octet), FI_Index
Resume-REQ-PDU	Function_Num(0x5F), Slot_Number, Index (255), Length, Extended_Function_Num (0x06), reserved (1 octet), FI_Index, [Execution_Argument]
Reset-REQ-PDU	Function_Num(0x5F), Slot_Number, Index (255), Length, Extended_Function_Num (0x07), reserved (1 octet), FI_Index
Call-REQ-PDU	Function_Num(0x5F), Slot_Number, Index (255), Length, Extended_Function_Num (0x08), Entity Number, FI_Index (=0...64999), [Execution_Argument] Function_Num(0x5F), Slot_Number, Index (255), Length, Extended_Function_Num (0x08), Entity Number, FI_Index (=65000...65199), [IMData_Execution_Argument]
Call-RES-PDU	Function_Num(0x5E), Slot_Number, Index (255), Length, Extended_Function_Num (0x08), Entity Number, FI_Index (=0...64999), [Result_Argument] Function_Num(0x5E), Slot_Number, Index (255), Length, Extended_Function_Num (0x08), Entity Number, FI_Index (=65000...65199), [IMData_Result_Argument]
Get_FI_State-REQ-PDU	Function_Num(0x5F), Slot_Number, Index (255), Length, Extended_Function_Num (0x09), reserved (1 octet), FI_Index
Get_FI_State-RES-PDU	Function_Num(0x5E), Slot_Number, Index (255), Length, Extended_Function_Num (0x09), reserved (1 octet), FI_Index, FI_State
Push-RES-PDU Terminate_Load-RES-PDU Start-RES-PDU Stop-RES-PDU Resume-RES-PDU Reset-RES-PDU	Function_Num(0x5F), Slot_Number, Index (255), Length
RM-REQ-PDU	Function_Num(0x56), Server_SAP, Send_Timeout
Get_Master_Diag-REQ-PDU	Function_Num(0x41), MDiag_Identifier
Get_Master_Diag-RES-PDU	Function_Num(0x41), MDiag_Identifier(=0..125), Diagnosis-RES-PDU Function_Num(0x41), MDiag_Identifier(=126), System_Diagnosis Function_Num(0x41), MDiag_Identifier(=127), Master_Status Function_Num(0x41), MDiag_Identifier(=128), Data_Transfer_List
Start_Seq-REQ-PDU	Function_Num(0x42), Area_CodeUpDownload, Timeout
Start_Seq-RES-PDU	Function_Num(0x42), Area_CodeUpDownload, Timeout, Max_Len_Data_Unit

APDU name	APDU structure
Download-REQ-PDU	Function_Num(0x43), Area_CodeUpDownload(=0..125), Add_Offset, DP_Slave_Parameter_Set Function_Num(0x43), Area_CodeUpDownload(=127), Add_Offset, Bus_Parameter_Set Function_Num(0x43), Area_CodeUpDownload(=129), Add_Offset, Statistic_Counters Function_Num(0x43), Area_CodeUpDownload(=136 to 139), Add_Offset, Master_Parameter_Set
Download-RES-PDU	Function_Num(0x43), Area_CodeUpDownload, Add_Offset
Upload-REQ-PDU	Function_Num(0x44), Area_CodeUpDownload, Add_Offset, Data_Len
Upload-RES-PDU	Function_Num(0x44), Area_CodeUpDownload(=0..125), Add_Offset, DP_Slave_Parameter_Set Function_Num(0x44), Area_CodeUpDownload(=127), Add_Offset, Bus_Parameter_Set Function_Num(0x44), Area_CodeUpDownload(=129), Add_Offset, Statistic_Counters Function_Num(0x43), Area_CodeUpDownload(=136 to 139), Add_Offset, Master_Parameter_Set
End_Seq-REQ-PDU	Function_Num(0x45)
End_Seq-RES-PDU	Function_Num(0x45)
Act_Para_Brct-REQ-PDU	Function_Num(0x46), Area_CodeActBrct
Act_Param-REQ-PDU	Function_Num(0x47), Area_CodeAct, Activate
Act_Param-RES-PDU	Function_Num(0x47), Area_CodeAct, Activate
ZERO-PDU	Octet1(0)
NULL-PDU	
NOTE 1 If a APDU consists only of user data octets the distinction is made by use of different Data Link Layer service access points (see Protocol Machines).	
NOTE 2 A ZERO-PDU contains one octet with all bits set to zero. This APDU is used to indicate diagnosis from a device without inputs.	
NOTE 3 A NULL-PDU contains no octets. It is used to submit no DLSDU to data link layer.	

