IO-Link

IODD

IO Device Description

Specification

related to

IO-Link Interface and System Specification V1.1.4

and

IODD Schemas V1.1

Version 1.1.4
June 2024

Order No: 10.012

IO-Link

File name: IO-Device-Desc-Spec_10012_V1.1.4_Jun24.docx

Prepared, approved and released by the IO-Link Community

This document covers all Change Requests within the IO-Link CR database up to ID 39.

Any comments, proposals, requests on this document are appreciated. Please use www.io-link-projects.com for your entries and provide name and email address.
Login: ***IO-Link-DD***
Password: ***Report***

Important notes:

NOTE 1 The IO-Link Community Rules shall be considered prior to the development and marketing of IO-Link products. The document can be downloaded from the www.io-link.com portal.

NOTE 2 Any IO-Link device shall provide an associated IODD file. Easy access to the file and potential updates shall be possible. It is the responsibility of the IO-Link device manufacturer to test the IODD file with the help of the IODD-Checker tool available per download from www.io-link.com.

NOTE 3 Any IO-Link devices shall provide an associated manufacturer declaration on the conformity of the device. A corresponding form with references to relevant documents is available per download from www.io-link.com.

Disclaimer:

The attention of adopters is directed to the possibility that compliance with or adoption of IO-Link Community specifications may require use of an invention covered by patent rights. The IO-Link Community shall not be responsible for identifying patents for which a license may be required by any IO-Link Community specification, or for conducting legal inquiries into the legal validity or scope of those patents that are brought to its attention. IO-Link Community specifications are prospective and advisory only. Prospective users are responsible for protecting themselves against liability for infringement of patents.

The information contained in this document is subject to change without notice. The material in this document details an IO-Link Community specification in accordance with the license and notices set forth on this page. This document does not represent a commitment to implement any portion of this specification in any company's products.

WHILE THE INFORMATION IN THIS PUBLICATION IS BELIEVED TO BE ACCURATE, THE IO-LINK COMMUNITY MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL INCLUDING, BUT NOT LIMITED TO ANY WARRANTY OF TITLE OR OWNERSHIP, IMPLIED WARRANTY OF MERCHANTABILITY OR WARRANTY OF FITNESS FOR PARTICULAR PURPOSE OR USE.

In no event shall the IO-Link Community be liable for errors contained herein or for indirect, incidental, special, consequential, reliance or cover damages, including loss of profits, revenue, data or use, incurred by any user or any third party. Compliance with this specification does not absolve manufacturers of IO-Link equipment, from the requirements of safety and regulatory agencies (TÜV, BIA, UL, CSA, etc.).

IO-Link ® is registered trade mark. The use is restricted for members of the IO-Link Community. More detailed terms for the use can be found in the IO-Link Community Rules on www.io-link.com.

Conventions:

In this specification the following key words (in **bold** text) will be used:

shall: indicates a mandatory requirement. Designers **shall** implement such mandatory requirements to ensure interoperability and to claim conformity with this specification.
should: indicates flexibility of choice with a strongly preferred implementation.
can: indicates flexibility of choice with no implied preference (possibility and capability).
may: indicates a permission.
highly recommended: indicates that a feature shall be implemented except for well-founded cases. Vendor shall document the deviation within the user manual and within the manufacturer declaration.

Publisher:
IO-Link Community
c/o PROFIBUS Nutzerorganisation e.V.
Ohiostrasse 8
76149 Karlsruhe
Germany

Phone: +49 721 / 98 61 97 0
Fax: +49 721 / 98 61 97 11
E-mail: info@io-link.com
Web site: www.io-link.com

CONTENTS

1 Introduction

An IODD (IO Device Description) is a set of files that formally describes a device e.g. IO-Link Device.

The IODD is created by the device vendor and shall be sufficient for IO-Link Tools to identify, communicate, parameterize and diagnose the device.

The set of files consists of the main IODD file, optional language files and optional picture files.

An IODD is mandatory for each IO-Link Device. This specification defines the IODD for IO-Link Devices that conform to the *IO-Link Interface and System* Specification Version 1.1.4 .

2 Related documents and references

2.1 References

The referenced documents listed in the Bibliography appendix, which are referenced in this specification are indispensable for the application 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.

2.2 Related documents

Related documents are listed in the Bibliography at the end of this document.

3 Definitions and abbreviations

3.1 Definitions

For the purposes of this document, the terms and definitions given in ISO 15745-1:2003 apply.

IO-Link Tool
Device engineering tool for the IO-Link Master and the connected IO-Link Devices. Used for parameterization and diagnosis of IO-Link Devices on the basis of the IODD.

3.2 Abbreviated terms

ANSI	American National Standards Institute (http://www.ansi.org/)
ASCII	American Standard Code for Information Interchange (see ANSI INCITS 4-1986 (R2007) and the US variant of ISO/IEC 646:1991)
BIPM	Bureau International des Poids et Mesures (http://www.bipm.org/)
C/Q	Connection for communication (C) or switching (Q) signal (SIO)
CRC	Cyclic Redundancy Check
DI	Digital Input
DO	Digital Output
I/Q	NC, DI or DO
IEC	International Electrotechnical Commision (http://www.iec.ch/)
IEEE	Institute of Electrical and Electronics Engineers (http://www.ieee.org/)
IETF	Internet Engineering Task Force (http://www.ietf.org/)
IO or I/O	Input / Output
IODD	IO Device Description
ISDU	Indexed Service Data Unit

ISO	International Standardization Organisation (http://www.iso.org/)
ITU	International Telecommunication Union (http://www.itu.int/)
LF	Line Feed
MSXML	Microsoft XML Core Services (see http://msdn.microsoft.com/en-us/library/ms763742%28VS.85%29.aspx)
NC	Not Connected
PLC	Programmable Logic Controller
PNG	Portable Network Graphics (see IETF RFC 2083 and ISO/IEC 15948:2004)
RFC	Request for Comments
SIO	Standard Input Output (digital switching mode)
UCS	Universal Multiple-Octet Coded Character Set (see The Unicode Standard or ISO/IEC 10646)
UL	Underwriters Laboratories (http://www.ul.com/)
UTC	Coordinated Universal Time (Temps Universel Coordonné) (coordinated by the BIPM) (corresponds to GMT = Greenwich Mean Time)
UTF	UCS Transformation Format (see The Unicode Standard or ISO/IEC 10646)
W3C	World Wide Web Consortium (http://www.w3.org/)
XML	Extensible Markup Language (see REC-xml-20081126)
XSD	XML Schema Definition (see REC-xmlschema-1-20041028, and REC-xmlschema-2-20041028)
XSL	Extensible Stylesheet Language
XSLT	XSL Transform (see REC-xslt-19991116)

4 Basic structure

The following figure shows the basic structure of the main IODD file. It follows the ISO 15745-1:2003 standard regarding the device profile and communication network profile. It consists of the elements DocumentInfo, ProfileHeader, ProfileBody, CommNetworkProfile, ExternalTextCollection and the Stamp.

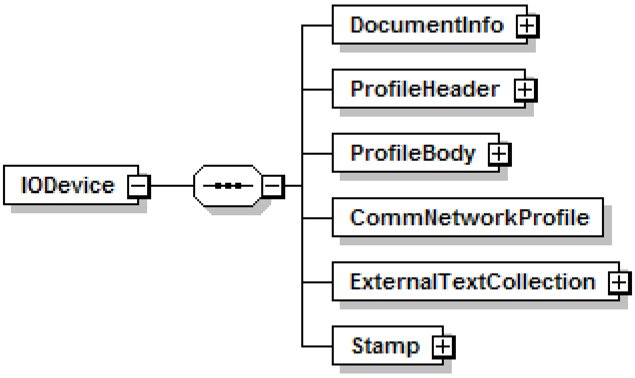


Figure 1 – Structure of main IODD file following ISO 15745-1

5 Files

Conceptionally, the IO Device Description consists of the set of files created by the device vendor, and the set of standard definition files which are part of this specification. IO-Link Tools combine information from both sets of files to get the complete device description.

All IODD XML files shall use “UTF-8” for the encoding. They shall use the namespace http://www.w3.org/2001/XMLSchema-instance with the prefix “xsi” and the namespace http://www.io-link.com/IODD/2010/10 with the prefix “iodd”. A schemaLocation for the namespace http://www.io-link.com/IODD/2010/10 to the required schema shall be given. For the main IODD file, this is IODD1.1.xsd, and for the language files this is IODD-Primitives1.1.xsd. The schema file name shall be given without any path prefix. No other namespaces shall be described. IODD XML files shall not use any DTD (Document Type Definition, see http://www.w3.org/TR/xml/#sec-prolog-dtd).

All XMLs generated by the vendor shall be checked by the IODD Checker software before delivery. This Checker is a tool available from the IO-Link web site (http://www.io-link.com/). It checks the content of the device description and if no errors were found writes a checksum over the file contents into the element Stamp at the end of the XML-file.

IO-Link Tools shall compare the checksum in the Stamp with the checksum calculated from the file contents. It is recommended to reject the IODD if there is a mismatch. Tools may then omit schema validation and additional checks.

IO-Link Tools shall use the file name of the IODD only to discover the language files that are associated with the main IODD file. Apart from that, tools shall not evaluate the file name; they always evaluate the file’s content. The device-specific file name is only intended for better legibility.

Adherence to the rules for file names makes it possible that all IODDs can be stored side-by-side in a single directory.

File names shall not only be different in upper and lower case. Case sensitivity of default parts of file names shall be adhered to.

The following special characters are permitted in vendor name and device name: _, #, -

All files of the set of files belonging to a specific IODD shall have the same <vendor name> part in their file names. The <vendor name> should be the same for all IODDs of the same vendor. It is not required that the <vendor name> in the file name matches exactly the content of the DeviceIdentity/@vendorName attribute or the standard variable V_VendorName in the device. Usually, the latter two also contain the legal form of the company, e.g. “Inc.”, “AG”, “S.A.” but this is not included in the vendor name part of the file name.

5.1 Main IODD file

The file name shall follow the following rule:

<vendor name>**-**<device name>**-**<release date>**-IODD**<schema version>**.xml**

e.g. VendorX-DeviceY-20110603-IODD1.1.xml

Contains information (in XML) about the identification of the device, communication characteristics, parameters, process data and diagnosis data.

The IODD shall always entirely contain texts in the PrimaryLanguage (English). The IODD may contain texts in further languages.

A style sheet for the vendor-specific description of Devices for a certain browser (optional):

e.g. VendorX-IODD1.1.xsl

The IODD shall not reference such style sheets with a processing instruction (<?xml-stylesheet … ?>).

5.2 Language files (optional)

To add support for additional languages after an IODD has been released, separate language files (in XML) may be created. Their file name shall exactly match the name of the main IODD file, except that there is an additional language designation before the file name extension:

<vendor name>**-**<device name>**-**<release date>**-IODD**<schema version>**-**<language>**.xml**

The “language” part follows ISO 639:2023. The “language” part shall correspond to the value of the ‘xml:lang’ attribute inside the language file. There shall be no additional language file for languages already covered in the main IODD file. The “language” part consists of two letters.

The ‘Text’ and ‘TextRedefine’ elements contained in the additional language file shall follow the same rules as specified for the respective elements in additional languages inside the main IODD.

e.g. VendorX-DeviceY-20110603-IODD1.1-ru.xml

Additional language file containing texts in Russian.

An IO-Link Tool shall select the appropriate language from the main IODD file or the accompanying language files according to its user interface language settings. A tool shall ignore files whose filename does not match to the naming convention of additional IODD files.

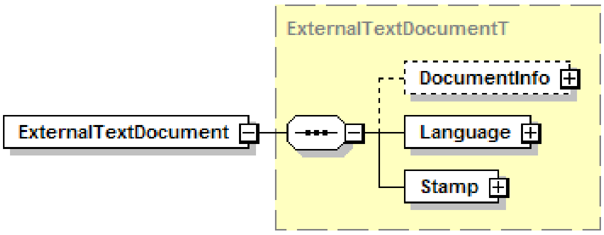


Figure 2 – Structure of language file

5.3 Image files (optional)

The file format shall be PNG (file extension .png, see IETF RFC 2083 or ISO/IEC 15948:2004). The same rules for permitted characters apply as in section ‘Files’ (see above).

<vendor name>**-logo.png**
Vendor logo. 160 x 90 pixel, landscape format. The background of the logo should be transparent.

<vendor name>**-**<picture name>**-icon.png**
Device variant icon. 48 x 48 pixel.

<vendor name>**-**<picture name>**-pic.png**
Device variant picture. Min. 160 x 160 pixel, max. 320 x 320, square.

<vendor name>**-**<picture name>**-con-pic.png**
Device variant connection picture. Min. 160 x 160 pixel, max. 320 x 320, square.

The device variant icons and device variant pictures are referenced from the DeviceIdentity/ DeviceVariantCollection/DeviceVariant elements. The device variant connection pictures are referenced from the CommNetworkProfile/TransportLayers/PhysicalLayer/Connection elements. The referenced image files shall accompany the main IODD file for stamping and delivery.

5.4 Standard definitions files

IODD-StandardDefinitions1.1.xml
This file contains the definition of standardized variables, error types and events (see *IO-Link Interface and System* Specification Version 1.1.4) plus English language texts.

IODD-StandardDefinitions1.1-de.xml, *-es.xml, *-fr.xml, *-it.xml, *-ja.xml, *-ko.xml, *-pt.xml, *-ru.xml, *-zh.xml.
Additional language file containing texts in German, Spanish, French, Italian, Japanese, Korean, Portuguese, Russian, Chinese.

IODD-StandardUnitDefinitions1.1.xml
This file contains the definitions of all available unit codes plus English language texts.

Those files are part of the standard and shall not be changed. Vendors of IO-Link Tools should use those files instead of hard-coding standardized things.

Additional language files for standard definitions files will be provided by the IODD subteam when needed on the IO-Link website.

5.5 Schema files

Schema files are needed to validate the structure of XML-files and to aid in editing.

IODD1.1.xsd
IODD-schema; includes the following sub-schemas:

IODD-Primitives1.1.xsd
includes basic schema elements

IODD-Datatypes1.1.xsd
includes schema elements for the definition of data types

IODD-Events1.1.xsd
includes schema elements for the definition of error types and events

IODD-Variables1.1.xsd
includes schema elements for the definition of variables

IODD-UserInterface1.1.xsd
includes schema elements for the definition of the user interface

IODD-Communication1.1.xsd
includes schema elements for the definition of the communication network profile

IODD-StandardDefinitions1.1.xsd
schema for the definition of system-specific elements used to validate the file IODD-StandardDefinitions1.1.xml and IODD-StandardUnitDefinitions1.1.xml

IODD-WirelessCommunication1.1.xsd
includes schema elements for the definition of the wireless communication network profile

6 Description mechanisms

6.1 Names of elements and attributes

Following one common pattern, the names of the elements begin with an uppercase letter while the names of the attributes begin with a lowercase letter. When names consist of several words, each word (except for the first in case of an attribute) starts with an uppercase letter. No separator character (like _) is used.

6.2 Ids

The values of the attribute ‘id’ shall follow the regular expression pattern:

“[A-Za-z][A-Za-z0-9 _-]*[A-Za-z0-9]”.

Ids shall be unique within the elements of the same type. The prefix “STD_” is reserved for ids in the standard definition files and shall not be used for elements in the main IODD.

6.3 Referencing

Each element that can be referenced within the IODD contains an explicit attribute ‘id’. The referencing element contains a type-dependent attribute with the following composition: <type>Id

Examples: textId, datatypeId, menuId, variableId

6.4 Text localization

All text components of the different languages which are referenced in the IODD are allocated in the ExternalTextCollection (for further information see “Language-Dependent Description Texts”).

The text components of the different languages are referenced in the relevant location according to a key (textId).

Further languages can be added in an appropriate file (see chapter 5.2).

The PrimaryLanguage in the IODD shall be completely available. If there is a further language added in the IODD or in a separate language file, not all entries shall be given. In this case, the interpreter has to go back to the entry of the PrimaryLanguage.

7 Device Description

For IO-Link Engineering Tools no conformance classes are specified. IO-Link EngineeringTools shall support the *IO-Link Interface and System* Specification Version 1.1.4 completely. For interpretation of this IO Device Description Specification 1.1.4 the following requirements shall be fulfilled.

An IO-Link Engineering Tool shall support:

- All IO-Link devices
- If the tool provides a catalog, at least one catalog entry for each IODD (derived from DeviceName)
- All data types
- All Variables incl. StandardVariables
- Menus, including Buttons
- ErrorTypes incl. StandardErrorTypes
- Conditions
- User Role "Specialist"
- Texts in the PrimaryLanguage (English)
- Refresh of dynamic variable values (on demand or cyclic)
- When variables marked with attribute modifiesOtherVariables set to “true” are changed, the tool shall either notify the user that other variables are possibly changed, or the tool shall automatically reload.
- When parameters are up- or downloaded, any ErrorTypes that occur shall be displayed in a way that the user is able to associate them with the parameter that triggered it, either by displaying it at the parameter or by telling the parameter name in the error log.
- the complete list of error types contained in the standard definition file.
- the complete list of event codes contained in the standard definition file.
- Variables with accessRights = “wo” shall never be part of any download sequence

- Variables with accessRights = “wo” shall always be handled as a single write request
- The Button assigned text referenced in IODD /Button/Description element to variable StdVariableRef[@id=’V_SystemCommand’]/StdSingleValueRef shall be shown in a message box with OK and Cancel, see chapter 7.5.8.4 Description.
  The affected System Commands are
  - DeviceReset (128)
  - ApplicationReset (129)
  - RestoreToFactorySettings (130)
  - BackToBox (131)
  Those commands influence a vast set of parameters within the device, so the user can abort the action.
- if the Button references a SystemCommand or another write-only variable which is defined as SingleValue, tools ought to apply ActionStartedMessage, see chapter 7.5.8.4 ActionStartedMessage

An IO-Link Tool should support, but is not obliged to:

- Separate catalog entries for each DeviceVariant
- A separate display of the Process Data (in addition to V_ProcessDataIn / V_ProcessDataOut)
- Display of the Connection Description
- Texts in other Languages than the PrimaryLanguage
- User Roles other than "Specialist"
- Vendor logo, pictures and icons.
- For menu entries having gradient and/or offset: additional display of the raw value
- Names of ValueRanges
- Events incl. StandardEvents
- ErrorTypes and Events: additional display of the raw values

IO-Link Tools distinguish devices by their VendorID and DeviceID, and the IODDs of a device by the IODD version they are based on and their DocumentInfo/@releaseDate. Tools shall prefer V1.0.1 IODDs over V1.0 IODDs, and within the IODD version newer IODDs over older IODDs. For V1.1 IO-Link devices there shall be only a single current (most recent) IODD based on V1.1, and if the device is compatible to IO-Link V1.0, additionally a single current IODD based on V1.0.1 or V1.0.

7.1 Notation of XML structure

The XML structure is hierarchical. As the whole structure is too complex to show in one figure, the description is split into a series of figures, starting with the root element and descending into the details.

Following each figure showing the structure of a particular section of the IODD, all the elements and their attributes are listed in the order in which they appear inside the figure.

The description of elements and attributes follows this pattern:

> **Element_or_attribute_name (Use[, XML_type])**
> Semantics of the element or attribute. If the element or attribute has a value, a possible default or fixed value is also described here.

Element_or_attribute_name is the name of the element or attribute. Remember that element names start with an uppercase letter while attribute names start with a lowercase letter.

Use is one of the following letters:

m Mandatory
o Optional
c Conditional (depends on, see description):
Schema is not powerful enough to formulate the complex IODD rules, therefore business logic has to be checked by IODD Checker, see Annex C

XML_type is the XML schema data type of the element or attribute value (if applicable). Do not confuse this with the data types that the device’s variables and process data may use. XML_type may be

- one of the basic XML types defined in REC-xmlschema-2-20041028. The namespace prefix “xsd:” is omitted for brevity.

- one of the IODD XML types defined in IODD-Primitives1.1.xsd (see Table 1).

Table 1 – IODD XML types

XML_type	**Defined as**	**Use**
IdT	xsd:string with pattern: "[A-Za-z][A-Za-z0-9 _-]*[A-Za-z0-9]"	Used for an ‘id’ attribute at an element so it can be referenced.
RefT	xsd:string with pattern: "[A-Za-z][A-Za-z0-9 _-]*[A-Za-z0-9]" (same as IdT)	Used as a reference to some element that has an ‘id’ attribute.
SubindexT	xsd:unsignedByte restricted to 1..255 (0x01..0xFF)	For sub-adressing within an index.
BitCountT	xsd:unsignedShort	For bit offsets and bit lengths.
IsduLengthT	xsd:unsignedByte restricted to 1..232	For lengths (in octets) which shall fit into an ISDU
DeviceIdT	xsd:unsignedInt restricted to 1..16777215 (0x000001..0xFFFFFF)	For a device ID.
CharacterEncodingT	xsd:string, either "UTF-8" or "US-ASCII"	The character encoding of a string.
VersionT	xsd:string with pattern: "V\d+(\.\d+){1,7}"	To express a version of e.g. the IO-Link specification, the IODD Checker, the IODD instance.
AccessRightsT	xsd:string, either "ro", "rw" or "wo"	Access rights read only, read-write or write-only.

Further restrictions to these XML types are mentioned directly at the XML type or in the element / attribute description.

7.2 Basic structure of the main IODD file

Figure 3 shows the basic structure of a device in a device description.

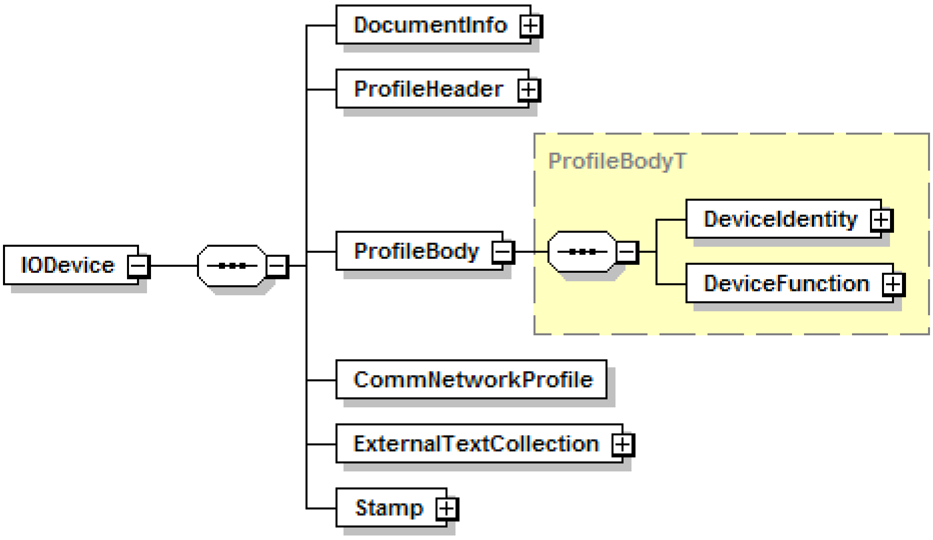


Figure 3 – Basic structure of main IODD file

7.3 Metainformation

7.3.1 DocumentInfo (m; o for language file)

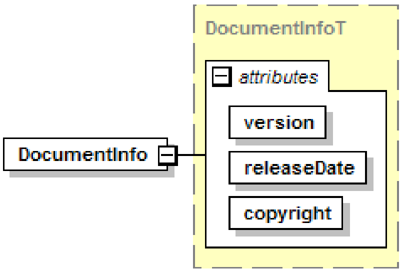


Figure 4 – DocumentInfo element

Here the vendor inserts the information for the IODD.

version (m, VersionT)
The 'version' attribute contains the version of the concrete instance and not the version of the IODD specification. The vendor shall increase this version for each official release of the IODD for a particular device.

releaseDate (m, date with pattern "\d{4}-\d{2}-\d{2}")
The date information in the IODD file name shall correspond to the 'releaseDate' attribute in the DocumentInfo element. The releaseDate attribute of a language file shall correspond to the releaseDate attribute oft the main IODD. There shall be no more than one official release of the IODD for a particular device per day. IO-Link Tools shall rely on this date for determining the newest version of the IODD for a device.

copyright (m, string)
Vendor-specific copyright text.

e.g.
File name: IO-Link-SampleDevice-20200801-IODD1.1.xml

DocumentInfo:

```
<DocumentInfo version="V5.17" releaseDate="2020-08-01" copyright="IO-Link Community"/>
```

7.3.2 ProfileHeader (m)

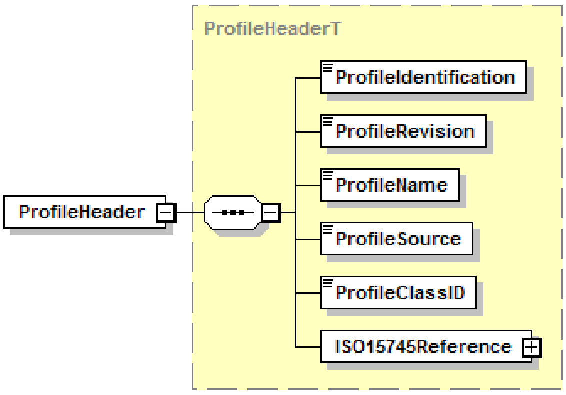


Figure 5 – ProfileHeader element

Within this element, the vendor shall give the following constant information in plain text.

ProfileIdentification (m, string)
Fixed to "IO Device Profile"

ProfileRevision (m, string)
Fixed to "1.1".

ProfileName (m, string)
Fixed to "Device Profile for IO Devices".

ProfileSource (m, string)
Fixed to "IO-Link Consortium"

ProfileClassID (m, NMTOKEN)
Fixed to "Device".