Table 5 defines structures for substitutions of elements of the APDU structures shown in Table 4.

Table 5 – Substitutions

Substitution name	Structure
Identifier_Format	Cfg_Identifier
Special_Identifier_Format	Special_Cfg_Identifier, [Length_Octet], [Length_Octet], [Manufacturer_Specific_Data*] Length_Octet fields shall always start with fields that indicate output data if present.
Extended_Special_Identifier_Format	Special_Cfg_Identifier, [Extended_Length_Octet], [Extended_Length_Octet], [Data_Type*], [Manufacturer_Specific_Data*] Extended_Length_Octet fields shall always start with fields that indicate output data if present. The field Data_Type shall only be omitted in case of an empty slot. Otherwise it shall be present in relation to the length fields.
Device_Related_Diagnosis	Header_Octet, Diagnosis_User_Data*
Identifier_Related_Diagnosis	Header_Octet, Identifier_Diagnosis_Data_Array
Channel_Related_Diagnosis	Identifier_Number, Channel_Number, Type_of_Diagnosis
Alarm	Header_Octet, Alarm_Type, Slot_Number, Alarm_Specifier, [Diagnosis_User_Data*]
Status	Header_Octet, Status_Type, Slot_Number, Status_Specifier, [Diagnosis_User_Data*]
Modul_Status	Header_Octet, Status_Type(=Modul_Status), Slot_Number(=0), Status_Specifier, Modul_Status_Array

Substitution name	Structure
DXB_Link_Status	Header_Octet, Status_Type(=DXB_Link_Status), Slot_Number(=0), Status_Specifier, (Publisher_Address, Publisher_Status)*
PrmCmdAck	Header_Octet, Status_Type(=PrmCmdAck), Slot_Number(=0), Status_Specifier, Function, Red_Status1, Red_Status2, Red_Status3
Red_Status	Header_Octet, Status_Type(=Red_Status), Slot_Number(=0), Status_Specifier, Function, Red_Status1, Red_Status2, Red_Status3
User_Prm_Data	[DPV1_Status_1, DPV1_Status_2, DPV1_Status_3], [Structured_Prm_Data*] ^ [User_Prm_Data_Element*] Special case DPV1=FALSE: [User_Prm_Data_Element*] Special case DPV1=TRUE and DPV1_Status_3.Prm_Structure=FALSE: DPV1_Status_1, DPV1_Status_2, DPV1_Status_3, [User_Prm_Data_Element*] Special case DPV1=TRUE and DPV1_Status_3.Prm_Structure=TRUE: DPV1_Status_1, DPV1_Status_2, DPV1_Status_3, Structured_Prm_Data* The fields DPV1_Status_1, DPV1_Status_2 and DPV1_Status_3 shall be present if DPV1=TRUE. Structured_Prm_Data shall be present if the DPV1_Status_3.Prm_Structure is TRUE, otherwise it shall not be used.
Structured_Prm_Data	Structure_Length, Structure_Type, Slot_Number, reserved (1 octet), User_Prm_Data_Element* Special Case PrmCmd: Structure_Length, Structure_Type(=2), Slot_Number(=0), Specifier, Function, Properties, Output_Hold_Time Special Case DXB LinkTable: Structure_Length, Structure_Type(=3), Slot_Number(=0), reserved (1 octet), Version(=1), (Publisher_Addr, Publisher_Length, Sample_Offset, Sample_Length)* Special Case IsoM Parameter: Structure_Length, Structure_Type(=4), Slot_Number(=0), reserved (1 octet), Version(=1), TBASE_DP, TDP, TMAPC, TBASE_IO, TI, TO, TDX, TPLL_W, TPLL_D Special Case F-Parameter: Structure_Length, Structure_Type(=5), Slot_Number, reserved (1 octet), User_Prm_Data* Special Case DXB-Subscribable: Structure_Length, Structure_Type(=7), Slot_Number(=0), reserved (1 octet), Version(=1), (Publisher_Addr, Publisher_Length, Sample_Offset, Dest_Slot_Number, Offset_Data_Area, Sample_Length)* Special Case Time AR Parameter: Structure_Length, Structure_Type(=8), Slot_Number(=0), reserved (1 octet), Clock Sync Interval, [CS Delay Time]
Features_Supported	Features_Supported_1, Features_Supported_2
Profile_Features_Supported	Profile_Features_Supported_1, Profile_Features_Supported_2
Add_Addr_Param	S_Type, S_Len, D_Type, D_Len, S_API, S_SCL, [S_Network_Address], [S_MAC_Address], D_API, D_SCL, [D_Network_Address], [D_MAC_Address]
Master_Status	USIF_State, Ident_Number, Hardware_Release_DP, Firmware_Release_DP, Hardware_Release_User, Firmware_Release_User, reserved (9 Octets)
DP_Slave_Parameter_Set	Slave_Para_Len, SI_Flag, Slave_Type, Max_Diag_Data_Len, Max_Alarm_Len, Max_Channel_Data_Length, Diag_Upd_Delay, Alarm_Mode, Add_SI_Flag, MS1_Timeout, reserved (4 Octets), Prm_Data_Len, Prm_Data, Cfg_Data_Len, Cfg_Data, Add_Tab_Len, Add_Tab, Slave_User_Data_Len, Slave_User_Data, Ext_Prm_Data_Len, [Ext_Prm_Data]
Bus_Parameter_Set	Bus_Para_Len, DL_Add, Data_rate, TSL, min TSDR, max TSDR, TQUI, TSET, TTR, G, HSA, max_retry_limit, Bp_Flag, Min_Slave_Interval, Poll_Timeout, Data_Control_Time, Alarm_Max, Max_User_Global_Control, reserved (4 Octets), Master_User_Data_Len, Master_Class2_Name, Master_User_Data*, TCT, maxTSH
Master_Parameter_Set	Bus_Parameter_Set, DP_Slave_Parameter_Set*
Statistic_Counters	Counter_Group*

Substitution name	Structure
Counter_Group	if Add_Offset 0..126: DLPDU_sent_count(Add_Offset), Error_count(Add_Offset) if Add_Offset 127: SD_sent_count, SD_error_count
Add_Tab	[Number_of_Entries], [Add_Tab_Entry_Header, I/O_Data_Length, IO_Config_Address, Host_Address]*
IMData_Execution_Argument	[I&M1] ^ [I&M2] ^ [I&M3] ^ [I&M4] ^ [I&M_Profile_Specific_x] ^ [I&M_Manufacturer_Specific_x]
IMData_Result_Argument	I&M0 ^ [I&M1] ^ [I&M2] ^ [I&M3] ^ [I&M4] ^ [I&M_Profile_Specific_x] ^ [I&M_Manufacturer_Specific_x]
I&M0	IM_Header, IM_Manufacturer_ID, OrderID, IM_Serial_Number, IM_Hardware_Revision, IM_Software_Revision, IM_Revision_Counter, IM_Profile_ID, IM_Profile_Specific_Type, IM_Version, IM_Supported
I&M1	IM_Header, IM_Tag_Function, IM_Tag_Location
I&M2	IM_Header, IM_Date
I&M3	IM_Header, IM_Descriptor
I&M4	IM_Header, IM_Signature
I&M_Profile_Specific_x	IM_Header, IM_Profile_Specific_Data
IM_Version	IM_Version_Major, IM_Version_Minor
IM_Software_Revision	SWRevisionPrefix, IM_SWRevision_Functional_Enhancement, IM_SWRevision_Bug_Fix, IM_SWRevision_Internal_Change
NOTE The DP_Slave_Parameter_Set and the Bus_Parameter_Set may exceed the maximum APDU size to a size up to 64 Koctets. The conveyance of the data is therefore segmented by means of a window addressed with the parameter Add_Offset.	