ISO15745Reference (m)
Information about the underlying ISO standard

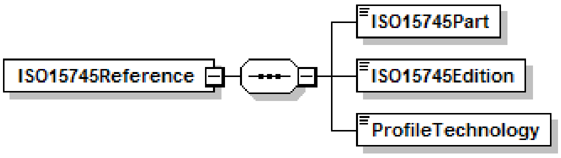


Figure 6 – ISO15745Reference element

ISO15745Part (m, positiveInteger)
Fixed to "1".

ISO15745Edition (m, positiveInteger)
Fixed to "1".

ProfileTechnology (m, string)
Fixed to "IODD".

7.3.3 ProfileBody (m)

The ProfileBody contains the description of identity and functionality of the device.

7.3.4 File validation

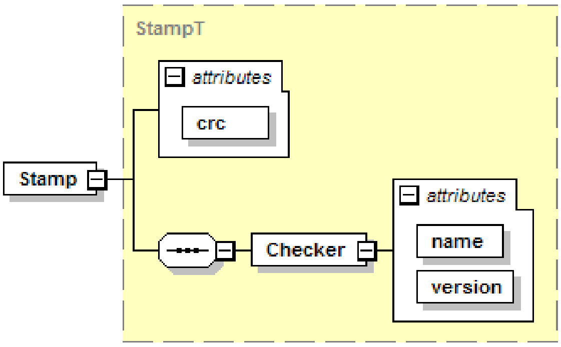


Figure 7 – Stamp element

There is a tool called “IODD Checker” that intensely tests the IODD if it is conformant to this specification. The Checker features a *check* and a *stamp* mode. In *check* mode, errors detected during the checking process are reported, but the file remains unchanged. In *stamp* mode, the Stamp element is always rewritten.

crc (m, unsignedInt)
If no errors are detected during the checking process, the ‘crc’ attribute is set to the CRC value calculated from the file contents. Otherwise, the ‘crc’ attribute is set to an invalid value. By checking the CRC, an IO-Link Tool can find out whether the IODD has been altered since the last successful check. In this case, the IODD should be rejected by the tool.

For the CRC, the CRC-32 algorithm is used (see section 8.1.1.6.2 of ITU-T recommendation V.42 (03/2002) or ISO/IEC 13239:2002). Before the actual calculation, the ‘crc’ attribute is set to an empty string and the checker inserts its name and version into the appropriate attributes. The generated CRC is then inserted into the ‘crc’ attribute.

The CRC calculation is done as follows: The IODD file is read in binary mode. The stream of bytes is fed into the CRC algorithm until the string `<Stamp crc="` has been processed. The value of the attribute crc is skipped, and the CRC calculation continues with the closing quotation mark.

The same is done with external language documents, but after the end-of-file has been reached, the CRC of the main IODD file is converted to decimal representation (no leading zeroes) and the character codes for the digits are fed into the CRC algorithm.

Checker (m)
Identification of the IODD Checker version used to check and stamp this file. If there is a severe bug in a specific Checker version, or the method of calculating the CRC shall be modified in the future, IO-Link Tools are able to adapt to this based on the Checker name and version.

name (m, string)
The name of the IODD Checker.

version (m, VersionT)
The version of the IODD Checker.

When writing a new IODD, before applying the IODD checker on it for the first time, it is recommended to set the attributes to the following values:

Stamp/@crc = "0"
Stamp/Checker/@name = "" (empty string)
Stamp/Checker/@version = "V0.0.0.0"

It is highly recommended, not to insert comments in or after the Stamp element.

7.4 Device identity

On import of a new IODD, IO-Link Tools shall use the pair vendorId and deviceId to decide wether this IODD describes a new device (catalog entries shall be added) or this IODD is a new description of an already known device (catalog entries shall be updated). This decision shall not be based on the filename of the IODD.

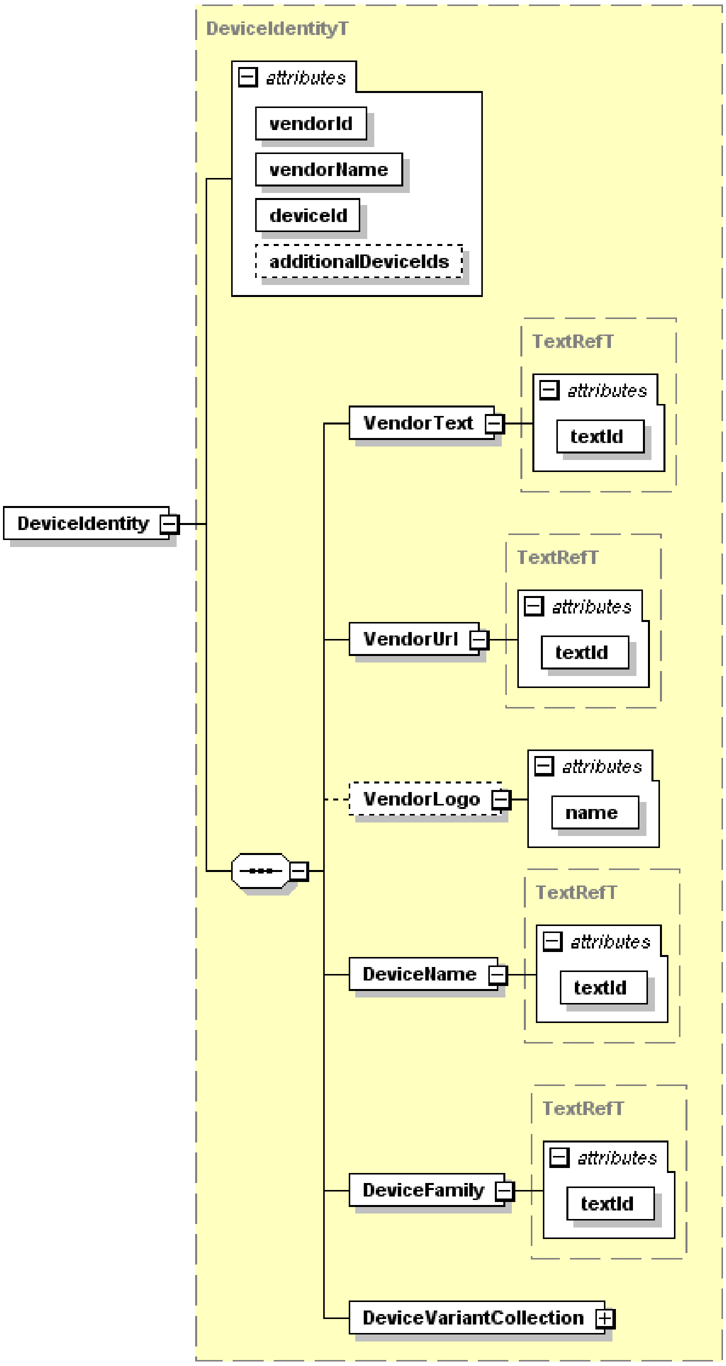


Figure 8 – DeviceIdentity element

vendorId (m, unsignedShort)
Unique identification of the vendor, assigned by the IO-Link Community. An IO-Link Tool shall display this id in decimal notation. The vendorId shall match the content of V_DirectParameters_1, subindex 8-9. The allowed value range is 1..65535 (0x0001..0xFFFF).

vendorName (m, string)
Name of the vendor of the device.

deviceId (m, DeviceIdT)
Vendor-internal unique identification of the device; an IO-Link Tool shall display this id in decimal notation. The deviceId shall match the content of V_DirectParameters_1, subindex 10-12.

additionalDeviceIds (o, list of DeviceIdT with min. one and max. 255 entries)
A list of device IDs which are supported by this device. In other words, this device may replace devices of the same vendor whose device IDs are listed in this attribute. The items of this list shall be separated by a space character.

VendorText (m)
textId (m, RefT)
A text describing the vendor (a slogan).

VendorUrl (m)
textId (m, RefT)
The vendor's URL.

VendorLogo (o)
name (m, string with pattern "([\p{L}\d_#]+-)+logo\.png")
File name of the vendor's logo; in PNG format, 160 x 90 pixels. If the element 'VendorLogo' is used, the image file referenced by the attribute 'name' shall be present.

DeviceName (m)
textId (m, RefT)
Common name for all variants. IO-Link Tools may use this to group the device variants of a device in the device catalog.

DeviceFamily (m)
textId (m, RefT)
Vendor-specific classification of the devices. IO-Link Tools may use this for grouping devices in the device catalog.

7.4.1 Device variant collection

Using the Vendor ID and Device ID read out from an unknown IO-Link Device, it shall be possible to uniquely find the latest version of the appropriate IODD. All variants referenced in the Device variant collection share the same data model and menu representation. They may differ only in the elements Variant and Connection. Thus is it not allowed that IO-Link Devices that differ in details described in ProfileBody/DeviceFunction or CommNetworkProfile except for element Connection share the same combination of Vendor ID and Device ID.

The things in which the devices may differ are those that are not "seen" by the IO-Link Tool, like:

- type of plug and length of cable
- materials: plastics, stainless steel
- shape: round, ..
- fastening: through-hole, bracket
- allowed environmental conditions: temperature range, humidity, shock resistance
- certificates: CE, UL

Devices that only differ in these things may use the same Vendor ID and Device ID and shall be described as different Device Variants in a single IODD. If the device supports ISDU Index 19 (V_ProductID), the value read from this ISDU shall match exactly to the 'productId' attribute of exactly one DeviceVariant.

Examples for things in which devices may **not** differ:

- measurement ranges (with sensors)
- power range (with actuators)

There shall be at least one device variant.

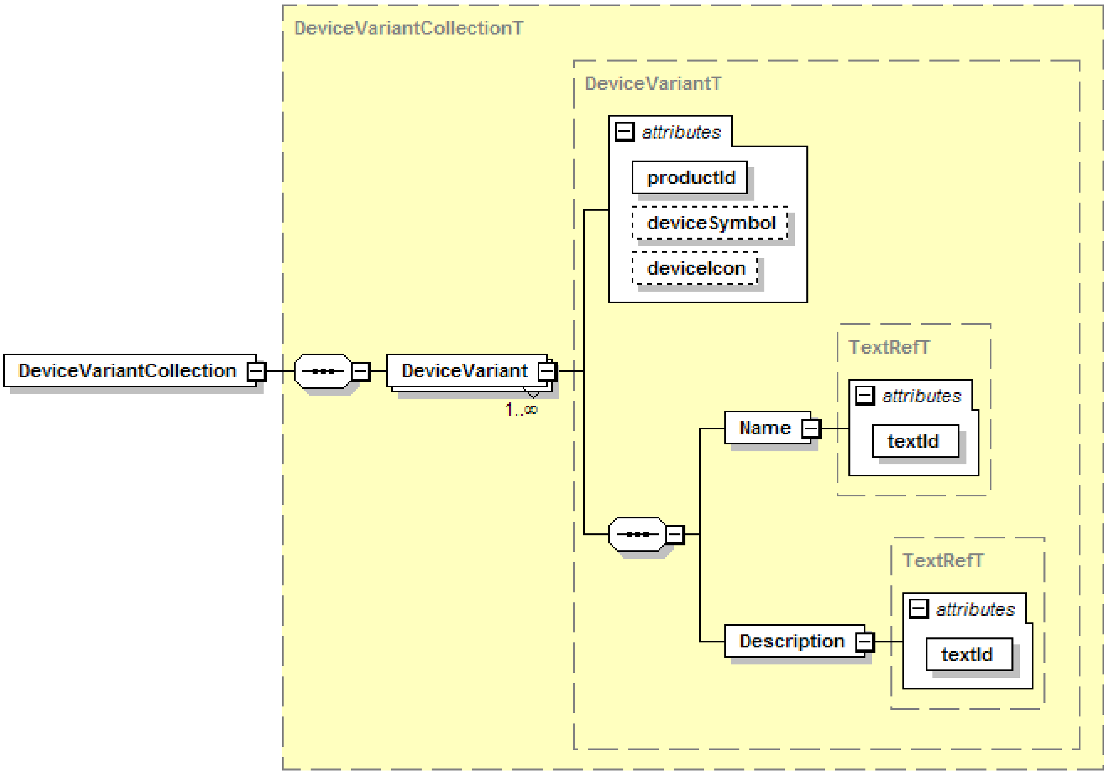


Figure 9 – DeviceVariantCollection element

productId (m, string)
Uniquely identifies the product within the DeviceVariants. 'productId' in IODD corresponds to the ISDU standard parameter V_ProductID. If V_ProductID is not implemented in the device only one single device variant shall be referenced in the IODD.

deviceSymbol (o, string with pattern "([\p{L}\d_#]+-)+pic\.png")
File name of the device symbol. If this attribute is used, the referenced image file shall be present.

deviceIcon (o, string with pattern "([\p{L}\d_#]+-)+icon\.png")
File name of the device icon. If this attribute is used, the referenced image file shall be present.

Name (m)
textId (m, RefT)
Used to build the catalog entries for the device variants in the IO-Link Tool. Shall be unique for each DeviceVariant within each supported language.

It shall correspond to the product name in the vendor's catalogue or to the name which is labelled on the product.

Description (m)
textId (m, RefT)
Descriptive text of the device.

7.5 Device function

The entire functionality of the device is collected here. Parameters, process data, data types, error codes and events are defined. Their significances, addresses, and data fields are identified as well as a grouping of the views in menus is defined.

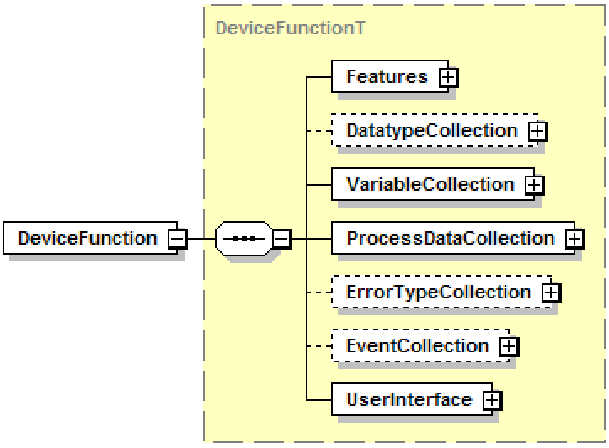


Figure 10 – DeviceFunction element

7.5.1 Features

Supported standardized features of the device are described.

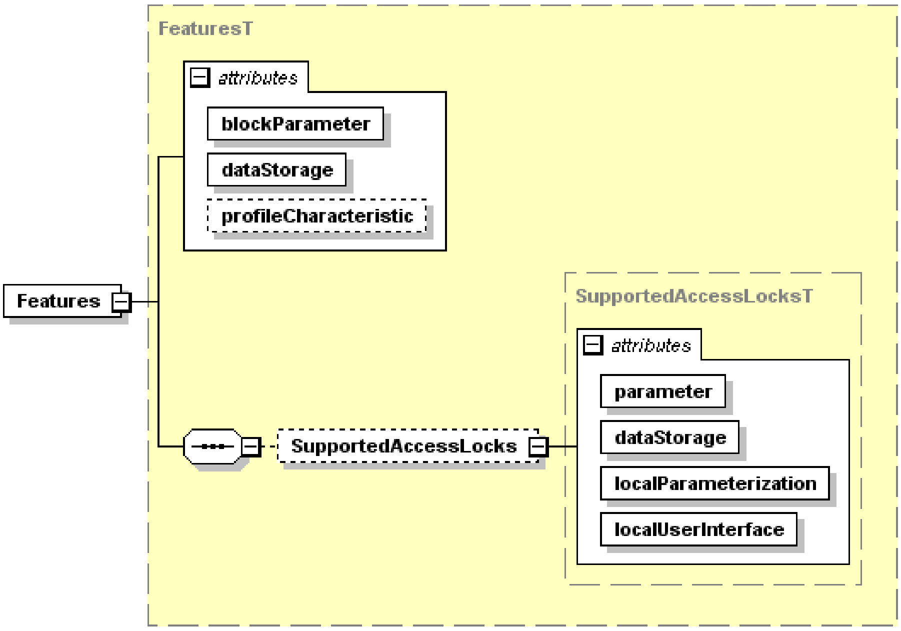


Figure 11 – Features element

blockParameter (m, boolean)
This attribute defines if a device supports the functionality of Block Parameter transmission. See *IO-Link Interface and System* Specification Version 1.1.4, chapter 10.3.5.

IO-Link Tools shall use Block Parameter transmission if the device supports it and if more than one variable is to be transferred due to a single user action.

Typical Block Parameter sequences:

Upload

- Send SystemCommand ParamUploadStart
- Upload all variables of the current user role
- Send SystemCommand ParamUploadEnd

Download

- Send SystemCommand ParamDownloadStart
- Download the changed (or all) variables of the current user role
- Send SystemCommand ParamDownloadEnd or ParamDownloadStore

dataStorage (m, boolean)
This attribute defines if a device supports data storage functionality.

If any standard ISDU variable with accessRights = “rw” is present, this attribute shall not be set to “false”.

If any device specific ISDU variable with accessRights = “rw” is present, it is higly recommended that this attribute is not set to “false”.

profileCharacteristic (o, list of unsignedShort with min. one and max. 32 entries)
A list of Profile Identifiers (PID) which are supported by this device. This list decribes the supported profiles and function classes. This attribute shall be given if any IO-Link Profile is supported.

Example: profileCharacteristic="10 49 16384"

How profiles are represented in the IODD is determined in the respective profile specification. In addition, there are XML snippets that formally describe the respective IODD content and also contain the instructions for checking the IODD. The definitions and rules are described in Annex D.

SupportedAccessLocks (c)
Shall be present if the standard variable V_DeviceAccessLocks is supported.

parameter (m, boolean)
Whether parameter access lock is supported. The assigned variable V_DeviceAccessLocks.ParameterAccessLock shall not be referenced in IODD menu.

According *IO-Link Interface and System* Specification Version 1.1.4, it is highly recommended, that parameter lock functionality shall not be implemented.

dataStorage (m, boolean)
Whether data storage access lock is supported. The assigned variable V_DeviceAccessLocks.DataStorageLock shall not be referenced in IODD menu.

According *IO-Link Interface and System* Specification Version 1.1.4, it is highly recommended, that dataStorage lock functionality shall not be implemented.

localParameterization (m, boolean)
Whether local parameterization access lock is supported.

localUserInterface (m, boolean)
Whether local user interface access lock is supported.

7.5.2 Data type collection

The DatatypeCollection incorporates all declarations for the reuse of data types (especially useful for records). There shall be no unreferenced Datatype elements. Standardized data types are described in the schema IODD-Datatypes1.1.xsd.

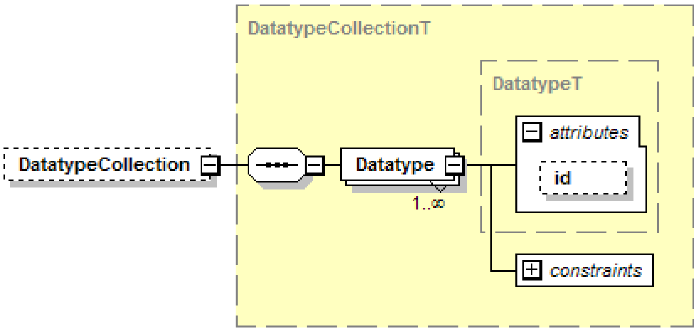


Figure 12 – DatatypeCollection element

For the Datatype element, this figure only shows the elements and attributes common to all data types. The actual selected data type needs additional elements and attributes. See chapter 15 for details.

id (c, IdT)
Datatype elements within the DatatypeCollection shall have an attribute ‘id’. Datatype elements outside of the DatatypeCollection shall not have an attribute ‘id’.

7.5.3 Data types

The IODD-Datatypes1.1.xsd schema provides derived types for all possible data types. The presence and type of required elements and attributes is checked by this schema.

Actually, the data types form the following hierarchy:

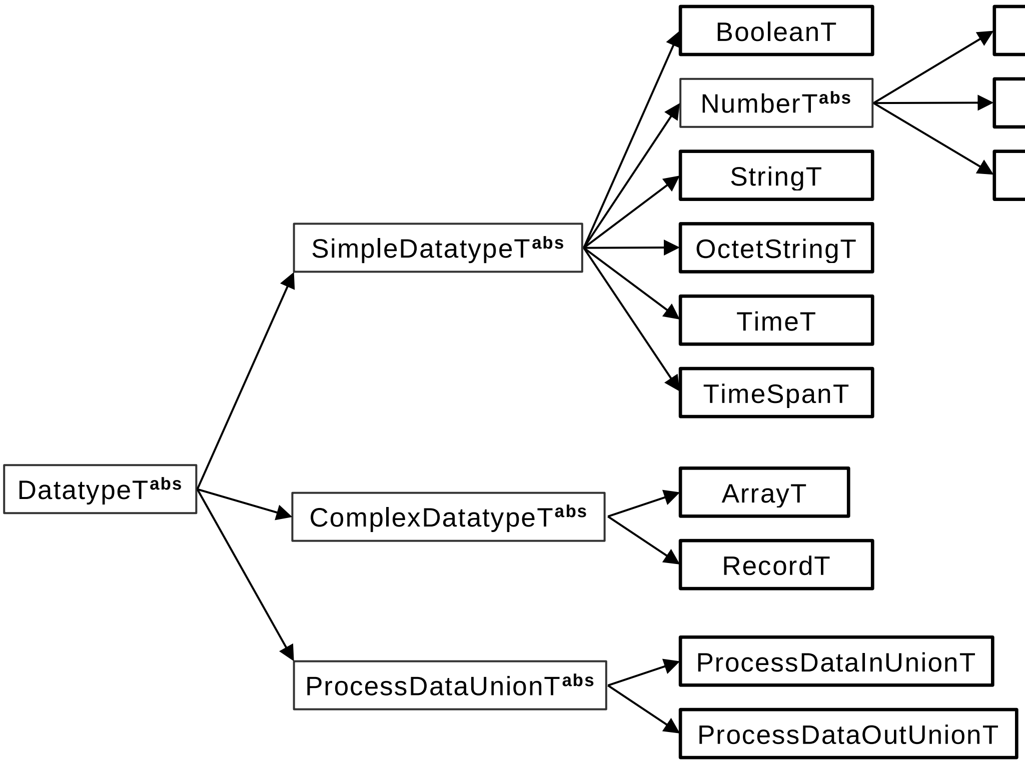


Figure 13 – Data type hierarchy

Each derivation adds elements and/or attributes appropriately.

Excursion on XML schema *abstract types*:
An abstract type can’t be used itself. Only non-abstract types which are derived from an abstract type can be used. The instance selects the desired derived type with xsi:type=”*name of the derived type*”.

This technique is used here with the ‘Datatype’ element to adapt the XML structure to the requirements of the specific data type.

For the serialization of the data types see *IO-Link Interface and System* Specification Version 1.1.4, Annex F.

7.5.3.1 Simple data types

The coding of simple data types is shown only for singular use which is characterized by

- Process data consisting of one simple data type
- Parameter consisting of one simple data type
- Subindex (>0) access on individual data items of parameters of complex data types (arrays, records)

7.5.3.1.1 General

When the Datatype element appears inside the DatatypeCollection, the attribute ‘id’ shall be present. Otherwise, the attribute ‘id’ shall not be present.

SingleValue and ValueRange elements are strongly typed.

Where SingleValue and / or ValueRange elements are permitted, the following rules shall be considered:

- When neither SingleValue nor ValueRange elements are given, the complete value range of the data type is allowed. When SingleValue(s) and / or ValueRange(s) are given, only these values are allowed.
- In ValueRanges, both the lowerValue and the upperValue are included in the range of allowed values.
- In ValueRanges, the lowerValue shall be less than the upperValue (not equal).
- SingleValues and ValueRanges shall not overlap.

7.5.3.1.2 BooleanT

Figure 14 shows the IODD representation of the data type BooleanT.

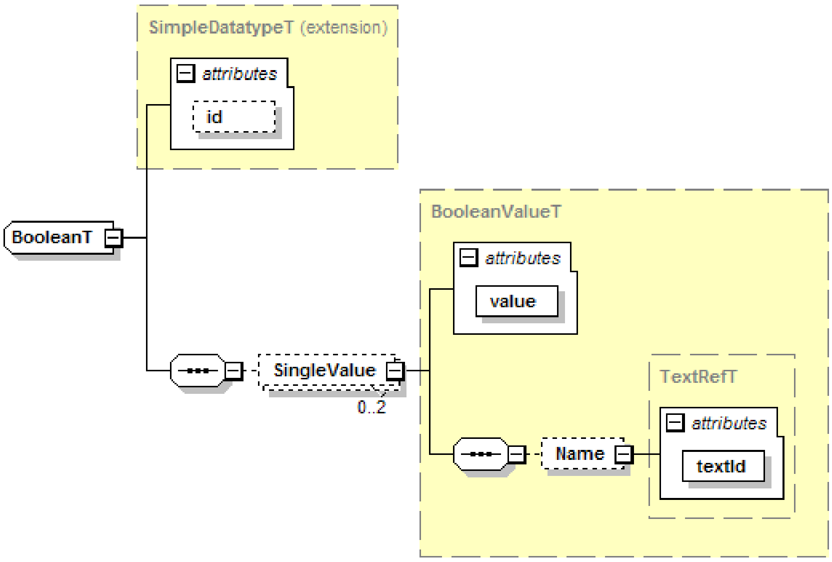


Figure 14 – BooleanT

Lexical representation:

Conforms to the representation of “boolean” in XML Schema, see http://www.w3.org/TR/xmlschema-2/#boolean

Regular expression pattern: “true|false|1|0”

Example:

```
<Datatype xsi:type=”BooleanT”>
        <SingleValue value=”false”>
                <Name textId=”TN_Inversion_Off”/>
        </SingleValue>
        <SingleValue value=”true”>
                <Name textId=”TN_Inversion_On”/>
        </SingleValue>
</Datatype>
```

7.5.3.1.3 UIntegerT

Figure 15 shows the IODD representation of the data type UIntegerT.