4.2 Data types

4.2.1 Notation for the Boolean type

Boolean::= BOOLEAN

-- TRUE if the value is non-zero.

-- FALSE if the value is zero.

4.2.2 Notation for the Integer type

Integer8::= INTEGER (-128..+127) -- range -2⁷ ≤ I ≤ 2⁷-1

Integer16::= INTEGER (-32768..+32767) -- range -2¹⁵ ≤ I ≤ 2¹⁵-1

Integer32::= INTEGER -- range -2³¹ ≤ I ≤ 2³¹-1

Integer64::= INTEGER -- range -2⁶³ ≤ I ≤ 2⁶³-1

4.2.3 Notation for the Unsigned type

Unsigned8::= INTEGER (0..255) -- range 0 ≤ I ≤ 2⁸-1

Unsigned16::= INTEGER (0..65535) -- range 0 ≤ I ≤ 2¹⁶-1

Unsigned32::= INTEGER -- range 0 ≤ I ≤ 2³²-1

Unsigned64::= INTEGER -- range 0 ≤ I ≤ 2⁶⁴-1

4.2.4 Notation for the Floating Point type

Floating32::= BIT STRING SIZE (4) -- IEEE 754 Single precision
Floating64::= BIT STRING SIZE (8) -- IEEE 754 Double precision

4.2.5 Notation for the OctetString type

OctetString::= OCTET STRING -- For generic use

4.2.6 Notation for VisibleString type

VisibleString2::= VISIBLE STRING -- For generic use

4.2.7 Notation for BinaryDate type

BinaryDate::= OctetString7

4.2.8 Notation for TimeOfDay type

TimeOfDay with date indication::= OctetString6
TimeOfDay without date indication::= OctetString4

4.2.9 Notation for TimeDifference type

TimeDifference with date indication::= OctetString6
TimeDifference without date indication::= OctetString4

4.2.10 Notation for Network Time type

Network Time::= OctetString8

4.2.11 Notation for Network Time Difference type

Network Time Difference::= OctetString8

5 Transfer syntax

5.1 Coding of basic data types

5.1.1 Encoding of a Boolean value

- a) The encoding of a Boolean value shall be primitive. The ContentsOctets shall consist of a single octet.
- b) If the Boolean value is FALSE, the ContentsOctets shall be 0 (zero). If the Boolean value is TRUE, the ContentsOctets shall be 0xff.

5.1.2 Encoding of an Integer value

- a) The encoding of a fixed-length Integer value of Integer8, Integer16, and Integer32 types shall be primitive, and the ContentsOctets shall consist of exactly one, two, or four octets, respectively.
- b) The ContentsOctets shall be a two's complement binary number equal to the integer value, and consist of bits 7 to 0 of the first octet, followed by bits 7 to 0 of the second octet, followed by bits 7 to 0 of each octet in turn up to and including the last octet of the ContentsOctets.

NOTE The value of a two's complement binary number is derived by numbering the bits in the ContentsOctets, starting with bit 0 of the last octet as bit zero and ending the numbering with bit 7 of the first octet. Each bit is assigned a numerical value of 2^N , where N is its position in the above numbering sequence. The value of the two's complement binary number is obtained by adding the numerical values assigned to each bit for those bits which are set to one, excluding bit 7 of the first octet, and then reducing this value by the numerical value assigned to bit 7 of the first octet if that bit is set to one.

5.1.3 Encoding of an Unsigned value

- a) The encoding of a fixed-length Unsigned value of Unsigned8, Unsigned16, and Unsigned32 types shall be primitive, and the ContentsOctets shall consist of exactly one, two, or four octets, respectively.
- b) The ContentsOctets shall be a binary number equal to the Unsigned value, and consist of bits 7 to 0 of the first octet, followed by bits 7 to 0 of the second octet, followed by bits 7 to 0 of each octet in turn up to and including the last octet of the ContentsOctets.