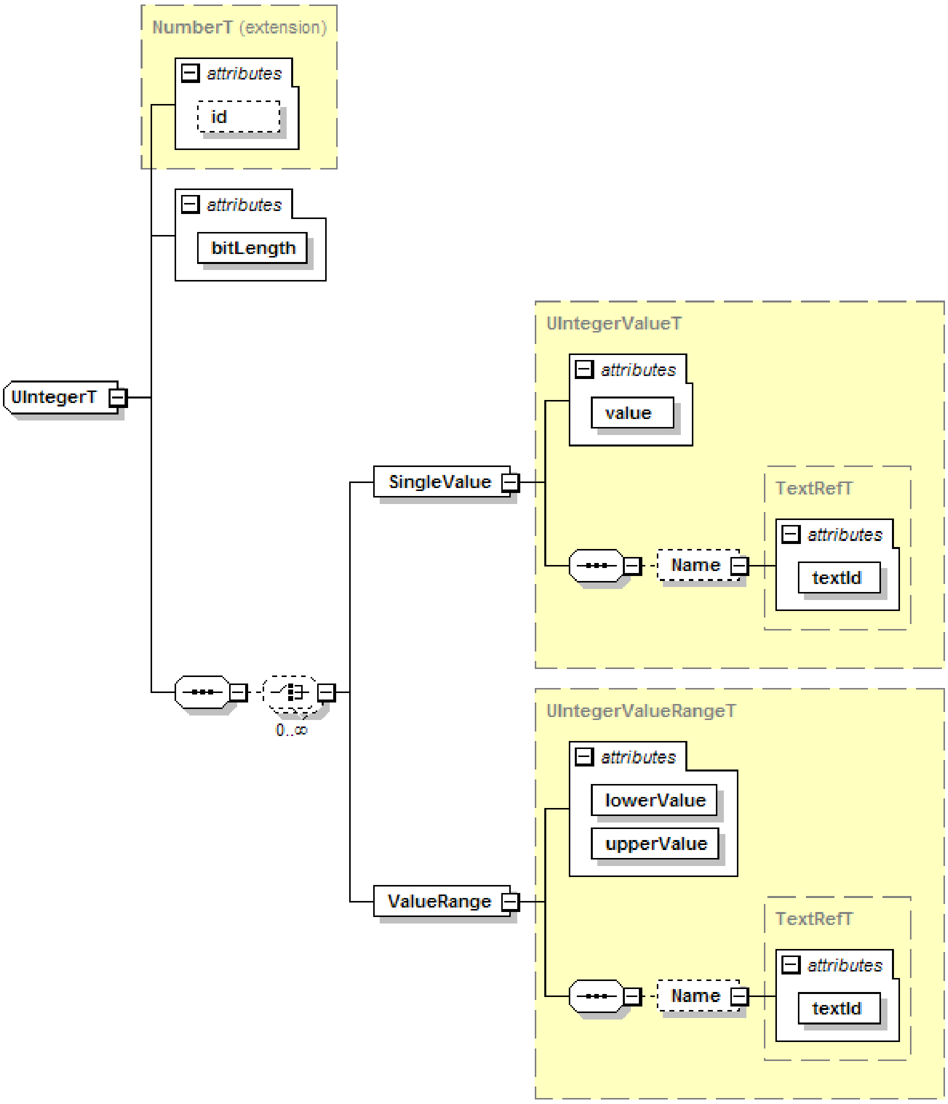


Figure 15 – UIntegerT

bitLength (m, BitCountT)
Specifies the size of the unsigned integer in bits. The allowed value range is 2..64.

Lexical representation:

Conforms to the representation of “unsignedLong” in XML Schema, see http://www.w3.org/TR/xmlschema-2/#unsignedLong

Regular expression pattern: “+?\d+”

Example:

```
<Datatype xsi:type=”UIntegerT” bitLength=”8”>
        <SingleValue value=”96”>
                <Name textId=”TN_System”/>
        </SingleValue>
</Datatype>
```

7.5.3.1.4 IntegerT

For the representation of the data type IntegerT in the IODD and an example see chapter 7.5.3.1.3.

Lexical representation:

Conforms to the representation of “long” in XML Schema, see http://www.w3.org/TR/xmlschema-2/#long

Regular expression pattern: “[+-]?\d+”

7.5.3.1.5 Float32T

Figure 16 shows the IODD representation of the data type Float32T.

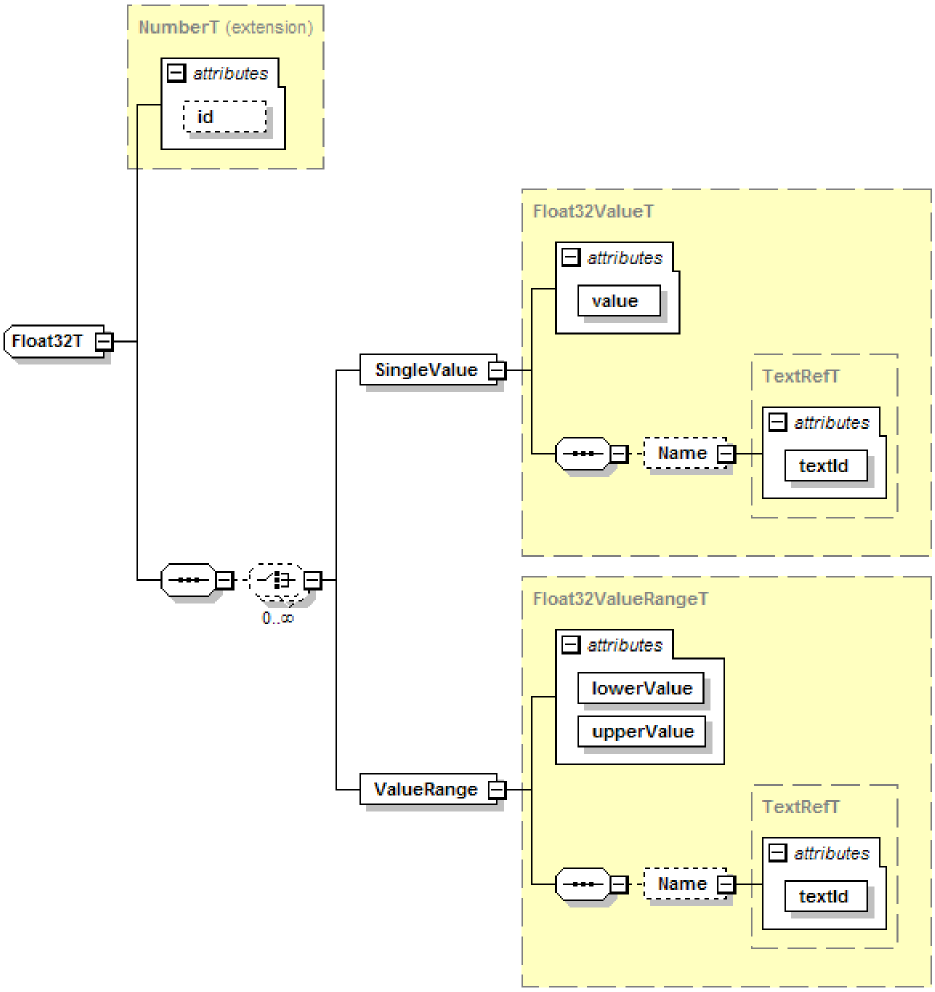


Figure 16 – Float32T

Lexical representation:

Conforms to the representation of “float” in XML Schema, see http://www.w3.org/TR/xmlschema-2/#float

Regular expression pattern: “[+-]?\d+(\.\d+)?([eE][+-]?\d+)?|-?INF”

Example:

```
<Datatype xsi:type=”Float32T”>
        <SingleValue value=”0.0”>
                <Name textId=”TN_Zero”/>
        </SingleValue>
        <ValueRange lowerValue=”1.0” upperValue=“1000.0“>
                <Name textId=”TN_Valid”/>
        </SingleValue>
</Datatype>
```

7.5.3.1.6 StringT

Figure 17 shows the IODD representation of the data type StringT.

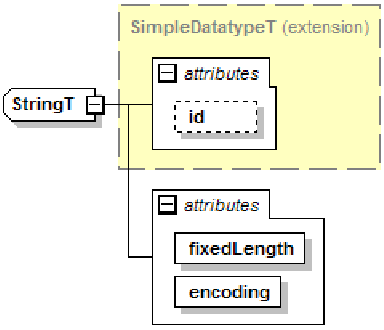


Figure 17 – StringT

fixedLength (m, IsduLengthT)
Specifies the length of the string in octets.

encoding (m, CharacterEncodingT)
The character encoding of the string. Note that US-ASCII consists of 7-bit characters only. Note that string constants in UTF-8 may need more than one octet per character.

Lexical representation:

Conforms to the representation of “string” in XML Schema, see http://www.w3.org/TR/xmlschema-2/#string

Regular expression pattern: “.*” (No restriction, just the string.)

Special characters shall be coded according to the XML syntax. See REC-xml-20081126, chapter 2.4 Character Data and Markup.

& → &amp;

‘ → &apos; (only required if inside a string enclosed in ‘ characters)

> → &gt;

< → &lt;

“ → &quot; (only required if inside a string enclosed in “ characters)

Example:

```
<Datatype xsi:type=”StringT” fixedLength=”64” encoding=”UTF-8”/>
```

7.5.3.1.7 OctetStringT

Figure 18 shows the IODD representation of the data type OctetStringT.

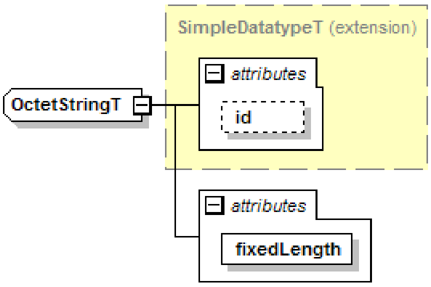


Figure 18 – OctetStringT

fixedLength (m, IsduLengthT)
Specifies the length of the octet string in octets.

Lexical representation:

Regular expression pattern: “(0x[0-9A-Fa-f][0-9A-Fa-f],)*0x[0-9A-Fa-f][0-9A-Fa-f]”

Example:

```
<Datatype xsi:type=”OctetStringT” fixedLength=”10”/>
```

7.5.3.1.8 TimeT

Figure 19 shows the IODD representation of the data type TimeT.

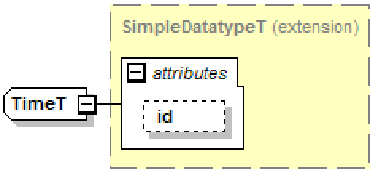


Figure 19 – TimeT

Lexical representation:

Follows the representation of “dateTime” in XML Schema, see http://www.w3.org/TR/xmlschema-2/#dateTime, but is stricter:

Regular expression pattern: “\d{4}\-\d{2}\-\d{2}(T\d{2}:\d{2}:\d{2}(\.\d{1,3})?)?”

(yyyy-mm-dd[Thh:mm:ss[.fff]] where fff = fraction of a second, up to millisecond)

Example:

```
<Datatype xsi:type=”TimeT”/>
```

7.5.3.1.9 TimeSpanT

Figure 20 shows the IODD representation of the data type TimeSpanT.

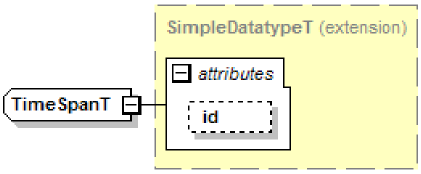


Figure 20 – TimeSpanT

Lexical representation:

Follows the representation of “duration” in XML Schema, see http://www.w3.org/TR/xmlschema-2/#duration, but is much stricter:

Regular expression pattern: “[+-]?PT\d+(\.\d{1,3})?S”

Example:

```
<Datatype xsi:type=”TimeSpanT”/>
```

7.5.3.2 Complex data types

7.5.3.2.1 General

Complex data types are combinations of simple data types. Complex data types consist of several simple data types in a packed manner within a sequence of octets. Unused bit space shall be padded with “0”.

The coding of simple data types within complex data types shall be the same as for singular use specified in chapter 16, except for:

BooleanT

The coding of BooleanT is only 1 bit wide. A value of “0” indicates “false” and a value of “1” indicates “true”. There is no padding to an octet.

UIntegerT and IntegerT

The coding of UIntegerT and IntegerT is as wide as indicated by the attribute ‘bitLength’. There is no padding to 1 / 2 / 4 / 8 octets.

7.5.3.2.2 Arrays

Figure 21 shows the IODD representation of the data type ArrayT.

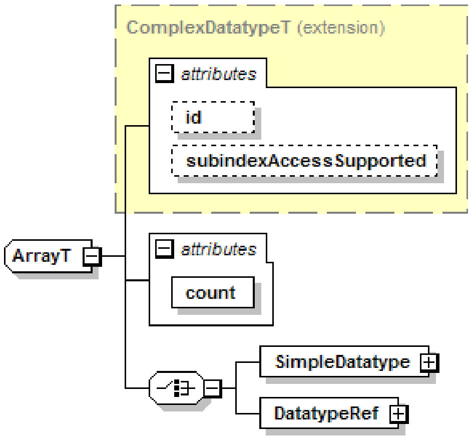


Figure 21 – ArrayT

subindexAccessSupported (o, boolean, default=”true”)
If this attribute is present and set to “false”, individual data items of the array cannot be accessed via their subindex. It is only possible to access the complete array via subindex 0.

count (m, SubindexT)
Specifies the fixed number of data items in the array.

The SimpleDatatype element allows any of the types derived from SimpleDatatypeT. Instead of defining the simple data type inside the array definition, it is also possible to reference the definition of a simple data type from the DatatypeCollection with DatatypeRef/@datatypeId.

Lexical representation:

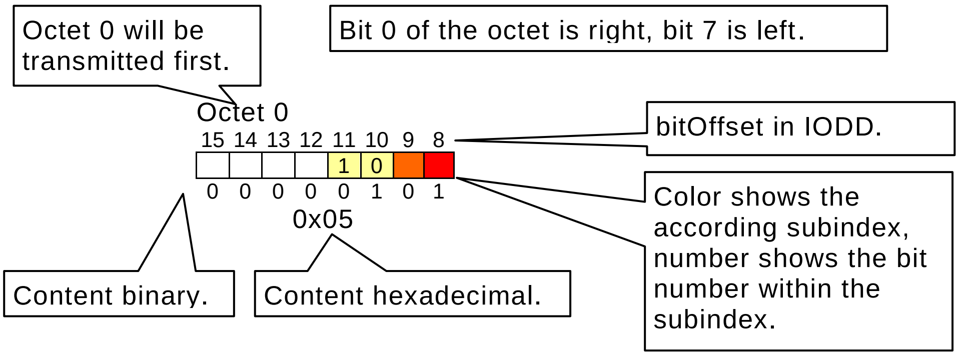
There is no lexical representation for a value of type ArrayT.

Alignment:

Array elements with bitLength < 8 bit shall not cross octet boundaries. For array elements with bitLength >= 8 bit use ‘bitLength’ as multiples of 8 bit.

Examples

Notation:


Octet 0 will be transmitted first.
Bit 0 of the octet is right, bit 7 is left.
Octet 0
15 14 13 12 11 10 9 8
1 0
0 0 0 0 0 1 0 1
0x05
bitOffset in IODD.
Color shows the according subindex, number shows the bit number within the subindex.
Content binary.
Content hexadecimal.

Bit array

```
<Datatype xsi:type=”ArrayT” count=”3”>
        <SimpleDatatype xsi:type=”BooleanT”/>
</Datatype>
```

Subindex	Value
1	true
2	false
3	true

Octet 0

7	6	5	4	3	2	1	0
0	0	0	0	0	1	0	1

0x05

Integer array

```
<Datatype xsi:type=”ArrayT” count=”4”>
        <SimpleDatatype xsi:type=”IntegerT” bitLength=”2”/>
</Datatype>
```

Subindex	Value
1	0
2	-1
3	1
4	1

Octet 0

7	6	5	4	3	2	1	0
1	0	1	0	1	0	1	0
0	0	1	0	0	1	0	1

0x25

Integer array

```
<Datatype xsi:type=”ArrayT” count=”7”>
        <SimpleDatatype xsi:type=”IntegerT” bitLength=”4”/>
</Datatype>
```

Subindex	Value
1	2
2	-4
3	4
4	-7
5	5
6	-1
7	0

Octet 0								Octet 1								Octet 2								Octet 3							
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				3	2	1	0	3	2	1	0	3	2	1	0	3	2	1	0	3	2	1	0	3	2	1	0	3	2	1	0
0	0	0	0	0	0	1	0	1	1	0	0	0	1	0	0	1	0	0	1	0	1	0	1	1	1	1	1	0	0	0	0
0x02								0xC4								0x95								0xF0							

7.5.3.2.3 Records

Figure 22 shows the IODD representation of the data type RecordT.

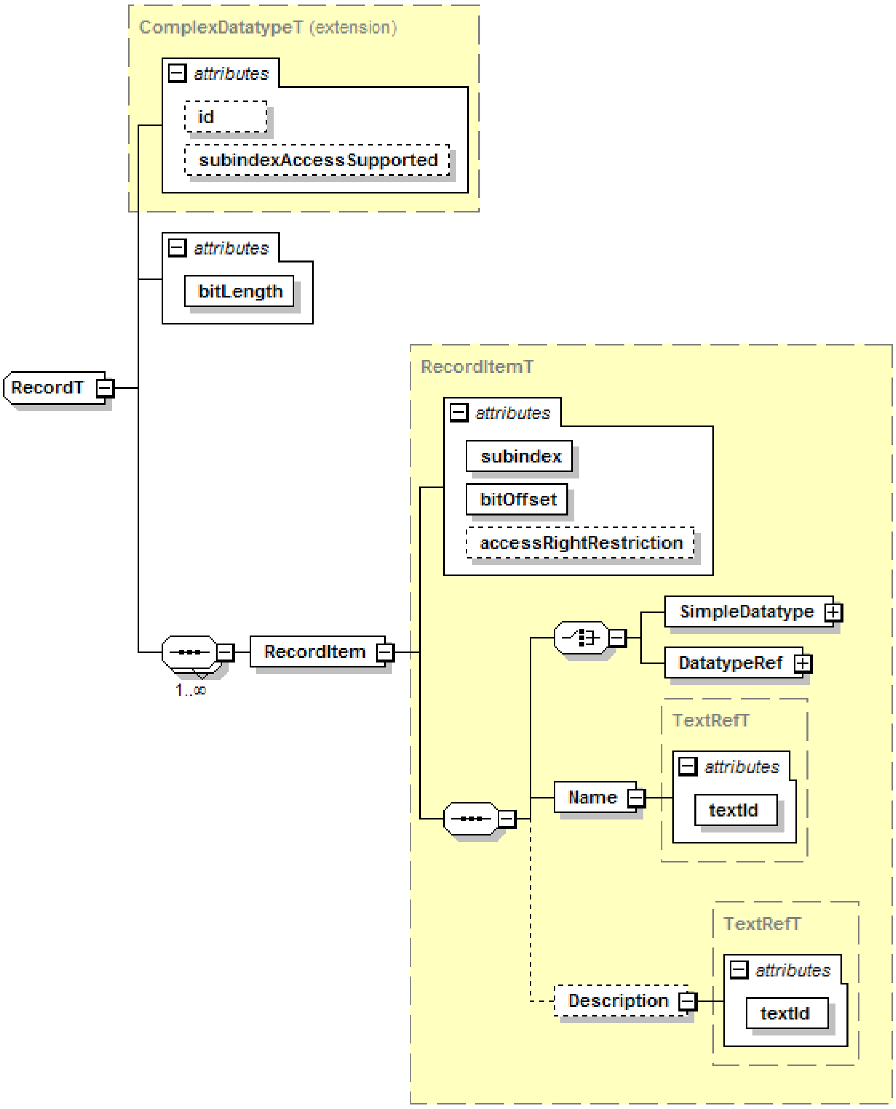


Figure 22 – RecordT

subindexAccessSupported (o, boolean, default=”true”)
If this attribute is present and set to “false”, individual record items cannot be accessed via their subindex. It is only possible to access the complete record via Subindex 0.

bitLength (m, BitCountT)
Specifies the total length of the record in bits. The allowed value range is 1..1856.

RecordItem (m)
An individual item of a record, addressed by the subindex.

subindex (m, SubindexT)
Specifies the Subindex assigned to this record item. The record items shall be ordered by Subindex within the record. The subindex shall be unique within the RecordItems of a Record.

It is recommended that the Subindices occur in increasing order within the octet sequence. If Subindices are placed in previously unused areas of the octet sequence, one might deviate from this recommendation. If compatible extensions are foreseen, it is better to reserve enough Subindices for the unused areas.

bitOffset (m, BitCountT)
bit position of the record item within the octet sequence. The record items within a record shall not overlap. The allowed value range is 0..1855.

accessRightRestriction (o, AccessRightsT)
This attribute is only applicable for variables, not for record items within process data.

Individual record items may have less access rights than the record in general. This is indicated by the attribute ‘accessRightRestriction’. For the access to the complete record, this means:

- If the Record is “rw” and the record item is restricted to “ro” the device shall tolerate (ignore) the data written to this Subindex.

Within the record item, the SimpleDatatype element allows any of the types derived from SimpleDatatypeT. Instead of defining simple data types inside the record definition, it is also possible to reference the definition of simple data types from the DatatypeCollection with DatatypeRef/@datatypeId.

- The attribute ‘accessRightRestriction’ shall not be set to “wo”. An exception to this rule is the V_DirectParameters_1.

Name (m)
textId (m, RefT)
Assigns a human readable name to the record item. This name shall be displayed additionally by the IO-Link Tool.

Description (o)
textId (m, RefT)
Contains a description of the RecordItem (e.g. information text, help, etc.)

Lexical representation:

There is no lexical representation for a value of type RecordT.

Alignment

For variables only, integers shall start on an octet boundary. For optimization of transmission performance, this rule is not applicable for process data.

If you have several booleans, it is highly recommended to group them together into one or few octets.

Examples:

Regarding the notation see chapter 7.5.3.2.2.

Several Booleans in an Octet

```
<Datatype xsi:type="RecordT" bitLength="4">
        <Name textId="TN_Switches"/>
        <RecordItem subindex="1" bitOffset="0">
                <SimpleDatatype xsi:type="BooleanT"/>
                <Name textId="TN_Switch1"/>
        </RecordItem>
        <RecordItem subindex="2" bitOffset="1">
                <SimpleDatatype xsi:type="BooleanT"/>
                <Name textId="TN_Switch2"/>
        </RecordItem>
        <RecordItem subindex="3" bitOffset="2">
                <SimpleDatatype xsi:type="BooleanT"/>
                <Name textId="TN_Switch3"/>
        </RecordItem>
        <RecordItem subindex="4" bitOffset="3">
                <SimpleDatatype xsi:type="BooleanT"/>
                <Name textId="TN_Switch4"/>
        </RecordItem>
</Datatype>
```

RecordItem	Subindex	Datentyp	bitLength	bitOffset	Value
1	1	BooleanT	–	0	true
2	2	BooleanT	–	1	false
3	3	BooleanT	–	2	true
4	4	BooleanT	–	3	false

Octet 0

7	6	5	4	3	2	1	0
0	0	0	0	0	1	0	1

0x05

A word and an octet

```
<Datatype xsi:type="RecordT" bitLength="24">
        <Name textId="TN_Values"/>
        <RecordItem subindex="1" bitOffset="8">
                <SimpleDatatype xsi:type="UIntegerT" bitLength="16"/>
                <Name textId="TN_Value1"/>
        </RecordItem>
        <RecordItem subindex="2" bitOffset="0">
                <SimpleDatatype xsi:type="UIntegerT" bitLength="8"/>
                <Name textId="TN_Value2"/>
        </RecordItem>
</Datatype>
```

RecordItem	Subindex	Datentyp	bitLength	bitOffset	Value
1	1	UIntegerT	16	8	0x9876
2	2	UIntegerT	8	0	0x12

Octet 0

23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8
1	0	0	1	1	0	0	0

0x98

Octet 1

15	14	13	12	11	10	9	8
7	6	5	4	3	2	1	0
0	1	1	1	0	1	1	0

0x76

Octet 2

7	6	5	4	3	2	1	0
7	6	5	4	3	2	1	0
0	0	0	1	0	0	1	0

0x12

Analog value and two signal bits

```
<Datatype xsi:type=”RecordT” bitLength=”16”>
        <Name textId=”TN_ProcessData”/>
        <RecordItem subindex=”1” bitOffset=”2”>
                <SimpleDatatype xsi:type=”UIntegerT” bitLength=”14”/>
                <Name textId=”TN_AnalogValue”/>
        </RecordItem>
        <RecordItem subindex=”2” bitOffset=”1”>
                <SimpleDatatype xsi:type=”BooleanT”/>
                <Name textId=”TN_Signal2”/>
        </RecordItem>
        <RecordItem subindex=”3” bitOffset=”0”>
                <SimpleDatatype xsi:type=”BooleanT”/>
                <Name textId=”TN_Signal1”/>
        </RecordItem>
</Datatype>
```

RecordItem	Subindex	Datentyp	bitLength	bitOffset	Value
1	1	UIntegerT	14	2	0x32F1
2	2	BooleanT	–	1	false
3	3	BooleanT	–	0	true

Octet 0

15	14	13	12	11	10	9	8
13	12	11	10	9	8	7	6
1	1	0	0	1	0	1	1

0xCB

Octet 1

7	6	5	4	3	2	1	0
5	4	3	2	1	0		
1	1	0	0	0	1	0	1

0xC5

Boolean and enumerations in an octet

```
<Datatype xsi:type=”RecordT” bitLength=”8”>
        <Name textId=”TN_ComplexSettings”/>
        <RecordItem subindex=”1” bitOffset=”0”>
                <SimpleDatatype xsi:type=”UIntegerT” bitLength=”4”/>
                <Name textId=”TN_Enum1”/>
        </RecordItem>
        <RecordItem subindex=”2” bitOffset=”4”>
                <SimpleDatatype xsi:type=”BooleanT”/>
                <Name textId=”TN_Switch1”/>
        </RecordItem>
        <RecordItem subindex=”3” bitOffset=”5”>
                <SimpleDatatype xsi:type=”BooleanT”/>
                <Name textId=”TN_Switch2”/>
        </RecordItem>
        <RecordItem subindex=”4” bitOffset=”6”>
                <SimpleDatatype xsi:type=”UIntegerT” bitLength=”2”/>
                <Name textId=”TN_Enum2”/>
        </RecordItem>
</Datatype>
```

RecordItem	Subindex	Datentyp	bitLength	bitOffset	Value
1	1	UIntegerT	4	0	0xF
2	2	BooleanT	–	4	false
3	3	BooleanT	–	5	true
4	4	UIntegerT	2	6	0x3

Octet 0

7	6	5	4	3	2	1	0
1	0			3	2	1	0
1	1	1	0	1	1	1	1

0xEF

With a gap (reserved area for future extension)

```
<Datatype xsi:type="RecordT" bitLength="40">
        <Name textId="TN_Gap"/>
        <RecordItem subindex="1" bitOffset="24">
                <SimpleDatatype xsi:type="UIntegerT" bitLength="16"/>
                <Name textId="TN_Value1"/>
        </RecordItem>
        <RecordItem subindex="3" bitOffset="0">
                <SimpleDatatype xsi:type="UIntegerT" bitLength="16"/>
                <Name textId="TN_Value2"/>
        </RecordItem>
</Datatype>
```

RecordItem	Subindex	Datentyp	bitLength	bitOffset	Value
1	1	UIntegerT	16	24	0xBABE
2	3	UIntegerT	16	0	0xCAFE

Octet 0

39	38	37	36	35	34	33	32
15	14	13	12	11	10	9	8
1	0	1	1	1	0	1	0

0xBA

Octet 1

31	30	29	28	27	26	25	24
7	6	5	4	3	2	1	0
1	0	1	1	1	1	1	0

0xBE

Octet 2

23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0

0x00

Octet 3

15	14	13	12	11	10	9	8
15	14	13	12	11	10	9	8
1	1	0	0	1	0	1	0

0xCA

Octet 4

7	6	5	4	3	2	1	0
7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	0

0xFE

Previous example, extended with two record items

```
<Datatype xsi:type="RecordT" bitLength="40">
        <Name textId="TN_GapFilled"/>
        <RecordItem subindex="1" bitOffset="24">
                <SimpleDatatype xsi:type="UIntegerT" bitLength="16"/>
                <Name textId="TN_Value1"/>
        </RecordItem>
        <RecordItem subindex="2" bitOffset="16">
                <SimpleDatatype xsi:type="UIntegerT" bitLength="4"/>
                <Name textId="TN_Enum"/>
        </RecordItem>
        <RecordItem subindex="3" bitOffset="0">
                <SimpleDatatype xsi:type="UIntegerT" bitLength="16"/>
                <Name textId="TN_Value2"/>
        </RecordItem>
        <RecordItem subindex="4" bitOffset="20">
                <SimpleDatatype xsi:type="BooleanT"/>
                <Name textId="TN_Switch"/>
        </RecordItem>
</Datatype>
```

RecordItem	Subindex	Datentyp	bitLength	bitOffset	Value
1	1	UIntegerT	16	24	0xBABE
2	2	UIntegerT	4	16	0xB
3	3	UIntegerT	16	0	0xCAFE
4	4	BooleanT	–	20	false

Octet 0

39	38	37	36	35	34	33	32
15	14	13	12	11	10	9	8
1	0	1	1	1	0	1	0

0xBA

Octet 1

31	30	29	28	27	26	25	24
7	6	5	4	3	2	1	0
1	0	1	1	1	1	1	0

0xBE

Octet 2

23	22	21	20	19	18	17	16
0	0	0	0	1	0	1	1

0x0B

Octet 3

15	14	13	12	11	10	9	8
15	14	13	12	11	10	9	8
1	1	0	0	1	0	1	0

0xCA

Octet 4

7	6	5	4	3	2	1	0
7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	0

0xFE

Two signal bits with reserved space

```
<Datatype xsi:type=”RecordT” bitLength=”12”>
        <Name textId=”TN_ProcessData”/>
        <RecordItem subindex=”1” bitOffset=”0”>
                <SimpleDatatype xsi:type=”BooleanT”/>
                <Name textId=”TN_Signal2”/>
        </RecordItem>
        <RecordItem subindex=”2” bitOffset=”1”>
                <SimpleDatatype xsi:type=”BooleanT”/>
                <Name textId=”TN_Signal1”/>
        </RecordItem>
</Datatype>
```

RecordItem	Subindex	Datentyp	bitLength	bitOffset	Value
1	1	BooleanT	–	0	false
2	2	BooleanT	–	1	true

Octet 0

14	14	13	12	11	10	9	8
				▨	▨	▨	▨
0	0	0	0	0	0	0	0

0x00

Octet 1

7	6	5	4	3	2	1	0
▨	▨	▨	▨	▨	▨		
0	0	0	0	0	0	1	0

0x02

7.5.3.3 Process data union data types

The types ProcessDataInUnionT and ProcessDataOutUnionT are restricted to the description of the process data standard variables (Index 40 and 41) in IODD-StandardDefinitions1.1.xml and thus are not allowed in a normal IODD. The IO-Link Tool shall take the data type of the appropriate ProcessDataIn / ProcessDataOut element. If more than one ProcessDataIn / ProcessDataOut element is given, it is necessary to select the currently valid element by evaluating the Condition elements.

7.5.3.3.1 ProcessDataInUnionT

A ProcessDataInUnionT corresponds to the data type used in ProcessDataCollection/ ProcessData/ProcessDataIn.

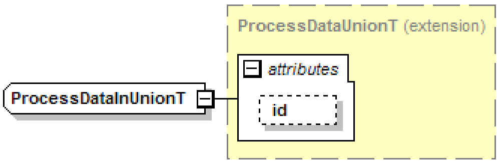


Figure 23 – ProcessDataInUnionT

7.5.3.3.2 ProcessDataOutUnionT

A ProcessDataOutUnionT corresponds to the data type used in ProcessDataCollection/ ProcessData/ProcessDataOut.

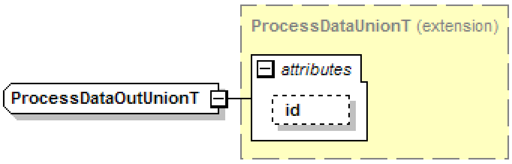


Figure 24 – ProcessDataOutUnionT

7.5.4 Variable collection

All parameters of the device are included here. Standard parameters are defined in IODD-StandardDefinitions1.1.xml and are referenced by StdVariableRef. DirectParameterOverlay allows defining a Record which is being layed over the DirectParameterPage 2 (DirectParameters 16 – 31). All other device-specific variables are named under ‘Variable’.

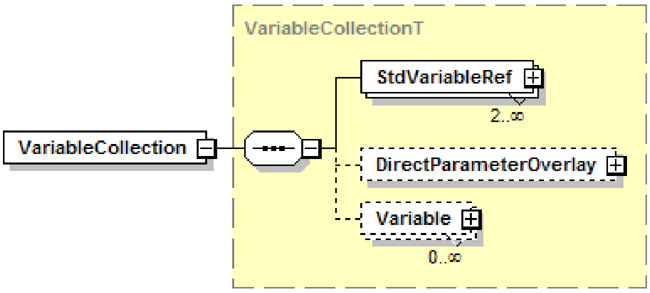


Figure 25 – VariableCollection element

All the variables that the device supports (i.e. the referenced standard variables, the direct parameter overlay and the vendor specific variables) shall have unique Names within each supported language.

7.5.4.1 StdVariableRef

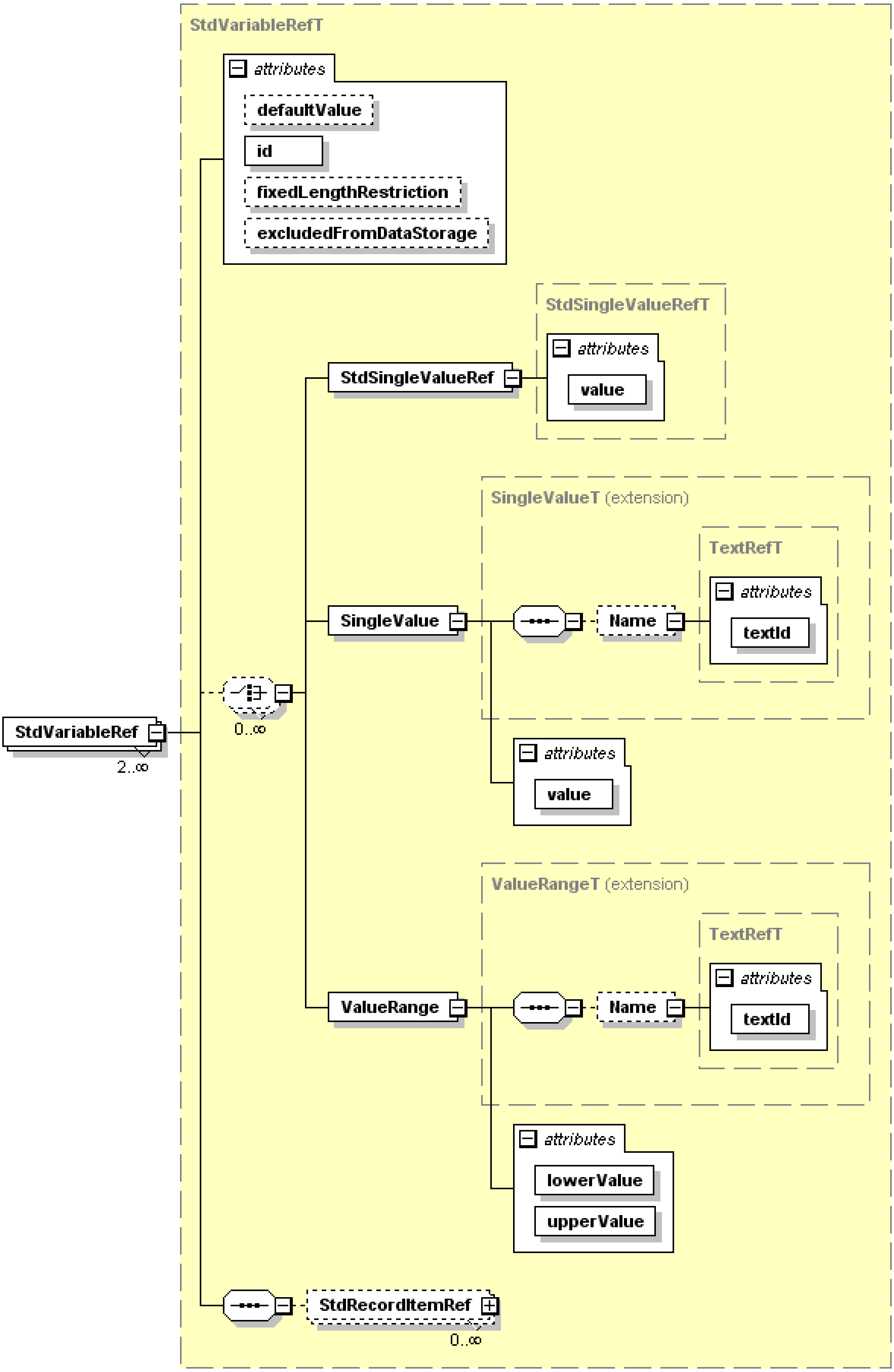
Here it is described, which of the standard variables are used. They are referenced here by an explicit key. Because direct parameter page 1 is mandatory, the variable V_DirectParameters_1 shall always be referenced.

It is highly recommended not to use V_DirectParameters_2 and ISDU in the same Device. Use the V_DirectParameter_2 only if your Device does not support ISDUs at all.

V_DirectParameters_2 and DirectParameterOverlay may only be present both or none. If V_ DirectParameters_2 and DirectParameterOverlay are present, V_DirectParameters_2 shall not be referenced and DirectParameterOverlay shall be referenced in menu.

All standard ISDU variables marked with the attribute mandatory="true" in the IODD-StandardDefinitions1.1.xml shall be referenced if the device supports ISDU access. The optional standard variables V_ProcessDataInput and V_ProcessDataOutput shall only be referenced if there is at least one ProcessDataIn / ProcessDataOut element in the ProcessDataCollection.

The standard variables V_ProfileCharacteristic, V_PDInputDescriptor, V_PDOutputDescriptor shall neither be described nor be referenced in the IODD.



1270

1271 **Figure 26 – StdVariableRef element**

id (m, IdT)
This id is special since it can be both starting and end point of a referencing process. As end point of the referencing process, it contains the key of those variables within the IODD. As starting point, it references to a standard variable.

defaultValue (o, anySimpleType)
The defaultValue shall conform to the data type of the standard variable. Offline default value; it always refers to the complete variable. If the variable is a record, use StdRecordItemRef element(s) to specify default values for individual record items. On a variable of type array, the specified defaultValue shall be applied to all array members. For references to V_ProcessDataInput or V_ProcessDataOutput this attribute shall not be specified.

For references to V_ProductID this attribute shall not be specified if more than one DeviceVariant is defined. If there is only one DeviceVariant, then the value of DeviceVariant/@productId shall be used as default value of V_ProductID.

If more than one device variant is available, the defaultValue of V_ProductName should not be specified. Alternatively a substitute text covering all Device variants is allowed to be used as defaultValue. A specific name of any of the variants shall not be used.

fixedLengthRestriction (o, SubindexT)
Only applicable to standard variables of type string, octet string or array. Standard variables of types string or octet string have a 'fixedLength' attribute describing the maximum length that the IO-Link standard allows. A Device may implement standard variables with (octet) string shorter than what the IO-Link standard allows.

Standard Variables of type array have a 'count' attribute describing the maximum size that the IO-Link standard allows. A Device may implement standard variables with arrays shorter than what the IO-Link standard allows.

'fixedLengthRestriction' shall be less or equal to the 'fixedLength' (on standard variables of type string or octetstring) or 'count' (on standard variables of type array). If 'fixedLengthRestriction' is used with V_ApplicationSpecificTag, it shall be greater or equal to 16.

excludedFromDataStorage (o, boolean)
If set to "true", indicates that the contents of the standard variable are not stored with the data storage mechanism. This attribute may only be set to "true" for standard variables with accessRights = "rw". Within StdVariableRefs, excludedFromDataStorage can only be applied to V_ApplicationSpecificTag, V_DeviceAccessLocks and V_OffsetTime. The default is "false".

Allowed values:
Only applicable to the standard variable V_SystemCommand and V_OffsetTime.

> **StdSingleValueRef (o)**
> Specifies a single supported standard value. The 'value' attribute shall match the 'value' attribute of a SingleValue defined at the standard variable.
>
> **SingleValue (o)**
> Specifies a single supported vendor-specific value with an optional name.
>
> **ValueRange (o)**
> Specifies a supported vendor-specific value range with an optional name.

In addition to the above, the following rules shall apply for referencing for standard variables with StdVariableRef or StdRecordItemRef:

- When neither SingleValue nor ValueRange nor StdSingleValueRef elements are given, the standard variable's value range as defined in IODD-StandardDefinitions1.1.xml is taken. This rule is not valid for V_SystemCommand.
- When SingleValue(s) or ValueRange(s) or StdSingleValueRef(s) are given, only these values are allowed.

- SingleValues and ValueRanges shall not overlap with SingleValues at the standard variable, no matter whether these are referenced by StdSingleValueRef or not (i.e. standard values can’t be redefined in a vendor specific way).

StdRecordItemRef (o)
Used to specify additional information for RecordItems of standard variables. At least one of the optional attributes and elements shall be present.

For StdVariableRef id=”V_DeviceAccessLocks”, StdRecordItemRef is only allowed for those subindices which refer to an access lock that is supported, i.e. where the respective attribute in Features/SupportedAccessLocks is set to “true”. It is recommended for V_DeviceAccessLocks to specify StdRecordItemRef with a defaultValue for a specific supported lock.

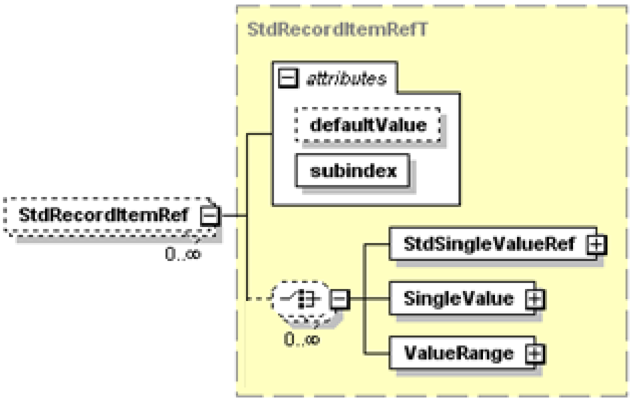


Figure 27 – StdRecordItemRef element

subindex (m, SubindexT)
Used for addressing the record item within the record. The subindex shall be unique within the StdRecordItemRefs of a StdVariableRef.

defaultValue (o, anySimpleType)
The defaultValue shall conform to the data type of the RecordItem. Offline default value.

Specifying allowed values for a RecordItem using StdSingleValueRef, SingleValue and / or ValueRange is only applicable to the standard variable V_DirectParameters_1, subindex 16 (system command for devices without ISDU support). The meaning and the rules regarding these elements shall be the same as with the StdVariableRef element shown above.

7.5.4.2 DirectParameterOverlay

This element corresponds to the device-specific data within the DirectParameter page. If the DirectParameterOverlay is used, TextRedefines should also be added to provide names for each used DirectParameter octet (see chapter 14).

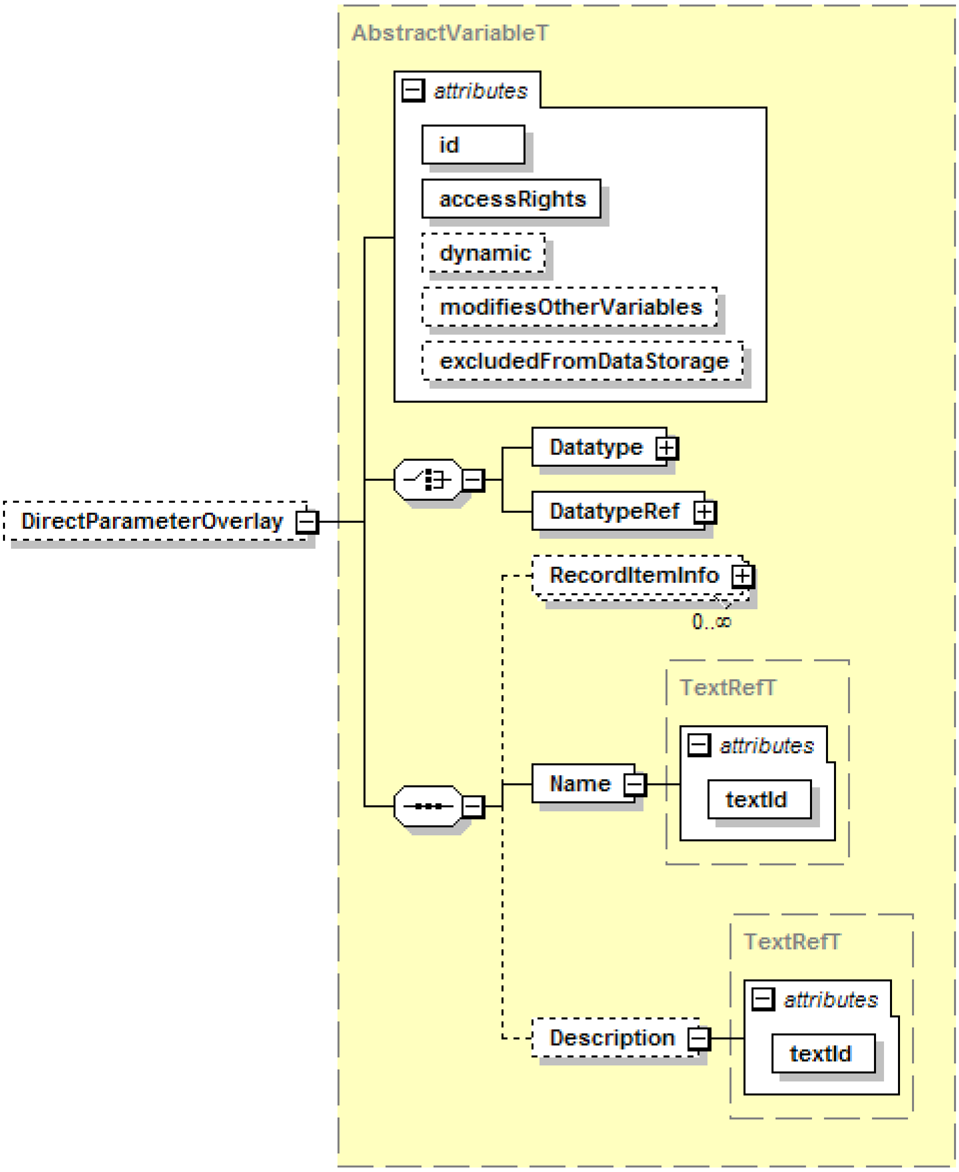


Figure 28 – DirectParameterOverlay element

id (m, IdT)
As the end point of a referencing process, it contains the key of the variable within the IODD. The id of any standard variable shall not be used as id for the DirectParameterOverlay, even if the standard variable is not referenced from this IODD.

accessRights (m, AccessRightsT)
“ro” read-only,
“wo”, write-only,
“rw”, read-write

dynamic (o, boolean)
Serves as information, whether the variable is autonomously changed by the device. This attribute may only be set to “true” for DirectParameterOverlay with accessRights = “rw” or “ro”. The default is “false”.

modifiesOtherVariables (o, boolean)
It is highly recommended to omit this attribute or set it to “false”.

excludedFromDataStorage (o, boolean)
It is highly recommended to omit this attribute or set it to “false”.

Datatype (c)
Directly given data type (see Note below)

DatatypeRef (c)
Reference to a data type that was defined in the DatatypeCollection (see Note below)

RecordItemInfo (o)
Contains additional information for record items. See chapter 7.5.4.4.

Name (m)
textId (m, RefT)
Contains the name of the variable

Description (o)
textId (m, RefT)
Contains a description of the variable (e.g. information text, help, etc.)

Note: The data type shall be a record with a minimum length of 1 bit and a maximum length of 128 bits. The last octet of this record is mapped to the last octet of the direct parameter page 2.

Example

RecordItem	Subindex	Datentyp	bitLength	bitOffset
1	1	UIntegerT	16	32
2	2	UIntegerT	32	0

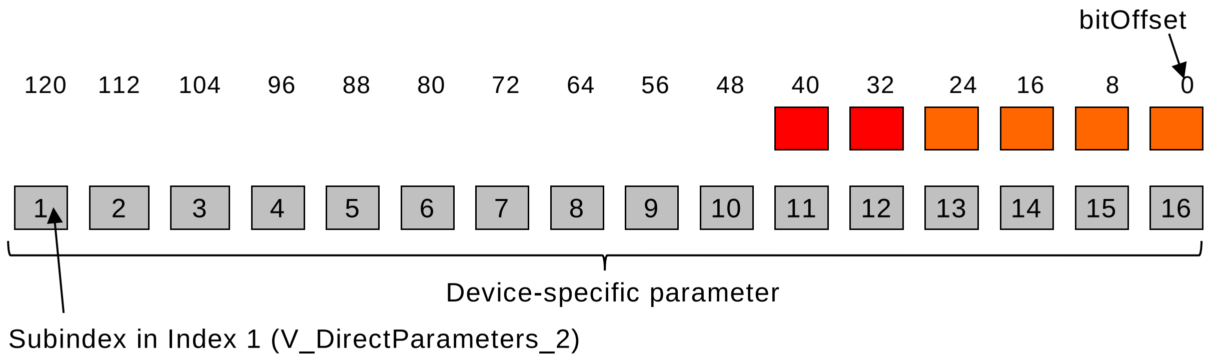

bitOffset
120
112
104
96
88
80
72
64
56
48
40
32
24
16
8
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
Device-specific parameter
Subindex in Index 1 (V_DirectParameters_2)

Figure 29 – Direct parameter overlay

Note: The communication of direct parameters is octet oriented.

- For record items, which cross an octet boundary the consistency cannot be guaranteed.
- If an octet contains more than one record item, the subindex access will influence all contained record items or parts.
- For record items, which cross an octet boundary, the device cannot rely on the order of the single accesses. This means, the device shall tolerate intermediate values that may exceed the allowed value range.

Recommendation: Use DirectParameterOverlay only for devices that do not support ISDU access.

7.5.4.3 Variable

Contains the description of a device parameter.