NOTE The value of a binary number is derived by numbering the bits in the ContentsOctets, starting with bit 0 of the last octet as bit zero and ending the numbering with bit 7 of the first octet. Each bit is assigned a numerical value of 2^N , where N is its position in the above numbering sequence. The value of the binary number is obtained by adding the numerical values assigned to each bit for those bits which are set to one.

5.1.4 Encoding of a Floating-Point value

- a) The encoding of a Floating32 value shall be primitive, and the ContentsOctets shall consist of exactly four octets.
- b) The ContentsOctets shall contain floating-point values defined in conformance with IEEE 754. The sign is encoded in bit 7 of the first octet. It is followed by the exponent starting from bit 6 of the first octet, and then the mantissa starting from bit 6 of the second octet for Floating32.

5.1.5 Encoding of a Visible String value

- a) The encoding of a variable length VisibleString value shall be primitive.
- b) There is no Length field; the length is encoded implicitly.
- c) The ContentsOctets shall be a sequence of octets. The leftmost string element is encoded in the first octet, followed by the second octet, followed by each octet in turn up to and including the last octet as rightmost of the ContentsOctets.

5.1.6 Encoding of an Octet String value

- a) The encoding of a variable length OctetString value shall be primitive.
- b) There is no Length field; the length is encoded implicitly.
- c) The ContentsOctets shall be a sequence of octets. The leftmost string element is encoded in the first octet, followed by second octet, followed by each octet in turn up to and including the last octet as rightmost of the ContentsOctets.

5.1.7 Encoding of a BinaryDate value

Coding of BinaryDate within IEC 61158-6-10 applies.

5.1.8 Encoding of a TimeOfDay with and without date indication value

Coding of TimeOfDay with and without date indication within IEC 61158-6-10 applies.

5.1.9 Encoding of a Time Difference with and without date indication value

Coding of Time Difference with and without date indication within IEC 61158-6-10 applies.

5.1.10 Encoding of a Network Time value

Coding of Network Time within IEC 61158-6-10 applies.

5.1.11 Encoding of a Network Time Difference value

Coding of Network Time Difference within IEC 61158-6-10 applies.

5.1.12 Encoding of a Null value

- a) The encoding of a NULL value shall be primitive.
- b) The ContentsOctet shall be omitted.

5.2 Coding section related to data exchange PDUs

5.2.1 General

IEC 61158-5-10, Clause 5 applies.

NOTE Data types with variable length (Visible String, Octet String, TimeOfDay, Time Difference) for Inp_Data or Outp_Data are only once present in the DataExchange-RES-PDU or DataExchange-REQ-PDU. The user data correspond to the contents of the Configuration-PDU.

5.2.2 Coding of the field Outp_Data

E.g. one of the following data types shall be used for each Outp_Data field:

Boolean, Integer, Unsigned, Floating Point, Visible String, Octet String, Date, TimeOfDay, Time-Difference.

5.2.3 Coding of the field Inp_Data

E.g. one of the following data types shall be used for each Inp_Data field:

Boolean, Integer, Unsigned, Floating Point, Visible String, Octet String, Date, TimeOfDay, Time-Difference.

5.3 Coding section related to slave diagnosis PDUs

5.3.1 Coding of the field Station_status_1

The coding of this field shall be according to 3.4 and the individual bits shall have the following meaning:

Bit 0: Diag.Station_Non_Existent

Shall be set to zero in case of Diagnosis-RES-PDU

Value (0): means FALSE if Get_Master_Diag-RES-PDU

Value (1): means TRUE if Get_Master_Diag-RES-PDU

Bit 1: Diag.Station_Not_Ready

FALSE (0)

TRUE (1)

Bit 2: Diag.Cfg_Fault (Configuration Fault)

FALSE (0)

TRUE (1)

Bit 3: Diag.Ext_Diag (Extended Diagnosis)

FALSE (0): means that the device is NOT in state diagnosis

TRUE (1): means that the device is in state diagnosis

Flag should be set to FALSE if no faulty conditions exist. If set to TRUE, the reason should be reported as Device Related Diagnosis, Identifier Related Diagnosis, or Channel Related Diagnosis.