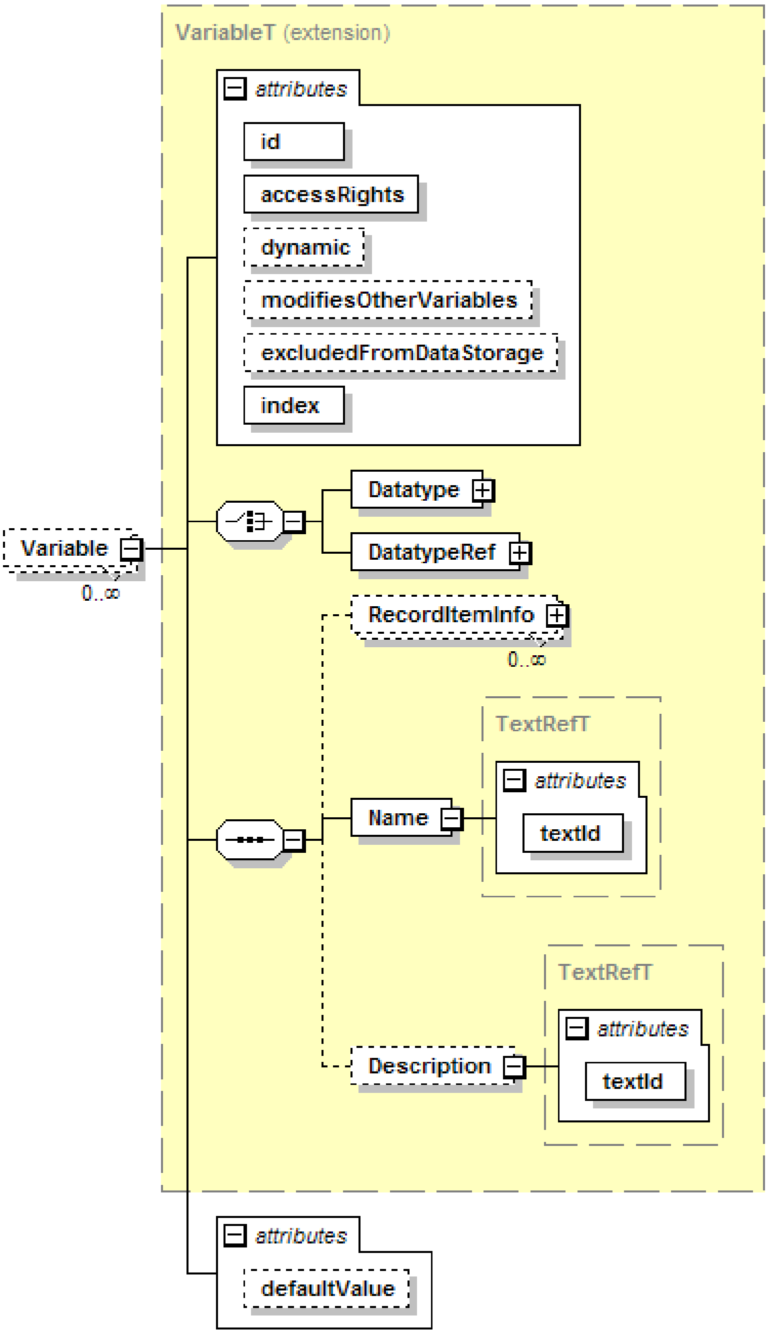


Figure 30 – Variable element

id (m, IdT)
As the end point of a referencing process, it contains the key of the variable within the IODD. The id of any standard variable shall not be used as id for the Variable, even if the standard variable is not referenced from this IODD.

accessRights (m, AccessRightsT)
“ro” read-only,
“wo”, write-only,
“rw”, read-write

For records and arrays the attribute accessRights=’wo’ is not permitted.

dynamic (o, boolean)
Serves as information, whether the variable is autonomously changed by the device. This attribute may only be set to “true” for variables with accessRights = “rw” or “ro”. The default is “false”.

modifiesOtherVariables (o, boolean)
If set to “true”, indicates that a write access to this variable (or to any of its subindices) may change the value of other variables. IO-Link Tools should re-load the variables of the device after a write access to this variable. This attribute may only be set to “true” for variables with accessRights = “rw” or “wo”. The default is “false”.

excludedFromDataStorage (o, boolean)
If set to “true”, indicates that this variable’s contents are not stored with the data storage mechanism. This attribute may only be set to “true” for variables with accessRights = “rw”. The default is “false”.

index (m, unsignedShort)
Index for the addressing of a variable. Shall be in the range for vendor specific indices or profile specific indices that are not already described as standard variables in IODD-StandardDefinitions1.1.xml.

defaultValue (o, anySimpleType)
The defaultValue shall conform to the data type of the variable. Offline default value; it always refers to the complete variable. If the variable is a record, use RecordItemInfo element(s) to specify default values for individual record items. On a variable of type array, the specified defaultValue shall be applied to all array members.

Datatype (c)
Directly given data type

DatatypeRef (c)
Reference to a data type that was defined in the DatatypeCollection

RecordItemInfo (o)
Only applicable if the variable is of type record. Contains additional information for record items. See chapter 7.5.4.4.

Name (m)
textId (m, RefT)
Contains the name of the variable

Description (o)
textId (m, RefT)
Contains a description of the variable (e.g. information text, help, etc.)

7.5.4.4 RecordItemInfo

For variables of type RecordT, contains optional attributes for a RecordItem addressed by the subindex. At least one of the optional attributes shall be present.

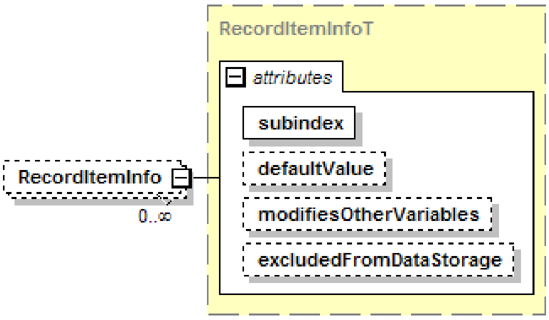


Figure 31 – RecordItemInfo element

subindex (m, SubindexT)
Used for addressing the record item within the record.

defaultValue (o, anySimpleType)
The defaultValue shall conform to the data type of the record item. Contains the default value for the RecordItem.

modifiesOtherVariables (o, boolean)
If set to “true”, indicates that a write access to this subindex may change the value of other variables. IO-Link Tools should re-load the variables of the device after a write access to this subindex. This attribute may only be set to “true” for record items with accessRights = “rw” or “wo”. This attribute shall not be specified both on the DirectParameterOverlay/Variable element and a subordinated RecordItemInfo element. The default is “false”.

excludedFromDataStorage (o, boolean)
If set to “true”, indicates that this subindex’s contents are not stored with the data storage mechanism. This attribute may only be set to “true” for record items with accessRights = “rw”. This attribute shall not be specified both on the DirectParameterOverlay/Variable element and a subordinated RecordItemInfo element. The default is “false”.

7.5.5 Process data collection

Contains all process data of the device

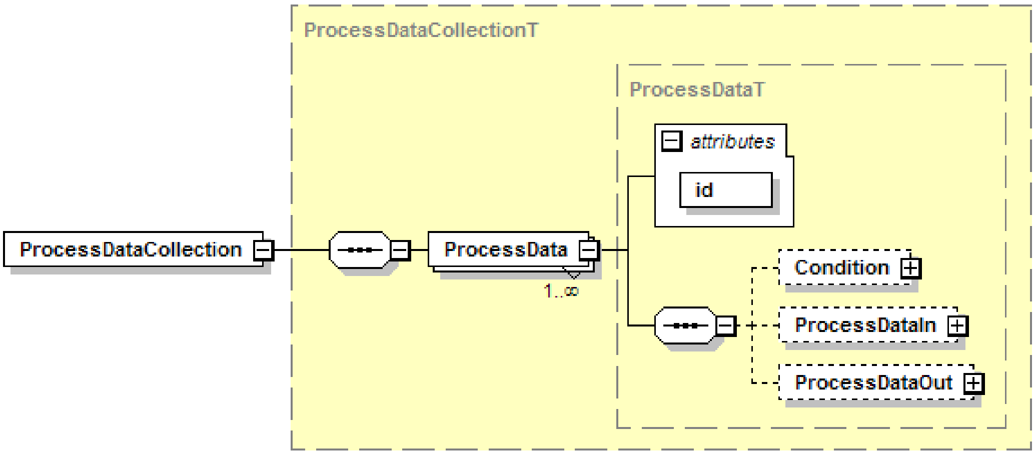


Figure 32 – ProcessDataCollection element

The element ProcessData may occur multiple times in the collection. If ProcessData occurs more than once,

- all the ProcessData elements shall contain a Condition element
- for each user role there shall be a menu (could be the same) that contains at least a read-only reference to the Variable / RecordItem used in the ProcessData/Condition elements
- the attributes ‘variableId’ and ‘subindex’ in the Condition elements shall be the same (there shall only be exactly one variable / record item used for the switching of the process data)
- the attribute ‘value’ in the Condition elements shall be unique within the ProcessData elements
- all defined values of the variable, which is used as Condition within the ProcessData elements shall be referenced
- the attribute ‘bitLength’ in the ProcessDataIn elements shall be the same for all ProcessData

- the attribute ‘bitLength’ in the ProcessDataOut elements shall be the same for all ProcessData
- the variable / record item referenced in the Condition elements selects the currently valid ProcessData element when its value matches the ‘value’ attribute of the Condition element

The attribute ‘id’ shall be unique within all the elements ProcessData, ProcessDataIn and ProcessDataOut.

ProcessData (m)
id (m, IdT)
Explicit id of the ProcessData

Condition (o)
Serves to switch between different ProcessData.

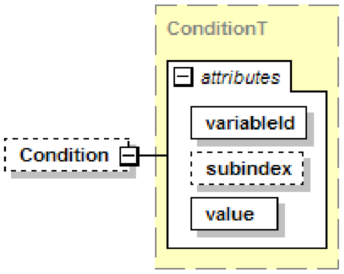


Figure 33 – Condition element

variableId (m, RefT)
References a variable. The variable shall be of data type BooleanT, IntegerT, UIntegerT or RecordT. The variable shall have a default value if it is not of type RecordT.

subindex (c, SubindexT)
This attribute shall be given if and only if the referenced variable is of type RecordT. Used for addressing the record item within the record. The record item shall be of data type BooleanT, IntegerT or UIntegerT and shall have a default value.

value (m, unsignedByte)
Shall be a valid value for the variable / record item. This attribute can only hold values 0..255, thus limiting the possible IntegerT and UIntegerT values. Also, BooleanT condition values shall be entered as “0” for “false” and “1” for “true”.

ProcessDataIn (o)
Description of the input process data

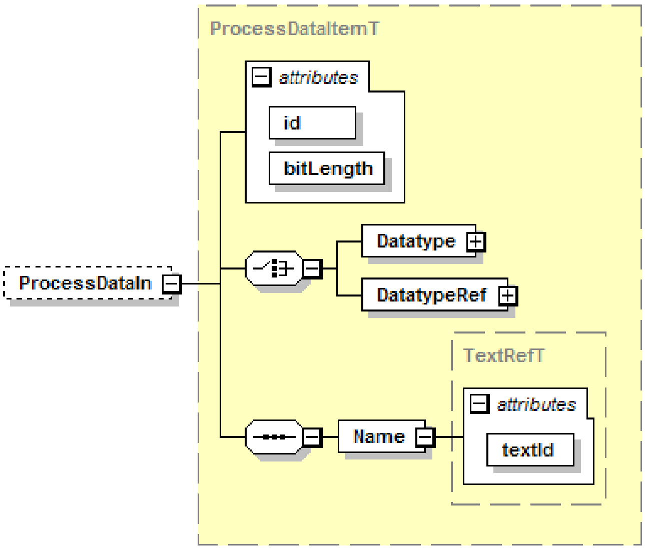


Figure 34 – ProcessDataIn element

id (m, IdT)
Explicit id of the ProcessDataIn description

bitLength (m, BitCountT)
Length of the input process data (in bits). The allowed value range is 1..256.

It shall represent the underlying ProcessDataIn data type in a bit granular manner. For record data types this ‘bitLength’ shall equal the ‘bitLength’ attribute of the record.

The value of the DirectParameterPage 1, subindex 5 (Process Data In), shall be calculated from the ‘bitLength’ attribute value by the following formula:

```
If bitLength <= 16 then
    ProcessDataIn = bitLength
Else
    ProcessDataIn = bitLength rounded up to the next multiple of 8
End If
```

Name (m)
textId (m, RefT)
Name specification of the input process data

ProcessDataOut (o)
Description of the output process data

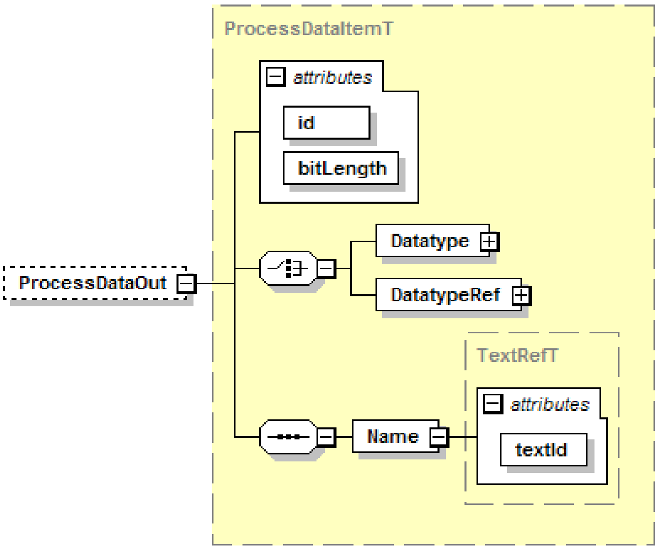


Figure 35 – ProcessDataOut element

id (m)
Explicit id of the ProcessDataOut description

bitLength (m, BitCountT)
Length of the output process data (in bits). The allowed value range is 1..256.

The description of the 'bitLength' attribute at the ProcessDataIn element above also applies here, but for Process Data Out (DirectParameterPage 1, subindex 6).

Name (m)
textId (m, RefT)
Name specification of the output process data

7.5.6 Error type collection

All error types that the device may return are collected here. There are system defined error types (code=128) and vendor specific error types (code=129), see *IO-Link Interface and System* Specification Version 1.1.4, annex C. The system defined error types are described in IODD-StandardDefinitions1.1.xml and referenced by 'StdErrorTypeRef', while the vendor specific error types are specified with 'ErrorType'.

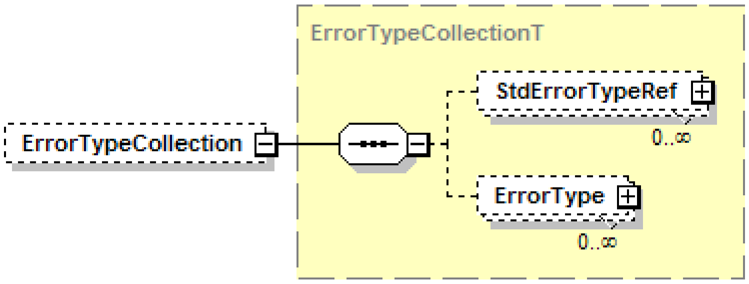


Figure 36 – ErrorTypeCollection element

StdErrorTypeRef (o)
Standard error types are referenced by their 'additionalCode'.

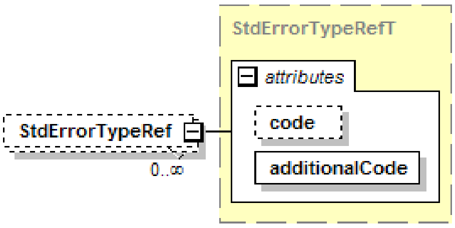


Figure 37 – StdErrorTypeRef element

code (o, unsignedByte)
Fixed to 128 by the IO-Link specification.

additionalCode (m, unsignedByte)
The additional code. Shall be unique within the ‘StdErrorTypeRef’ elements, and shall reference one of the error types defined in IODD-StandardDefinitions1.1.xml.

ErrorType (o)
Vendor specific error type, identified by its ‘additionalCode’.

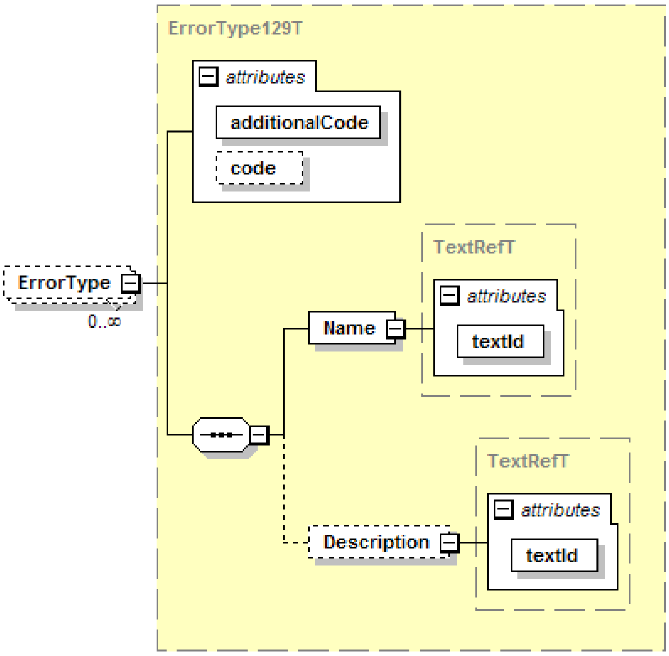


Figure 38 – ErrorType element

code (o, unsignedByte)
Fixed to 129 by the IO-Link specification.

additionalCode (m, unsignedByte)
The additional code. Shall be unique within the ‘ErrorType’ elements.

Name (m)
textId (m, RefT)
Use this text for the error message.

Description (o)
textId (m, RefT)
Use this text for the possible cause of the error and the remedy.

Error Types shall be handled by IO-Link Tools.

Since a device can respond to an ISDU with an ‘ErrorType’, IO-Link Tools shall list all incoming ErrorTypes during an up- or download. Up- or downloads shall not be interrupted. If a tool supports device replication and the device responds with a single ‘ErrorType’, this replication shall be interrupted.

7.5.7 Event collection

All events that the device may return are collected here. There are system defined events and vendor specific events, see *IO-Link Interface and System* Specification Version 1.1.4, annex D. The system defined events are described in IODD-StandardDefinitions1.1.xml and referenced by ‘StdEventRef’, while the vendor specific events are specified with ‘Event’.

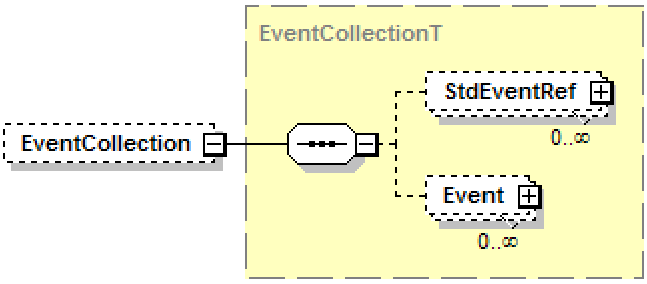


Figure 39 – EventCollection element

StdEventRef (o)
Indicates that the device may return the standard event identified by the ‘code’.

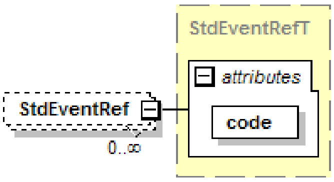


Figure 40 – StdEventRef element

code (m, unsignedShort)
The event code that identifies the standard event described in IODD-StandardDefinitions1.1.xml.

Event (o)
Describes a vendor specific event.

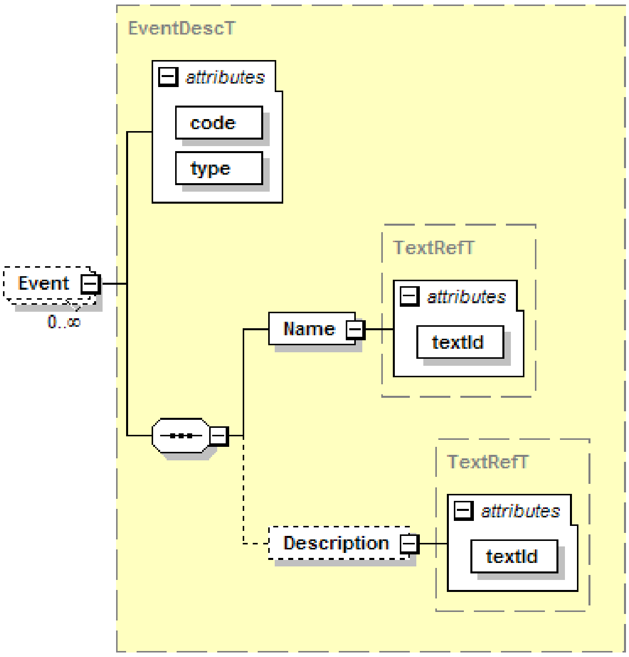


Figure 41 – Event element

code (m, unsignedShort)
The event code that identifies the vendor specific event. Shall be in the range for vendor specific or profile specific event codes.

type (m, string)
The TYPE part of the EventQualifier (see *IO-Link Interface and System* Specification Version 1.1.4, chapter A.6.4). One of “Notification”, “Warning” or “Error”.

Note that the MODE depends on the TYPE, SOURCE is always 0 (device application) and INSTANCE is always 4 (application).

Name (m)
textId (m, RefT)
Use this text for the event message.

Description (o)
textId (m, RefT)
Use this text for the possible cause of the event and the remedy.

7.5.8 User interface

Contains the menus of the device

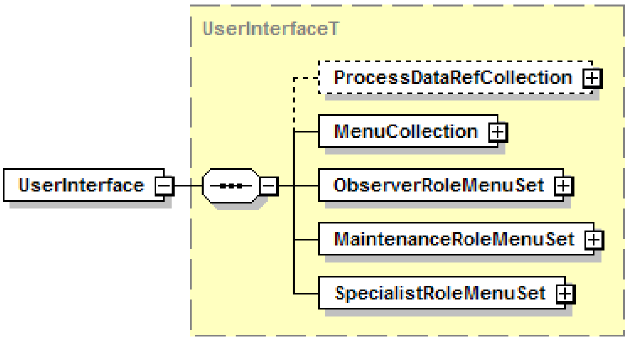


Figure 42 – UserInterface element

7.5.8.1 ProcessDataRef collection (o)

Some IO-Link masters support read access to the process data independently of the device. Process data read this way are shown in a separate menu by the IO-Link Tools for these masters.

Because not all IO-Link Masters support this access, a device should support the optional standard variables V_ProcessDataInput and V_ProcessDataOutput for access to the process data.

Usually these variables are then referenced from the Observation menu.

The ProcessDataRefCollection allows defining how the process data read via the master or VariableRef with V_ProcessDataInput and V_ProcessDataOutput are to be displayed.

If an IO-Link Tool wants to display process data and the IODD does not contain a corresponding ProcessDataRef for it, the tool may display the process data just according to its data type.

For compatibility it is still allowed to define the Observation menu with ‘RecordItemRef’ or ‘VariableRef’ to control how the process data read from the device are to be displayed.

The description in the ProcessDataRefCollection and in the Observation Menu should be entered using the same attribute values.

Non referenced subindices shall not be displayed.

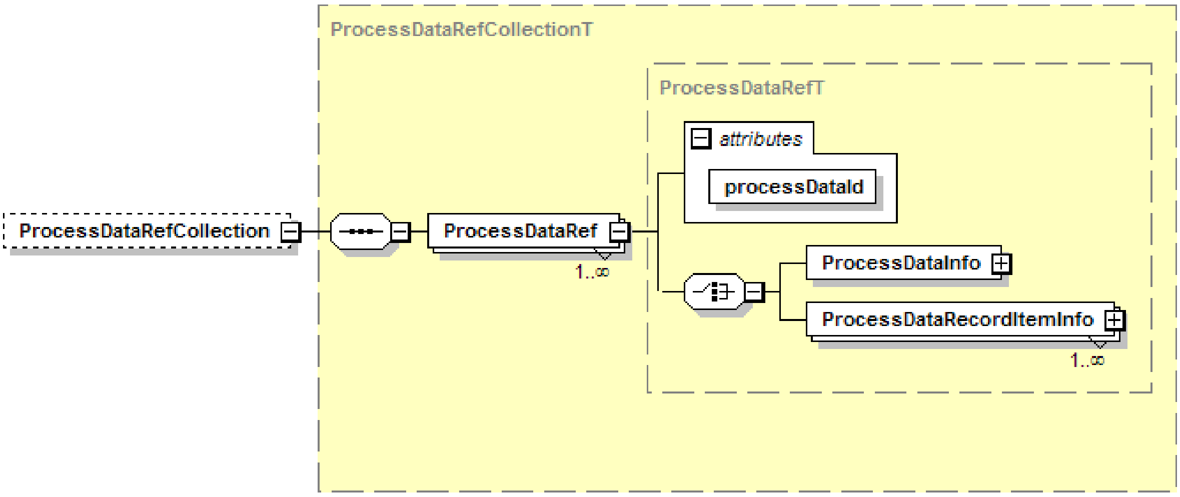


Figure 43 – ProcessDataRefCollection element

processDataId (m, RefT)
Refers to DeviceFunction/ProcessData/ProcessDataIn or ProcessDataOut.

ProcessDataInfo (c)
Corresponds to the ‘VariableRef’ element of a menu except for ‘accessRightRestriction’ and ‘Button’ which are not applicable (see chapter 7.5.8.4.).

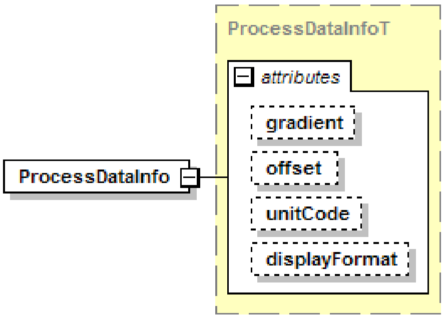


Figure 44 – ProcessDataInfo element

Use this for process data which is not of type record.

ProcessDataRecordItemInfo (c)
Corresponds to the ‘RecordItemRef’ element of a menu except for ‘accessRightRestriction’ and ‘Button’ which are not applicable (see chapter 7.5.8.5.).

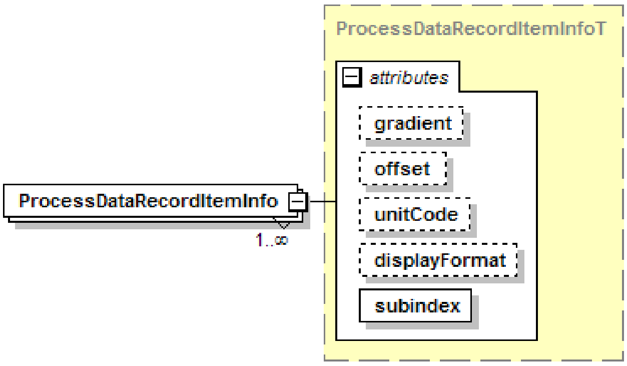


Figure 45 – ProcessDataRecordItemInfo element

Use this for process data which is of type record.

7.5.8.2 MenuSets (m)

User Roles
A user interface shall be divided into three user roles. It is up to the vendor how the roles are organized. The IO-Link Tool shall assign the entered UserLevel to the respective menu. At most three menu levels below the role assignment are acceptable.

Example:

ObservationRoleMenuSet
→ IdentificationMenu
→ Menu1
→ MenuRef1
→ Menu2
→ MenuRef1
MaintenanceRoleMenuSet
→ ObservationMenu
→ MenuX
→ MenuRefY

IO-Link Tools shall upload or download only the variables of the current user role. If the tool supports a special function to replicate an IO-Link device, this function shall use the variables of the specialist role and can be available in all user roles.

ObserverRoleMenuSet (m)

This menu is designed for users who may not carry out any modifications on the device.

The role name in English: “Operator”.

MaintenanceRoleMenuSet (m)

This menu is designed for observers and to undertake “uncritical” editing. It is up to the vendor to assess that.

The role name in English: “Maintenance”.

SpecialistRoleMenuSet (m)

If the user is logged in as a specialist, he/she has total access to the device. Again, the vendor can decide which parameters may be edited.

The role name in English: “Specialist”.

For each role, there is a set of fixed top-level menus given.

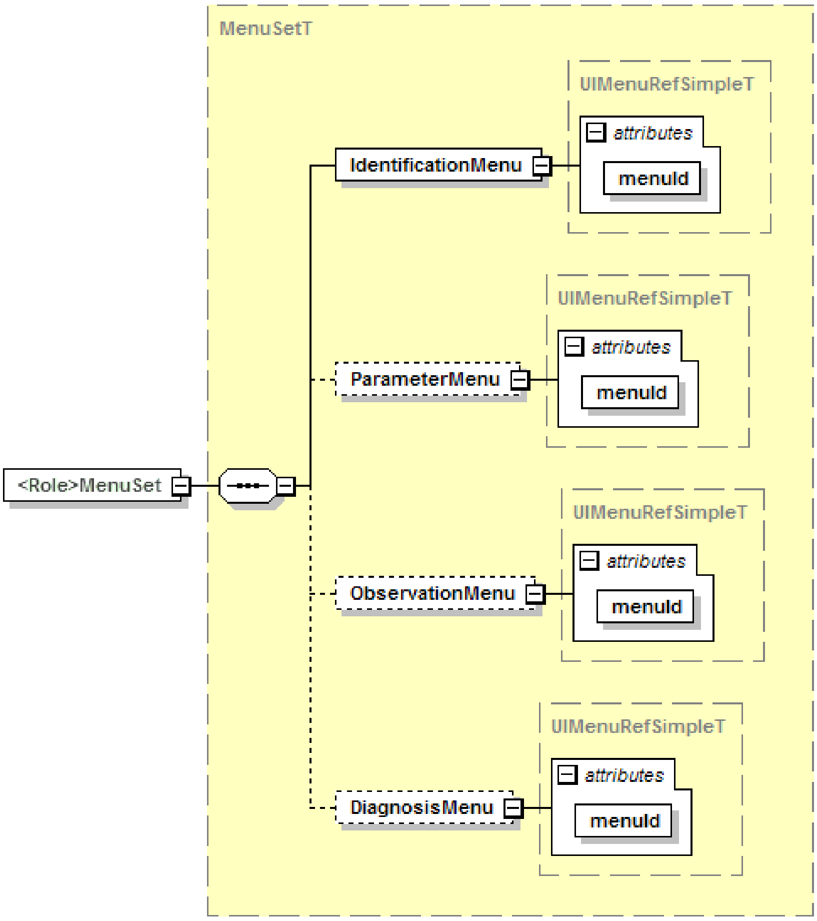


Figure 46 – <Role>MenuSet element

IdentificationMenu (m)

The attribute ‘menuId’ references a menu from the MenuCollection. This menu should contain variables which serve the identification of the device.

The menu name in English: “Identification”.

ParameterMenu (o)
The attribute 'menuId' references a menu from the MenuCollection. This menu should contain variables which serve the parameterization of the device.

The menu name in English: "Parameters".

ObservationMenu (o)
The attribute 'menuId' references a menu from the MenuCollection. This menu should contain variables which serve the observation of the device (process data, dynamic variables, etc.).

The menu name in English: "Observation".

DiagnosisMenu (o)
The attribute 'menuId' references a menu from the MenuCollection. This menu should contain variables which serve the diagnosis of the device (events, etc.).

The menu name in English: "Diagnosis".

The English 'name' definition within this chapter is translated into common languages, and the text templates are delivered within this package. If a tool supports one of those common languages, it shall apply only those translations.

7.5.8.3 Menu collection

The names of top level menus, like IdentificationMenu, ParameterMenu, ObservationMenu or DiagnosisMenu are given from tooling. If a name is specified, it shall be ignored by tooling.

In underlying menus, a menu name shall be given by IODD.

MenuCollection (m)
All menu entries of the device are collected in the MenuCollection. These menu entries may be referenced by different roles (ObserverRole, MaintenanceRole, and SpecialistRole). There shall be no unreferenced Menu elements.

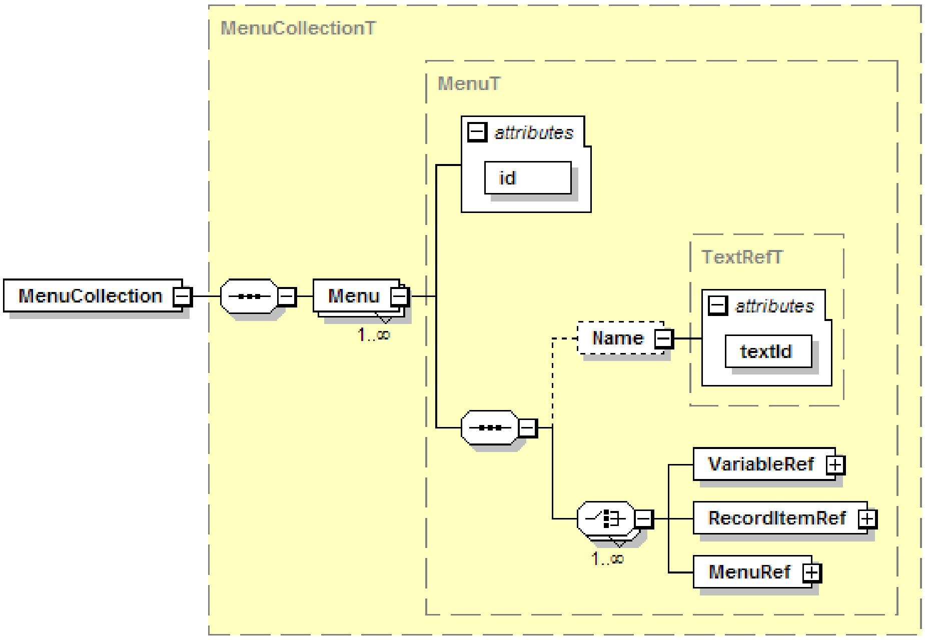


Figure 47 – MenuCollection element

Menu (m)
Variables, RecordItems and other menus may be referenced here.

id (m, IdT)
Explicit id of the menu.

Name (c)
textId (m, RefT)
Name of the menu. Top-level menus (i.e. those referenced from one of the MenuSets) may have a Name element, but it shall be ignored by IO-Link Tools. Instead, hard-coded names shall be used by the tools. Nested menus shall have a Name element which is shown by the tools.

7.5.8.4 VariableRef

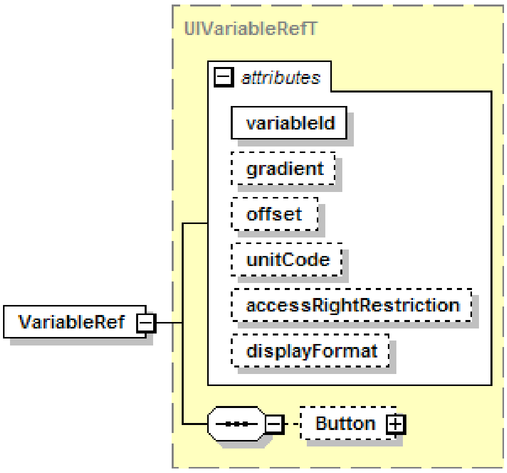


Figure 48 – VariableRef element

variableId (m, RefT)
Referenced variable

Regardless of the type of the referenced Variable or RecordItem, if gradient and / or offset are given, they shall be specified as floating point values.

Displayed value = (value read from the Device x gradient) + offset

When applying gradient and / or offset to convert the Variable or RecordItem value to the displayed value, the value will be implicitly converted to a floating point value. Consequently, the only allowed displayFormat on such values shall be “Dec”. (The displayFormat “Hex”, “Bin”, … does not force a conversion back to the original type of the Variable or RecordItem.)

When applying gradient and / or offset to convert an entered value back to the new value of a Variable or RecordItem, the resulting floating point value will be rounded to the nearest possible value of the type of the Variable or RecordItem.

Single array members can’t be referenced with RecordItemRef. If you need to access a single member, you have to define a record instead of an array.

A variable of type array can only be referenced as a whole, i.e. with VariableRef. All the elements and attributes in VariableRef (gradient, offset, unitCode, ‘accessRightRestriction’, displayFormat and Button) apply to each of the array members.

gradient (o, decimal)
Gradient of the indicated variables. The value shall not be zero. When offset is specified and gradient is not specified, a value of 1.0 shall be used.

offset (o, decimal)
Zero-offset of the indicated variables. When gradient is specified and offset is not specified, a value of 0.0 shall be used.

unitCode (o, positiveInteger)
Unit code to which the indicated variable refers. For valid unit codes see IODD-StandardUnitDefinitions1.1.xml.

unitCode shall only be used with datatypes UIntegerT, IntegerT and Float32T.

unitCode shall only be used without displayFormat or with displayFormat Dec and Dec.x.

accessRightRestriction (o, AccessRightsT)
For certain UserRoles, the access rights may be limited here.

The attribute ‘accessRightRestriction’ shall not be set to “wo”.

displayFormat (o, string with pattern)
Specifies how an IO-Link Tool shall display the value in the menu. The values of the attribute ‘displayFormat’ shall follow the regular expression pattern:

“Bin|Hex|Dec(\.\d)?”

Meaning of the values:

Bin: Binary notation with postfix “b”, e.g. 0101 1010 1010 0101b
Hex: Hexadecimal notation with postfix “h”, e.g. 5AA5h
Dec: Decimal notation without postfix, e.g. 23205
Dec.2: Decimal notation with given precision (number of digits after the decimal point) e.g. 23.00

The following table shows the valid combinations of the data type of the referenced Variable / RecordItem and the displayFormat, gradient and offset. Combinations not listed here shall not be used.

Table 2 – Allowed combinations of datatype, displayFormat, gradient and offset

datatype	allowed displayFormat	gradient and/or offset allowed	IO-Link Tool behaviour
BooleanT	Dec	No	Display as “0” for “false” and “1” for “true”.
	default	No	Display as “false” or “true”.
UIntegerT	Bin	No	Display as e.g. “0101 1010 1010 0101b”. Show 8, 16, 32 or 64 binary digits.
	Hex	No	Display as e.g. “5AA5h”. Show 2, 4, 8 or 16 hexadecimal digits.
	Dec	Yes	Without gradient and/or offset: Display as e.g. “23205”. Do not show leading zeroes. With gradient and/or offset: See Float32T, displayFormat=Dec
	Dec.x	Yes	Without gradient and/or offset: Display as e.g. “23205.00”. Do not show leading zeroes. With gradient and/or offset: See Float32T, displayFormat=Dec.x

	default	Yes	Same as Dec.
IntegerT	Bin	No	Display as e.g. “1111 1011 0010 1110b”. Show 8, 16, 32 or 64 binary digits. Show negative values as two’s complement.
	Hex	No	Display as e.g. “FB2Eh”. Show 2, 4, 8 or 16 hexadecimal digits. Show negative values as two’s complement.
	Dec	Yes	Without gradient and/or offset: Display as e.g. “-1234”. Do not show leading zeroes. With gradient and/or offset: See Float32T, displayFormat=Dec
	Dec.x	Yes	Without gradient and/or offset: Display as e.g. “-1234.00”. Do not show leading zeroes. With gradient and/or offset: See Float32T, displayFormat=Dec.x
	default	Yes	Same as Dec.
Float32T	Dec	Yes	Display digits after the decimal point as needed (up to an implementation-defined maximum).
	Dec.x	Yes	Display exactly “x” digits after the decimal point (also in exponential representation). Rounding shall be done with midpoint rounding away-from-zero (e.g. with “Dec.3” 23.3455 gets rounded to 23.346, and −23.3455 gets rounded to −23.346).
	default	Yes	Same as Dec.
StringT	*default*	No	Display just the string.
OctetStringT	*default*	No	Display as e.g. 0x00,0x56,0x78.
TimeT	*default*	No	Display as yyyy-mm-dd hh:mm:ss.fff where yyyy is the year, mm is the month, dd is the day, hh is the hour, mm is the minute, ss is the second and fff is the milliseconds.
TimeSpanT	*default*	No	Display as [+-][d]hh:mm:ss.fff where d is the days (optional, one or more digits), hh is the hour, mm is the minute, ss is the second and fff is the milliseconds.

ArrayT	Display all array elements. Button is not allowed. The allowed displayFormat, gradient/offset and unitCode is determined by the data type of the array elements.		
RecordT	*default*	No	Display all RecordItems in the order in which they appear in the Record definition, i.e. with ascending subindices, with their default display format. Button and / or unitCode are not allowed.

Some standard variables have complex types which are not modelled as special types in IODD because:

- The types cannot be used with other variables in a meaningful way.
- The types are difficult to describe in XML, increasing the complexity of the IODD.
- The types are IO-Link specific, complicating the general use of the IODD.

These standard variables could be displayed as bare numbers the way they are described in IODD-StandardDefinitions1.1.xml, but it is recommended that IO-Link Tools identify them by their name or index and display them specially, as described in the following table:

Table 3 – Standard variables with special display

Standard Variable / RecordItem	**Special display**
V_DirectParameters_1 (index 0), MasterCycleTime (subindex 2)	The octet consists of a Time Base in bits 7 to 6 and a Multiplier in bits 5 to 0. The IO-Link Tool shall calculate the time according chapter B.1.3 of the *IO-Link Interface and System* Specification Version 1.1.4 and display it as a decimal number with the unit milliseconds (ms).
V_DirectParameters_1 (index 0), MinCycleTime (subindex 3)	The octet consists of a Time Base in bits 7 to 6 and a Multiplier in bits 5 to 0. The IO-Link Tool shall calculate the time according chapter B.1.3 of the *IO-Link Interface and System* Specification Version 1.1.4 and display it as a decimal number with the unit milliseconds (ms).
V_DirectParameters_1 (index 0), M-sequence Capability (subindex 4)	The octet consists of a PREOPERATE M-sequence type in bits 5 to 4, an OPERATE M-sequence type in bits 3 to 1, and ISDU in bit 0. The IO-Link Tool shall decode this into text according to chapter B.1.4 of the *IO-Link Interface and System* Specification Version 1.1.4.
V_DirectParameters_1 (index 0), RevisionID (subindex 5)	The octet consists of a MajorRev in bits 7 to 4 and a MinorRev in bits 3 to 0. It shall be displayed as Vx.y, where x is the MajorRev and y is the MinorRev.
V_DirectParameters_1 (index 0), ProcessDataIn (subindex 6)	The octet consists of BYTE in bit 7, SIO in bit 6 and Length in bits 4 to 0. The IO-Link Tool shall display whether SIO is supported and the length in bits / octets according chapter B.1.6 of the *IO-Link Interface and System* Specification Version 1.1.4 .

V_DirectParameters_1 (index 0), ProcessDataOut (subindex 7)	The octet consists of BYTE in bit 7, SIO in bit 6 and Length in bits 4 to 0. The IO-Link Tool shall display whether SIO is supported and the length in bits / octets according chapter B.1.7 of the *IO-Link Interface and System* Specification Version 1.1.4 .
V_DetailedDeviceStatus (index 37)	Each array element shall be treated as an event data structure (see *IO-Link Interface and System* Specification Version 1.1.4, chapter A.6). It shall be decoded and displayed to text using the EventCollection in the IODD.
V_OffsetTime (index 48)	The octet consists of a Time Base in bits 7 to 6 and a Multiplier in bits 5 to 0. The IO-Link Tool shall calculate the time according chapter B.2.24 of the *IO-Link Interface and System* Specification Version 1.1.4 and display it as a decimal number with the unit milliseconds (ms).
V_DeviceAccessLocks (index 12)	V_DeviceAccessLocks shall only be referenced in menu via RecordItemRef to apply to the supported subindices, see chapter 7.5.8.5.

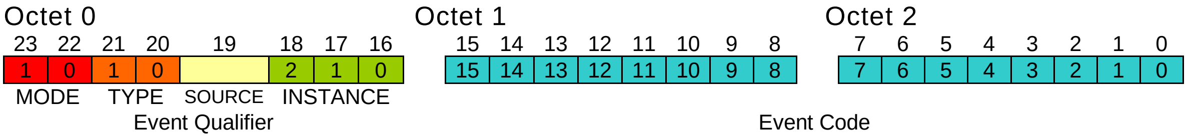


Figure 49 – Event data structure

Button (o)

Buttons are intended for implementing a command interface to the device. Several commands can be implemented on the same variable / record item using different values to be written.

If this element is given, the IO-Link Tool shall display a button instead of a value. The attributes ‘gradient’, ‘offset’, ‘unitCode’ and ‘displayFormat’ shall not be used when the element ‘Button’ is present.

The button shall be labelled with the Name that is given to the SingleValue at the data type of the referenced Variable or RecordItem whose ‘value’ corresponds to the ‘buttonValue’.

Use of this element is restricted to the data types BooleanT, UIntegerT and IntegerT, but it is highly recommended to use data type UIntegerT with bitLength=”8” Note that this does not include arrays of these data types.

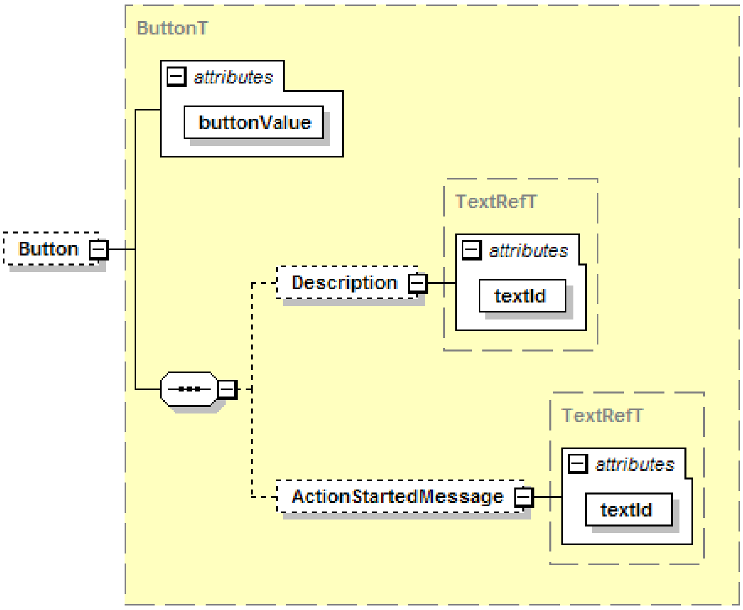


Figure 50 – Button element

A variable referenced as ‘Button’
shall have accessRights “wo”
shall only be displayed as a button
shall not be used as a condition variable, to switch menus or process data

NOTE: If a RecordItem is referenced as ‘Button’, the Record should be defined with subindexAccessSupported = “true”, otherwise correct tool functionality is not guaranteed.

The buttonValue
will be sent to the device immediately by pushing the button.
shall not be part of the block-download sequence.

buttonValue (m, union of boolean, unsignedLong and long)
This value shall correspond to a SingleValue/@value of the data type of the referenced Variable or RecordItem. It is sent to the device when the button is clicked.

Description (c)
textId (m, RefT)
A text that explains the action that will be started by pressing the button.

- For button values described as StdSingleValueRef, the Description Element is mandatory. The definition in template IODD-SystemCommandDefinitions_V114.xml shall be used.
- For button values described as SingleValue the Description Element is optional.

ActionStartedMessage (o)
textId(m, RefT)

A text that is shown after the button value was successfully sent to the device. Use this as a feedback to the user for actions that may take a while to complete or that require some user action to complete.

ActionStartedMessages shall be omitted for buttons referencing StdSingleValueRefs [@id=”128” or “129” or “130” or “131”].

7.5.8.5 RecordItemRef

Corresponds to VariableRef with an additional subindex. The variable referenced by variableId shall be of type record. If ‘Button’ is specified, the referenced variable shall support subindex access.

A RecordItemRef with variableId="V_DeviceAccessLocks" is only allowed for those subindices which refer to an access lock that is supported, i.e. where the respective attribute in Features/SupportedAccessLocks is set to "true".

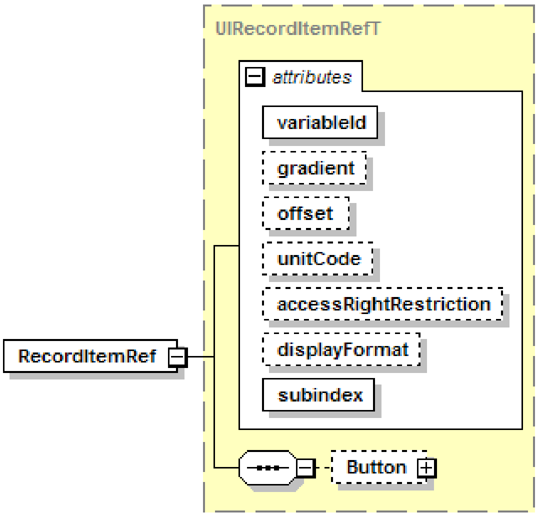


Figure 51 – RecordItemRef element

subindex (m, SubindexT)
Addresses the record item of a variable of type record.

For the other attributes and the element 'Button', see VariableRef above.

NOTE:

- If not all parts of a Record are referenced in an IODD menu, the tool functionality is only guaranteed when the Record can be accessed via subindices.
- If a Record contains mixed access rights, tool functionality is also not guaranteed, because access via subindex 0 is not forbidden
- Read-Modify-Write is not possible for IO-Link tools, because there might be other clients on the same IO-Link port

7.5.8.6 MenuRef

Reference to a (sub)menu nested inside this menu.

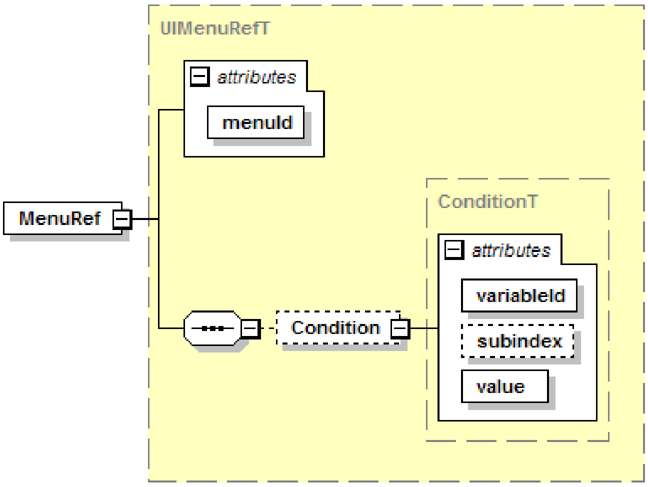


Figure 52 – MenuRef element

menuId (m, RefT)
References the (sub)menu from the MenuCollection.

Condition (o)
Condition for the display of this menu; an IO-Link Tool shall show the referenced menu only if the value of the referenced variable / record item equals the value of the attribute ‘value’.

variableId (m, RefT)
References a variable. The variable shall be of data type BooleanT, IntegerT, UIntegerT or RecordT. The variable shall have a default value if it is not of type RecordT.

subindex (c, SubindexT)
This attribute shall be given if and only if the referenced variable is of type RecordT. Used for addressing the record item within the record. The record item shall be of data type BooleanT, IntegerT or UIntegerT and shall have a default value.

value (m, unsignedByte)
Shall be a valid value for the variable / record item. This attribute can only hold values 0..255, thus limiting the possible IntegerT and UIntegerT values. Also, BooleanT condition values shall be entered as 0 for “false” and 1 for “true”.

Conditions may be used in all menu levels.