Bit 4: Diag.Not_Supported

FALSE (0)

TRUE (1)

Bit 5: Diag.Invalid_Slave_Response

Shall be set to zero if Diagnosis-RES-PDU

Value (0): means FALSE if Get_Master_Diag-RES-PDU

Value (1): means TRUE if Get_Master_Diag-RES-PDU

Bit 6: Diag.Prm_Fault (Parameter Fault)

FALSE (0), TRUE (1)

Bit 7: Diag.Master_Lock

Shall be set to zero if Diagnosis-RES-PDU

Value (0): means FALSE if Get_Master_Diag-RES-PDU

Value (1): means TRUE if Get_Master_Diag-RES-PDU

5.3.2 Coding of the field Station_status_2

The coding of this field shall be according to 3.4 and the individual bits shall have the following meaning:

Bit 0: Diag.Prm_Req (Parameterization Requested)

FALSE (0)

TRUE (1)

Bit 1: Diag.Stat_Diag (Static Diagnosis)

FALSE (0)

TRUE (1)

Bit 2: DP (DP Protocol)

Shall always set to ones

Bit 3: Diag.WD_On (Watchdog on)

FALSE (0)

TRUE (1)

Bit 4: Diag.Freeze_Mode

FALSE (0)

TRUE (1)

Bit 5: Diag.Sync_Mode (Synchronisation Mode)

FALSE (0)

TRUE (1)

Bit 6: reserved

Bit 7: Diag.Deactivated

Shall be set to zero if Diagnosis-RES-PDU

Value (0): means FALSE if Get_Master_Diag-RES-PDU

Value (1): means TRUE if Get_Master_Diag-RES-PDU

5.3.3 Coding of the field Station_status_3

The coding of this field shall be according to 3.4 and the individual bits shall have the following meaning:

Bit 0 to 6: reserved

Bit 7: Diag.Ext_Diag_Overflow (Overflow of extended Diagnosis)

FALSE (0)

TRUE (1)

5.3.4 Coding of the field Diag_Master_Add

This field shall be coded as data type Unsigned8 with the following values:

Decimal (0-125)

means the address of the DP-master (Class 1) which has parameterized this DP-slave

Decimal (255)

means DP-slave has not been parameterized or has got invalid parameterisation.

Decimal (126-254)

not allowed

5.3.5 Coding of the field Ident_Number

This field shall be coded as data type Unsigned16.

5.3.6 Coding of the field Header_Octet

The coding of this field shall be according to 3.4 and the individual bits shall have the following meaning:

Bit 0 to 5: Block_Length

It shall contain the length of the diagnosis block according to the values of Table 6, Table 7, Table 8, and Table 9. The Header_Octet itself shall always be counted.

Table 6 – Block_Length for Selection:= 0

Value (decimal)	Meaning
2 to 63	Device Related Diagnosis in base diagnosis format
4 to 63	Device Related Diagnosis in DPV1 diagnosis format

Table 7 – Block_Length for Selection:= 1

Value (decimal)	Meaning
2 to 32	Identifier Related Diagnosis

NOTE The Identifier Related Diagnosis is limited to 32 because the identifiers itself are limited to 244.

Table 8 – Block_Length for Selection:= 2

Value (decimal)	Meaning
0 to 63	Content of the field Header_Octet.Block_Length shall be used as Identifier_Number See 5.3.15

NOTE One Channel Related Diagnosis entry contains three octets.

Table 9 – Block_Length for Selection:= 3

Value (decimal)	Meaning
1 to 63	Content of the field Header_Octet.Block_Length shall be used as Revision_Number See 5.3.18

NOTE One Revision_Number entry contains one octet.

Bit 6 to 7: Selection

The Selection shall contain the values according to Table 10.

Table 10 – Selection range

Value (decimal)	Meaning
0	Device Related Diagnosis
1	Identifier Related Diagnosis
2	Channel Related Diagnosis
3	Revision_Number

5.3.7 Coding of the field Alarm_Type

The coding of this field shall be according to 3.4 and the individual bits shall have the following meaning:

Bit 0 to 6: Alarm_Type

The Alarm_Type shall contain the values according to Table 11.