Condition variables shall appear as VariableRef or RecordItemRef at least in a read-only way in a menu which is referenced in the same user role.

If there is more than one ProcessData element, selected by conditions, and the variable V_ProcessDataInput or V_ProcessDataOutput is referenced in a menu, one of the following shall hold:

- The type of reference (VariableRef / RecordItemRef) and the gradient, offset, unitCode and displayFormat fit to each of the ProcessData elements.
- The menu is conditioned in the same way as one of the ProcessData elements, and the type of reference (VariableRef / RecordItemRef) and the gradient, offset, unitCode and displayFormat fit to this particular ProcessData element.

“Conditioned in the same way” means that this or one of the parent menus has the same condition (same variable, same subindex, same value).

7.5.9 Rules for write-only variables

For variables with accessRights=“wo” (write-only) the following rules shall be considered. If not accompanied with attribute displayFormat=“button”:

- “wo”- Variables generally are handled as commands
- “wo”- Variables are edited like any other variable
- “wo”- Variables shall never be part of any download sequence
- “wo”- Variables shall always be handled as a single write request

7.6 Communication characteristics

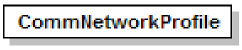


Figure 53 – CommNetworkProfile element

Excursion on XML schema *abstract types*:
An abstract type can’t be used itself. Only non-abstract types which are derived from an abstract type can be used. The instance selects the desired derived type with xsi:type=”*name of the derived type*”.

This technique is used here with the ‘CommNetworkProfile’ element to adapt the XML structure to the requirements of the specific communication. This allows easy extension of the IODD to non-IO-Link devices with different communication characteristics as long as the applicative concept remains the same (i.e. addressing via index/subindex, standardized variables).

7.6.1 IOLinkCommNetworkProfileT

For IO-Link, the following derived type IOLinkCommNetworkProfileT describes the communication characteristics of an IO-Link interface.

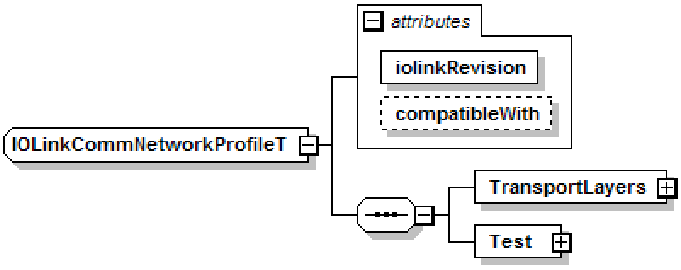


Figure 54 – CommNetworkProfile element – IO-Link variant

iolinkRevision (m, VersionT)
Implemented protocol version. Fixed to “V1.1”.

compatibleWith (o, VersionT restricted to “V1.0”)
Specify this attribute if the device is compatible with IO-Link revision 1.0, i.e. also runs on a V1.0 IO-Link Master. This requires an IODD V1.0.1 (or V1.0).

TransportLayers (m)

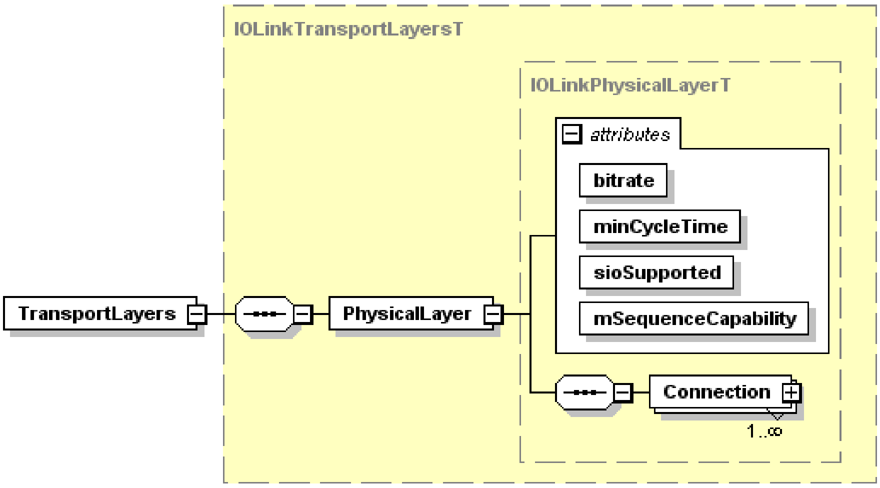


Figure 55 – TransportLayers element – IO-Link variant

PhysicalLayer (m)

bitrate (m, string)
Allowed values are “COM1”, “COM2” or “COM3”.

minCycleTime (m, unsignedInt)
The minimum cycle time of the slave; specified in 1 microsecond (µs) units. E.g. the value 2300 represents 2.3 milliseconds (ms). The allowed value range is 0..6300 in steps of 100, 6400..31600 in steps of 400 and 32000..132800 in steps of 1600 (see *IO-Link Interface and System* Specification Version 1.1.4, chapter B.1.3).

sioSupported (m, boolean)
Whether the fall-back to SIO mode is supported.

mSequenceCapability (m, unsignedByte)
Enter the content of V_DirectParameters_1, subindex 4 (M-sequence Capability) here as a decimal number.

Test (m)
Contains information to enable automatic testing of the device.

Enter appropriate data for the ISDU and event test configurations (see IO-Link Test Specification Version 1.1.4).

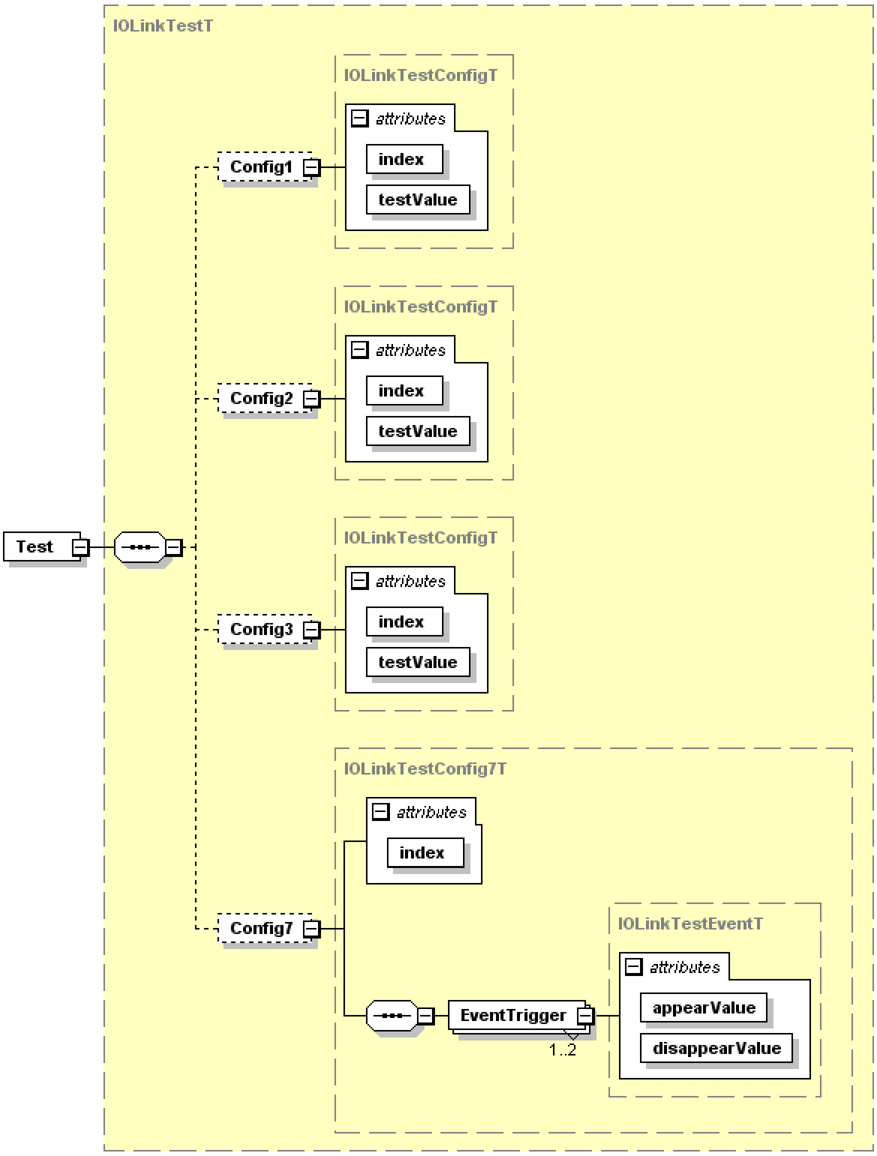


Figure 56 – Test element

Config1 (c)
Shall be present if the device supports ISDU access, and refers to a read-write 8 bit index. The testValue shall be small enough (<=12 octets) so that the ExtLength coding of the ISDU is not used.

Config2 (c)
Shall be present if the device supports ISDU access, and refers to a read-write 16 bit index. If the device supports a read-write variable on such an index, the testValue shall be accepted by the device. If the device does not support such an index a valid ‘ErrorType’ shall be returned (Index not available).

Config3 (c)

Shall be present if the device supports ISDU access, and refers to a read-write 8 bit index. The testValue shall have a length big enough (>12 octets) to cause the ExtLength coding of the ISDU.

Config7 (c)

Shall refer to an index raising different types of events. For details, please refer to the IO-Link Test Specification Version 1.1.4.

index (m, RefT)

References an Index to be used for testing.

testValue (m, string with pattern: “(0x[0-9A-Fa-f][0-9A-Fa-f],)*0x[0-9A-Fa-f][0-9A-Fa-f]”)

Shall be an acceptable octet string value for the index.

EventTrigger (m)

Trigger values for up to two events.

appearValue (m, unsignedByte)

The value that triggers an event when written to Config7/@index.

disappearValue (m, unsignedByte)

The value that quenches the event triggered by @appearValue when written to Config7/@index.

Connection (m)

Describes, how the device can be connected. A Connection may be used for multiple device variants, which are referenced by the ProductRef/@productId attributes.

This element has the following XML abstract type:

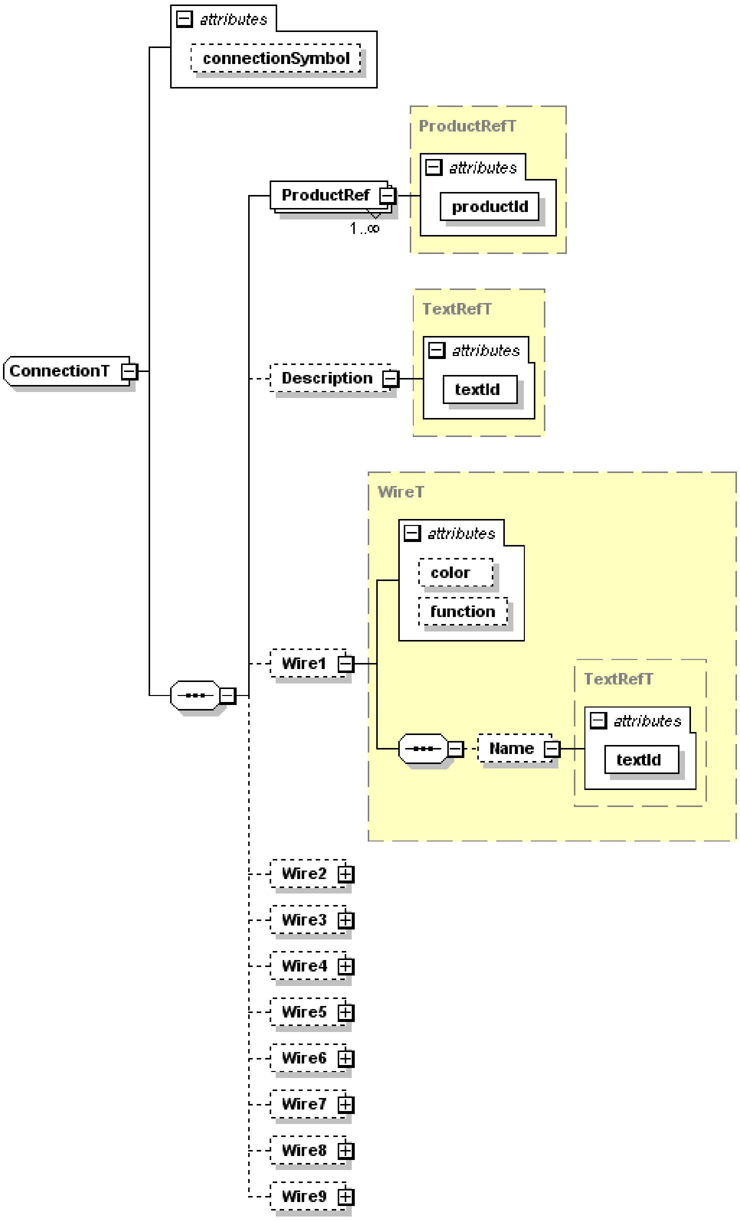


Figure 57 – ConnectionT abstract type

connectionSymbol (o, string with pattern “([\p{L}\d_#]+-)+con-pic\.png”)
File name of the connection symbol. If this attribute is used, the referenced image file shall be present.

ProductRef (m)
productId (m, string)
Selects the device variants that use this Connection. Shall correspond to one of the DeviceIdentity/DeviceVariantCollection/DeviceVariant/@productId values.

Description (o)
textId (m, RefT)
Description of the connection.

Wire<X>

Describes one of the wires. If the connection is some type of connector, the number <X> also designates the pin / hole number.

For OtherConnectionT and CableConnectionT, if the number of wires exceeds 9, the wires relevant for IO-Link shall be described. In this case, the <X> does not necessarily equal the pin / hole number. The real pin number should be described in the Wire<X>/Name element.

color (o, string)

A color code according to IEC 60757.

Table 4 – Wire colors

Code	Color
“BK”	Black
“BN”	Brown
“RD”	Red
“OG”	Orange
“YE”	Yellow
“GN”	Green
“BU”	Blue (including light blue)
“VT”	Violet (purple)
“GY”	Grey (slate)
“WH”	White
“PK”	Pink
“GD”	Gold
“TQ”	Turquoise
“SR”	Silver

function (o, string)

The function of the wire.

Table 5 – Wire functions

Function	Description
“NC”	Not connected
“L+”	Power supply (+), pin 1, brown
“L-”	Power supply (-), pin 3, blue
“P24”	Extra power supply (+)
“N24”	Extra power supply (-)
“Other”	e.g. signal (DI, DO, analog) or power supply
“C/Q”	Communication signal, pin 4, black

Name (o)

textId (m, RefT)

An additional textual description of the wire. Do not repeat the wire color or wire function in textual form here.

The allowed XML derived types are: M5ConnectionT, M8ConnectionT, M12-4ConnectionT, M12-5ConnectionT, OtherConnectionT and CableConnectionT. They restrict the abstract type accordingly.

M5ConnectionT

The M5 connector as specified in *IO-Link Interface and System* Specification Version 1.1.4, chapter 5.5.

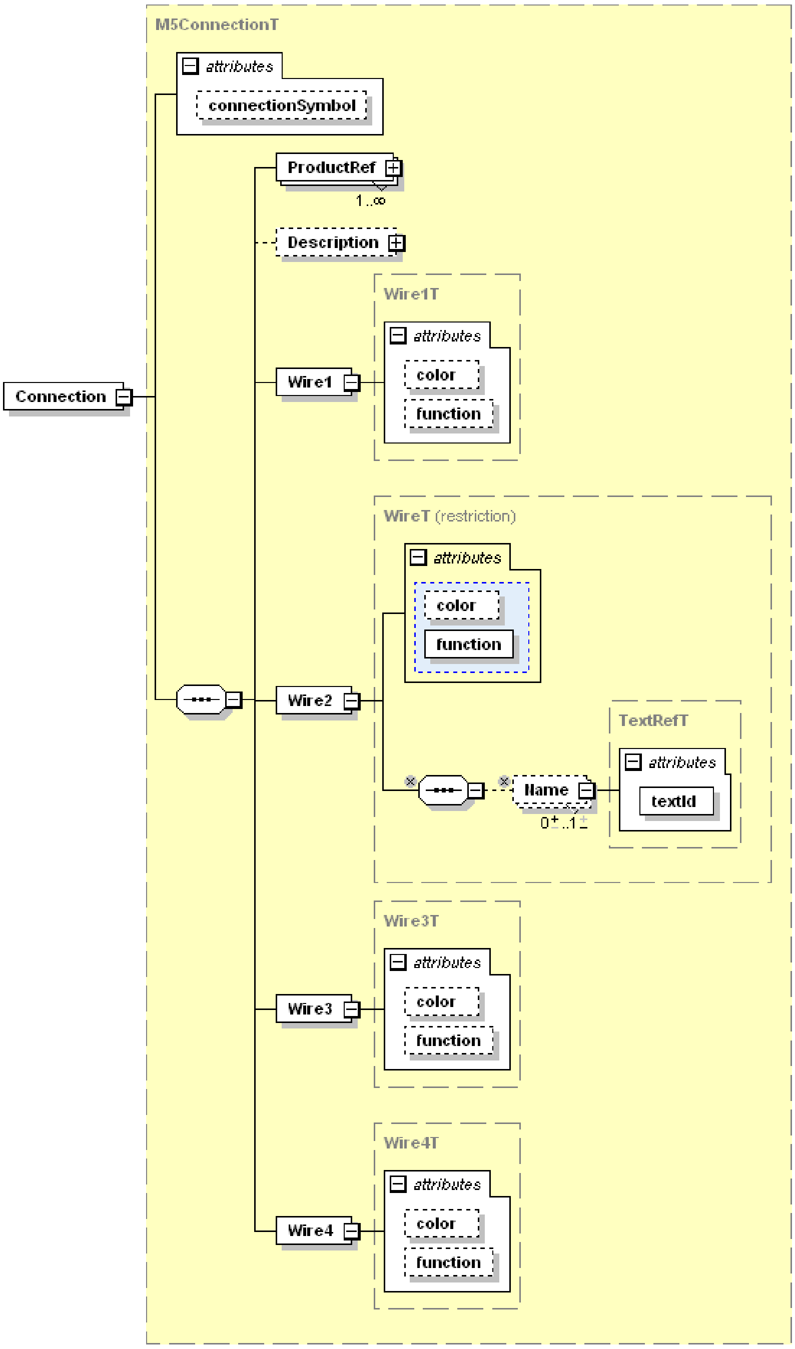


Figure 58 – Connection element – M5ConnectionT variant

Wire1, Wire3 and Wire4 have fixed color and function. No Name is allowed.

Wire 2 has a fixed color “WH” (white), and a function restricted to “NC” or “Other”. The function attribute is mandatory.

M8ConnectionT and M12-4ConnectionT

Same as M5ConnectionT.

M12-5ConnectionT

The M12-5 connector as specified in *IO-Link Interface and System* Specification Version 1.1.4, chapter 5.5.

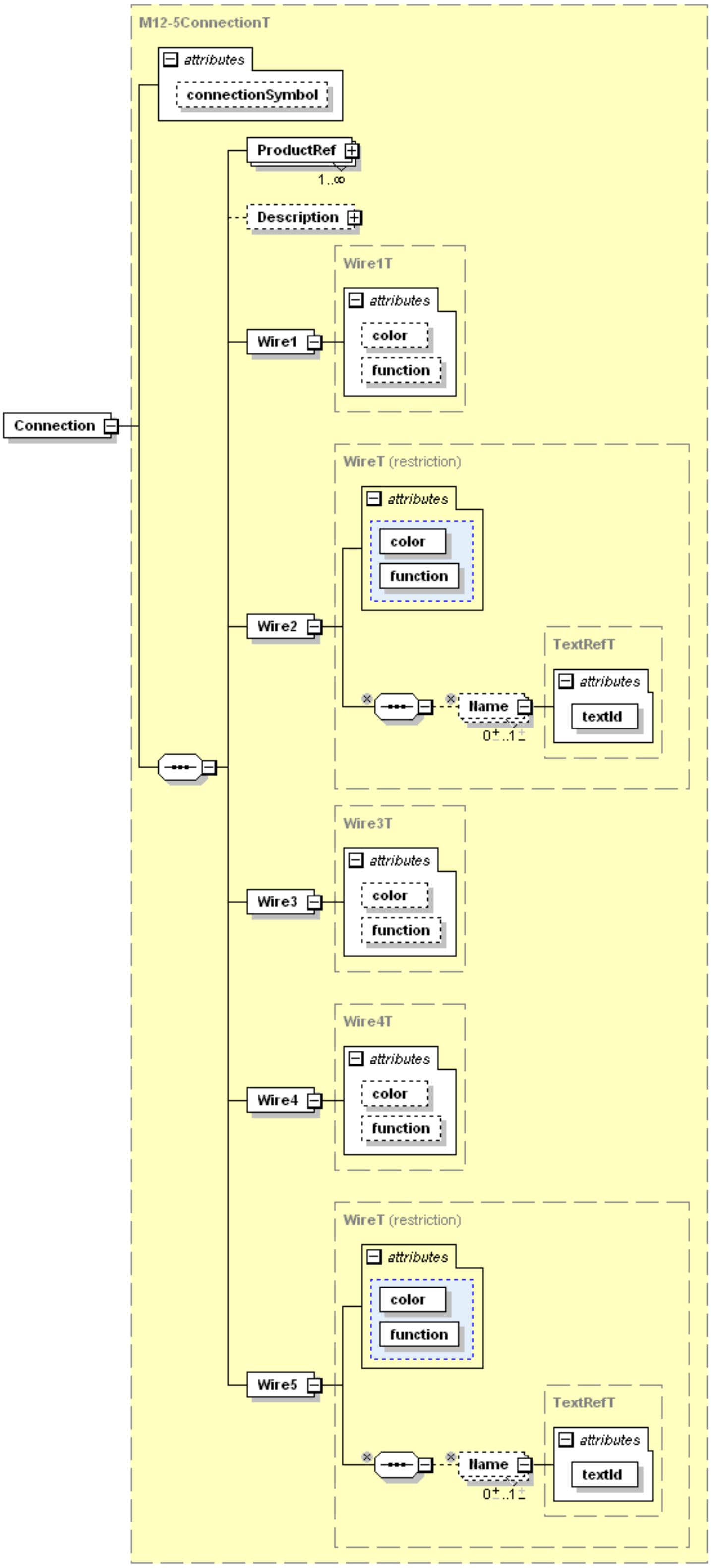


Figure 59 – Connection element – M12-5ConnectionT variant

Wire1, Wire3 and Wire4 have fixed color and function. No Name is allowed.

Wire 2 has a function restricted to “NC”, “P24” or “Other”. If its function is I/Q, its color shall be “WH” (white). The color and function attributes are mandatory.

Wire 5 has a function restricted to “NC” or “N24”. The color and function attributes are mandatory.

OtherConnectionT
Some non-standard connector.

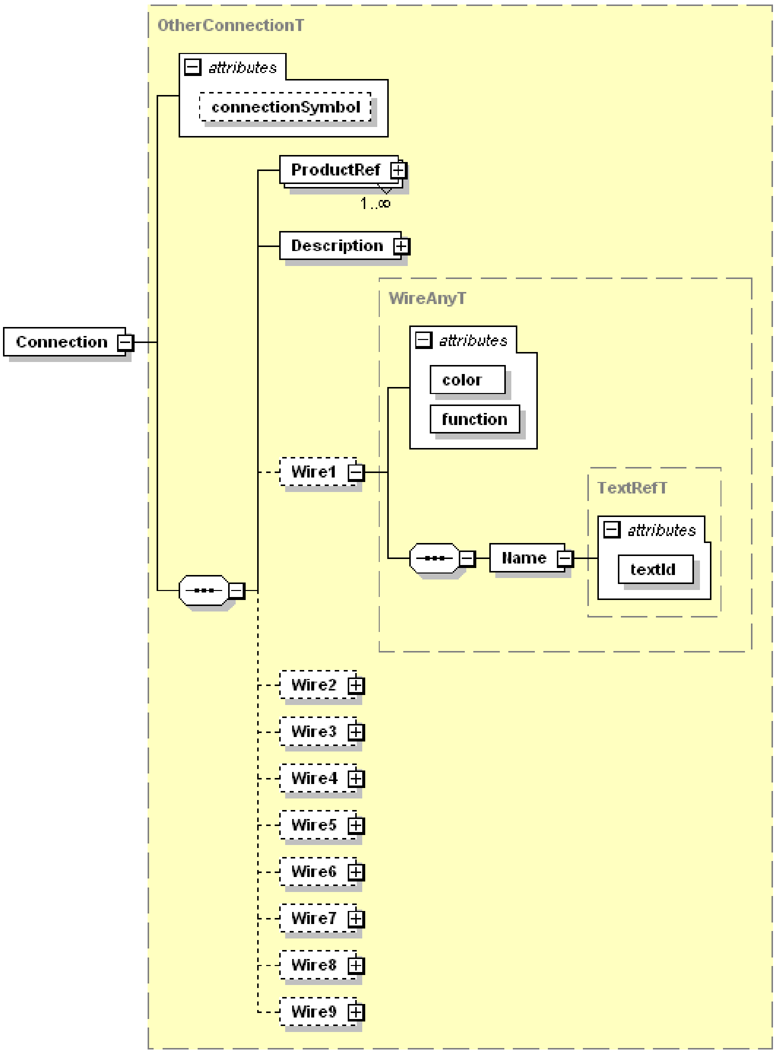


Figure 60 – Connection element – OtherConnectionT variant

The Description is mandatory. For all given Wires, the element ‘Name’ and the attributes ‘color’ and ‘function’ are mandatory.

Wires with functions “L+”, “L-” and “C/Q” shall be present.

CableConnectionT
Same as OtherConnectionT, but Description is not mandatory. The wire number in the ‘Wire…’ element name does not designate a pin number here, but any arbitrary numbering of the wires.

7.6.2 IOLinkWirelessCommNetworkProfileT

For Wireless IO-Link, the following derived type IOLinkWirelessCommNetworkProfileT describes the communication characteristics of a wireless IO-Link interface. Please see the *IO-Link Wireless System Extensions Specification* for details of this communication network profile.

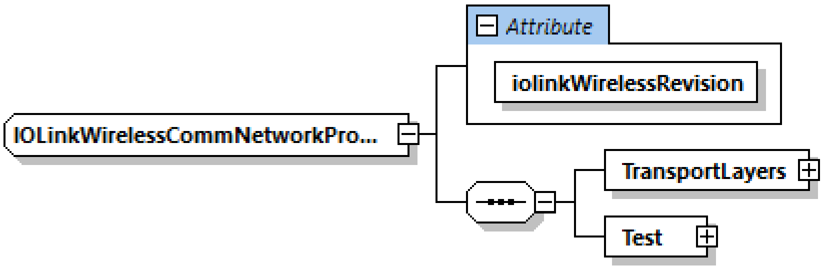


Figure 61 – CommNetworkProfile element – IO-Link Wireless variant

7.7 Language dependent description texts

All text components of the different languages are given in the ExternalTextCollection. There may be one or more languages deposited. Additional languages may be stored in separate files.

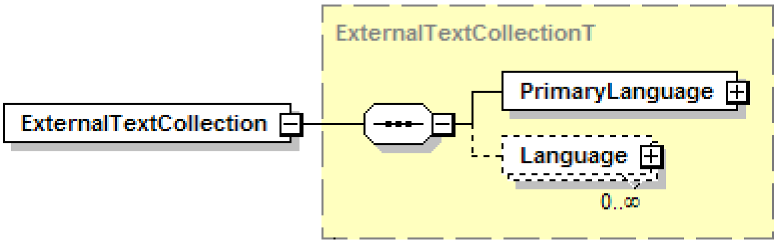


Figure 62 – ExternalTextCollection element

7.7.1 PrimaryLanguage (m)

Shall be in English.

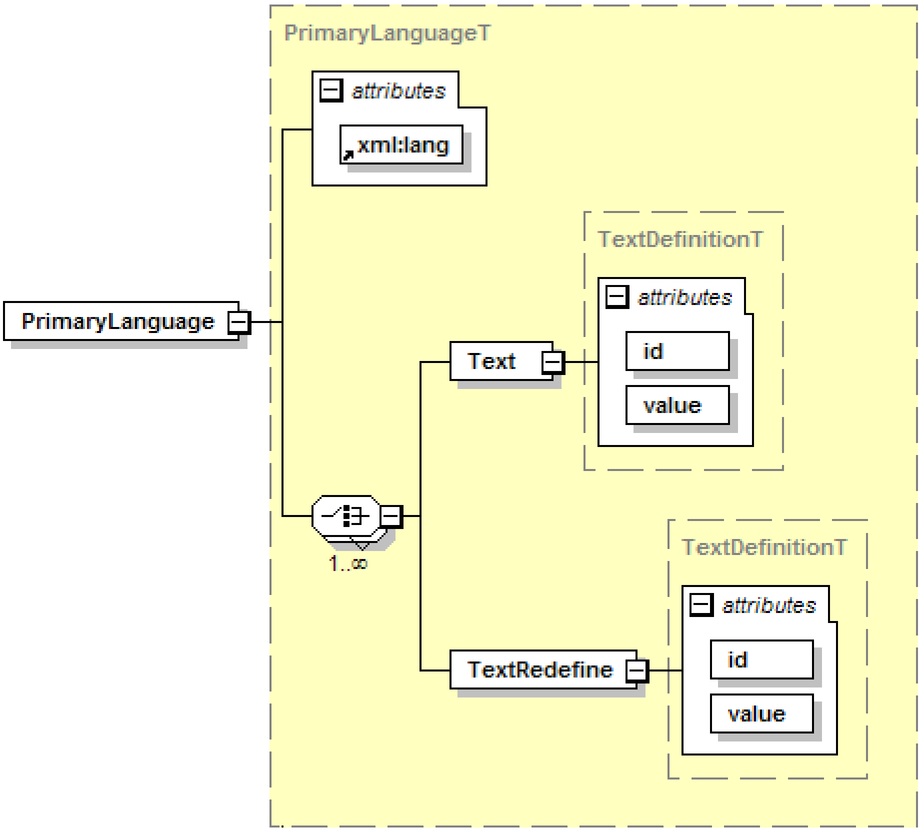


Figure 63 – PrimaryLanguage element

xml:lang (m, language)
The code for the language according to ISO 639:2023. Shall be “en” for English.

Text (m)
Language dependent text which is referenced by its id.

id (m, IdT)
Shall be referenced by other elements via their ‘textId’ attribute (there shall be no unreferenced Text elements)

value (m, string)
Text in the denoted language.

NOTE: Special characters shall be coded according to the XML syntax. See REC-xml-20081126, chapter 2.4 Character Data and Markup.

&	→	&amp;	
‘	→	&apos;	(only required if inside a string enclosed in ‘ characters)
>	→	&gt;	
<	→	&lt;	
“	→	&quot;	(only required if inside a string enclosed in “ characters)
LF	→	&#10;	

Only the line feed is allowed for formatting the text.

TextRedefine (o)
Language dependent text which overrides a standard text. Only applicable for texts describing the octets of DirectParameter page 2.

id (m, IdT)
Shall be one of STD_TN_DeviceSpecific_1 to STD_TN_DeviceSpecific_16.

value (m, string)
Text in the denoted language. The Note at Text/@value also applies.

7.7.2 Language (o)

Optional specification of texts in another language. The attribute ‘xml:lang’ specifies the language (see ISO 639:2023). The structure of this element corresponds to the structure of the element PrimaryLanguage.

8 Compatibility

IO-Link devices conforming to the *IO-Link Interface and System* Specification Version 1.1.4. shall be described with an IODD according to this IO Device Description Specification Version 1.1.4.

IO-Link devices conforming to the IO-Link Community, IO-Link Communication Specification Version 1.0 shall be described with an IODD according to the IO-Link Community, IO-Link Device Description Specification Version 1.0.1 .

It is not possible to describe a V1.0 device with an IODD V1.1 or a V1.1 device with an IODD V1.0.1 or V1.0.

Two IODDs having the same vendorId and deviceId, one being based on V1.0.1 (or V1.0) and the other being based on V1.1 are only allowed in the following use cases:

- A V1.0 device exists and has a V1.0.1 (or V1.0) IODD. A new, compatible version of the device is built using the same vendorId and deviceId based on V1.1 needing a V1.1 IODD.

- A V1.1 device is newly built with a new vendorId /deviceId pair. It is also V1.0 compatible and thus needs two IODDs.

In both cases, the DeviceFamily and all DeviceVariant/Name (previously ProductName) for all languages, as well as all DeviceVariant/@productId shall be same.

An IO-Link tool may show V1.1 as well as V1.0.1 (or V1.0) IODDs in its catalog. According compatibility feasibility of the IO-Link master, the appropriate IODD may be instantiated.

Annex A IODD schemas

The following schemas and standard definition files are part of this specification:

Schema files

• IODD1.1.xsd	main IODD schema
• IODD-Primitives1.1.xsd	basic definitions
• IODD-Datatypes1.1.xsd	data types
• IODD-Events1.1.xsd	events
• IODD-Variables1.1.xsd	variables
• IODD-UserInterface1.1.xsd	user interface
• IODD-Communication1.1.xsd	communication network profile
• IODD-StandardDefinitions1.1.xsd	main schema for the standard definition files
• IODD-WirelessCommunication1.1.xsd	wireless communication network profile

Standard definition files

• IODD-StandardDefinitions1.1.xml	list of standard variables, error types and events + english texts
• IODD-StandardDefinitions1.1-de.xml	german texts
• IODD-StandardDefinitions1.1-es.xml	spanish texts
• IODD-StandardDefinitions1.1-fr.xml	french texts
• IODD-StandardDefinitions1.1-it.xml	italian texts
• IODD-StandardDefinitions1.1-ja.xml	japanese texts
• IODD-StandardDefinitions1.1-ko.xml	korean texts
• IODD-StandardDefinitions1.1-pt.xml	portuguese texts
• IODD-StandardDefinitions1.1-ru.xml	russian texts
• IODD-StandardDefinitions1.1-zh.xml	chinese texts
• IODD-StandardUnitDefinitions1.1.xml	list of available unit codes + english texts

Annex B Definitions of IODD quantity structure

The following table lists limits on the number and the length of elements of the IODD. IODDs shall not exceed these limits. IO-Link Tools shall accept all IODDs that do not exceed these limits.

Table 6 – IODD quantity structure

Element	Maximum Number/Length	Comments
Length of filename	255	
Number of DeviceVariants	255	
Number of ValueRanges	32	See note
Number of SingleValues	255	See note
Number of ProcessData Elements	16	
Number of Datatypes	255	See note
Number of Variables	1024	
Length of text which is referenced by DeviceVariant/Description/@textId in all languages	1024	Characters, not octets
For all others Description/@textId: Length of text which is referenced by Description/@textId in all languages	255	Characters, not octets
For all Name/@textId: Length of text which is referenced by Name/@textId in all languages	64	Characters, not octets
Length of URL which is referenced by VendorUrl/@textId in all languages	255	Characters, not octets
Length of @vendorName and all other texts referenced by a @textId (e.g. VendorText, DeviceName, DeviceFamily, …)	64	Characters, not octets
Length of @productId value	64	Octets UTF-8 coded
Length of DocumentInfo/@copyright text	255	Characters, not octets
Number of Menus in MenuCollection	1023	
Number of elements per menu (VariableRef+RecordItemRef+MenuRef)	64	
Variable/@id length	64	Characters, not octets
Datatype/@id length	64	Characters, not octets
@textId length	64	Characters, not octets
Number of supported languages	see ISO 639:2023	Currently 185
Number of Menu Levels	3	

Note: The entries “Number of <element name>” are not meant as the total number of XML elements with that name in the whole IODD. Instead, they are meant as “Number of XML elements of that name within the superordinated element”.

“Number of Datatypes” is the number of Datatype elements within the DatatypeCollection element. This does not constitute the total number of data types which may appear in an IODD, because each Variable and ProcessDataIn/ProcessDataOut element may additionally use an embedded data type.

“Number of Variables” is the sum over all StdVariableRef, DirectParameterOverlay and Variable elements within the VariableCollection.

Annex C

IODD test (normative)

1.1 General

An IODD shall conform to the underlying schema files and a set of business rules, defined in this specification.

1.2 Schema test via an IODD Checker tool

The IO-Link Community provides the IODD Checker for download free of charge from its web site. It is mandatory for each IODD associated with a Device to pass the test with this Checker. The Device’s manufacturer declaration shall state the successful result of the test.

The latest released version of the IODD Checker shall be applied, when new device Id has been specified.

The requirements for the Checker consist of two main parts.

Within the first part the Checker tests the schema consistency of a particular IODD. Within the second part the Checker uses the following business rule set in Table 7. For more detailed information see the IODD Checker’s user manual.

Column ‘Reference’ indicates either the chapter within this document or with prefix [1] the chapter within the *IO-Link Interface and System* Specification Version 1.1.4.

An “X” in column “M” indicates relevance for the main IODD

An “X” in column “L” indicates relevance for the language file.

An “X” in column “W” indicates that this business rule check creates a warning.

An “X” in column “H” indicates ‘highly recommendation’. Checker creates a warning, which cannot be disabled or made invisible.

An “X” in column “V” indicates validity of the check for legacy Devices (V1.0).

Table 7 – Checker business rule set for IODDs

IODD_TC No	Rule name	Reference	Definition	M	L	W	H	V
0001	Encoding	5	Encoding shall be UTF-8	X	X			X
0002	NameSpace	5	http://www.w3.org/2001/XMLSchema-instance with the prefix “xsi” http://www.io-link.com/IODD/2010/10 with the prefix “iodd”	X	X			
0003	AdditionalNameSpaces	5	No additional name spaces shall be included	X	X			X
0004	SchemaLocation	5	http://www.io-link.com/IODD/2010/10	X	X			
0005	SchemaMainIODD	5	IODD1.1.xsd	X				
0006	SchemaLanguageFile	5	IODD-Primitives1.1.xsd		X			
0007	FileNameConvention - IODD SpecialCharacters	5	Special characters are permitted in vendor name and device name part of filename: _, #, -	X	X			X
0008	FileNameConvention IODD-VendorNamePart	5	All files of the set of files belonging to a specific IODD shall have the same <vendor name> part in their file names.	X	X			X

IODD_TC No	Rule name	Reference	Definition	M	L	W	H	V
0009	FileNameConvention MainIODD	5.1	<vendor name>-<device name>-<date of file creation>-IODD<schema version>.xml; IODD shall be done with upper case letters	X				X
0010	FileNameConvention LanguageFile	5.2	<vendor name>-<device name>-<date of file creation>-IODD<schema version>-<language>.xml; IODD shall be done with upper case letter		X			X
0011	LanguagePart - LanguageFileName-ISO	5.2	The “language” part follows ISO 639:2023.		X	X		X
0012	LanguagePart – LanguageFileName-2Letters	5.2	The “language” part consists of two letters		X			X
0013	LanguagePart – LanguageFileName-Unique	5.2	There shall be no additional language file for languages already covered in the main IODD file		X			X
0014	LanguagePart – LanguageFileName-Inside	5.2	The language part of the language file name shall be the same as the definition inside the language file		X			X
0015	LanguageStandard-Definitions	5.2	If an IODD contains a language, which is not existing for IODD-StandardDefinitions, the checker will show a warning	X	X	X		
0016	VendorLogo	5.3	160 x 90 pixel, landscape format	X				X
0017	DeviceIcon	5.3	48 x 48 pixel	X				X
0018	DevicePicture	5.3	Min. 160 x 160 pixel, max. 320 x 320, square	X				X
0019	ConnectionSymbol	5.3	Min. 160 x 160 pixel, max. 320 x 320, square	X				
0020	ImageFilesExist	5.3	If the attributes are used, the referenced image files shall be present	X				X
0021	ImageFileNameConvention	5	All files of the set of files belonging to a specific IODD shall have the same <vendor name> part in their file names	X				X
0022	DateIODD - FileName	7.3.1	The date information in the IODD file name shall correspond to the releaseDate attribute in the DocumentInfo element	X				X
0023	DateLanguage - FileName	7.3.1	The date information in the language file name shall correspond to the releaseDate attribute in the DocumentInfo element, if the DocumentInfo element exists		X			X
0024	ProfileHeader	7.3.2	It shall correspond exactly to the given values in the specification	X				X
0025	Stamp	7.3.4	If no errors are detected during the checking process, the crc attribute is set to a CRC value calculated across the file contents. Otherwise, the crc attribute is set to an invalid value.	X	X			X
0026	Comments - InOr-AfterStamp	7.3.4	Comments shall not be included in or after the Stamp element.	X	X			X
0027	ProductId	7.4.1	ProductID in IODD corresponds to the ISDU standard parameter. Multiple device variants are only allowed, if StdVariableRef with id=”V_ProductID” is present.	X				X
0028	ProductId-Length	7.4.1	The maximum length of ProductId shall not exceed 64 octets UTF-8 coded.	X				X
0029	Declarations-Data-types	7.5.2	There shall be no unreferenced data type elements.	X				X

IODD_TC No	Rule name	Reference	Definition	M	L	W	H	V
0030	DatatypeId	7.5.2	For data types in the DatatypeCollection, the attribute id shall be specified.	X				X
0031	NoDatatypeId	7.5.2	For Datatypes outside the DatatypeCollection, the attribute id shall not be specified	X				X
0032	StdVariableRef	7.5.4.1	V_DirectParameters_1 shall always be referenced. All standard ISDU variables marked with the attribute mandatory="true" in the IODD-StandardDefinitions shall be referenced, if the Device supports ISDU access.	X				X
0033	V_DirectParameters_2 and DirectParameterOverlay	7.5.4.1	StdVariableRef with id="V_DirectParameters_2" and DirectParameterOverlay may only be present both or none.	X				
0034	DirectParameterOverlay reference in menu	7.5.4.1	If StdVariableRef with id="V_DirectParameters_2" and DirectParameterOverlay are present, V_ DirectParameters_2 shall not be referenced and DirectParameterOverlay shall be referenced in menu.	X				
0035	StdVariableRef - ReservedIds	7.5.4.2 7.5.4.3 [1], B.2.1	The id of any standard variable shall not be used as id for the Variable or DirectParameterOverlay, even if the standard variable is not referenced from this IODD.	X				X
0036	StdVariableRef - DefaultValue	7.5.4.1	@defaultValue shall not be specified for references to V_ProcessDataInput or V_ProcessDataOutput	X				X
0037	FixedLength-Restriction	7.5.4.1	If referenced variable is of type OctetString or String, it shall be less or equal its fixedLength attribute. If referenced variable is of type Array, it shall be less or equal to its 'count' attribute.	X				X
0038	V_Application-SpecificTag	[1], B.2.16	If fixedLengthRestriction is used, it shall be equal or greater than 16.	X				X
0039	StdSingleValueRef	7.5.4.1	Check if StdSingleValue exists as SingleValue in StdDefinitions1.1.xml.	X				X
0040	StdValueRangeRef	7.5.4.1	Check if StdValueRange exists as ValueRange in StdDefinitions1.1.xml.	X				X
0041	StdVariableRef - SingleValue	7.5.4.1	Check against overlapping with StdSingleValue and StdValueRange (even if they are optional and not referenced), other SingleValue or ValueRange.	X				X
0042	StdVariableRef - ValueRange	7.5.4.1	Check against overlapping with StdSingleValue and StdValueRange (even if they are optional and not referenced), other SingleValue or ValueRange.	X				X
0043	StdRecordItemRef	7.5.4.1	Check if referenced RecordItem exists in StdDefinitions1.1.xml.	X				X
0044	DirectParameter-Overlay - Subindex-AccessSupported	7.5.3.2.3 [1], B.1.1	Statement subindexAccessSupported="false" shall not be used. Index 1 can only be accessed octet by octet.	X				
0045	DirectParameter-Overlay - Datatype	7.5.4.2	The data type shall be a record.	X				
0046	DirectParameter-Overlay - Usage	7.5.4.1	It is highly recommended not to use V_DirectParameters_2 and ISDU in the same Device.	X		X	X	

IODD_TC No	Rule name	Reference	Definition	M	L	W	H	V
0047	VariableIndex	[1], B.2.1	Preferred and extended Index for Device or vendor specific variables are 64 to 254 and 256 to 16383. Indices 2 to 63, 255, 16384 to 65535 shall not be used as Device or vendor specific variable.	X				X
0048	VariableIndex - Profiles and extensions	[1], B.2.1	If indices from ranges reserved for profiles and extensions are used, the checker shall generate a warning.	X		X		X
0049	RecordItemInfo	7.5.4.3	RecordItem shall exist.	X				X
0050	RecordItemInfo - NoRecord	7.5.4.3	Only applicable if the variable is of type record.	X				X
0051	DefaultValue	7.5.4.1 7.5.4.3	The default value shall match the given datatype.	X				X
0052	DefaultValue - String	7.5.4.1 7.5.4.3	Check whether used letters are valid in respect to given encoding.	X				X
0053	ProcessData - Condition	7.5.5	If ProcessData occur more than once, the individual ProcessData elements shall be distinguished by the Condition element.	X				X
0054	ProcessData - bitLength	7.5.5	The attribute 'bitLength' shall represent the underlying ProcesssDataIn (-Out) datatype in a bit by bit manner. For record data types this 'bitLength' shall equal the 'bitLength' attribute of the record.	X				X
0055	ProcessData - ConditionVariable	7.5.5	There shall only be exactly one variable used for the switching of process data. The referenced variable shall contain a default value. The process data length (of ProcessDataIn and ProcessDataOut respectively) shall be the same for all ProcessData.	X				X
0056	ProcessData - ConditionDatatype	7.5.5	Conditions shall only be of datatype IntegerT, UIntegerT and BooleanT.	X				X
0057	ProcessData - Condition – Menu	7.5.5	All Variable / RecordItem referenced in the ProcessData/Condition elements shall be referenced from at least one menu within each user role.	X				
0058	StdErrorTypeRef	7.5.6	Check whether referenced 'ErrorType' exists in StdDefinitions1.1.xml.	X				
0059	StdEventRef	7.5.7	Check whether referenced Event exists in StdDefinitions1.1.xml.	X				
0060	EventCode	[1], D.2	Vendor or device specific codes are: 0x1800-0x18FF and 0x8CA0-0x8DFF.	X				
0061	EventCode – Profiles and extensions	[1], D.2	If EventCodes from ranges reserved for profiles and extensions are used, the checker shall generate a warning.	X		X		
0062	ProcessDataInfo	7.5.8.1	Check whether displayFormat and Datatype are matching.	X				
0063	MenuLevel	7.5.8.2	At most three menu levels below the role assignment are acceptable.	X				X
0064	NotUsedMenus	7.5.8.3	There shall be no unreferenced Menu elements	X				
0065	MenuName	7.5.8.3	For underlying menus, a menu name shall be specified.	X				X

IODD_TC No	Rule name	Reference	Definition	M	L	W	H	V
0066	GradientOffset	7.5.8.4	When applying gradient and/or offset to convert the Variable or RecordItem value into the displayed value, the value will be implicitely converted to a floating-point value. Consequently, the only allowed displayFormat for such values shall be “Dec” or “Dec.*”.	X				X
0067	RecordItemRef - Array	7.5.8.4	Single array members cannot be referenced by RecordItemRef.	X				X
0068	UnitCode	7.5.8.4	Unit code to which the indicated variable refers. See IODD-StandardUnitDefinitions1.1.xml for valid unit codes.	X				X
0069	Menu - AccessRightRestriction	7.5.8.4	The accessRight of the referenced element shall include the accessRights given by accessRightRestriction!.	X				X
0070	ButtonValue	7.5.8.4	The value of the ‘buttonValue’ attribute shall be defined as a ‘SingleValue’ of the Variable/-RecordItem	X				X
0071	ButtonValue - Datatype	7.5.8.4	It is highly recommended to use data type UIntegerT with bitLength=“8”	X		X	X	
0072	ButtonReference	7.5.8.4	A variable referenced as “Button” shall have accessRights “wo”, shall only be displayed as a button, and shall not be used as a condition variable to switch menus or processdata.	X				X
0073	RecordItemRef	7.5.8.5	The variable referenced by variableId shall be of type record. The subindex shall be defined in the referenced record.	X				X
0074	RecordItemRef - Button	7.5.8.5	If referenced as “Button”, the referenced variable shall support subindex access.	X				X
0075	Button - NoDisplay-Format	7.5.8.5	If in menu subelement ‘Button’ exists for VariableRef or RecordItemRef, the attributes ‘gradient’, ‘offset’, ‘unitCode’ and ‘displayFormat’ shall not be used.	X				
0076	MenuRef - CircularReferences	7.5.8.6	Circular references to menus are not allowed.	X				X
0077	MenuCondition	7.5.8.6	Conditions shall only be of datatype IntegerT, UIntegerT, and BooleanT.	X				X
0078	MenuCondition - ProcessDataCondition	7.5.8.6	If there is more than one ProcessData element selected by conditions, and the variable V_ProcessDataIn or V_ProcessDataOut is referenced in a menu, one of the following shall hold: a) The type of reference (VariableRef/-RecordItemRef) and the gradient, offset, unitCode and displayFormat match each of the ProcessData elements. b) The menu is conditioned in the same way as one of the ProcessData elements, and the type of reference (VariableRef / RecordItemRef) and the gradient, offset, unitCode and displayFormat match this particular ProcessData element (“conditioned in the same way” means that this menu or one of the parent menus has the same condition: same variable, same subindex, same value).	X				X

IODD_TC No	Rule name	Reference	Definition	M	L	W	H	V
0079	MinCycleTime	7.6 [1], B.1.3	The minimum cycle time of the device; specified in units of 1 µs. For example, 2300 represents 2,3 ms.	X				X
0080	PrimaryLanguage	7.7.1	The primary language shall be English (the attribute xml:lang shall have the value “en”).	X				X
0081	TextRedefine	7.7.1	Language dependent text overriding a standard text. Only applicable for texts describing the octets of DirectParameter page 2. It shall be one of STD_TN_DeviceSpecific_1 to STD_TN_DeviceSpecific_16.	X				
0082	NotUsedTexts	7.7.1	There shall be no unreferenced Text elements.	X	X			X
0083	NotTranslatedTexts	7.7.1	Check whether texts are not translated.	X	X	X		X
0084	UniqueDeviceVariant - NameText	7.4.1	Texts referenced by DeviceVariantCollection/DeviceVariant/Name/textId shall be unique within each supported language.	X	X			
0085	UniqueVariable - NameText	7.5.4	Texts referenced by StandardVariables, DirectParameterOverlay or Variables shall be unique within each supported language.	X	X			
0086	OverlappingSingle-Value - ValueRange	7.5.3.1.1	SingleValues and ValueRanges shall not overlap	X				X
0087	Arrays	[1], F.2.3 [1], F.2.4	UIntegerT and IntegerT with a length of ≥ 58 bit and < 64 bit are not permitted.	X				X
0088	Array Alignment	7.5.3.2.2	Array elements with bitLength < 8 bit shall not cross octet boundaries. For array elements with bitLength >= 8 bit use ‘bitLength’ as multiples of 8 bit.	X				
0089	Record - Subindices	[1], F.3.3	The Subindices within the IODD shall be listed in ascending order from 1 to n describing an octet sequence. Gaps within the list of Subindices are allowed.	X				X
0090	RecordItems - Alignment Data-types	[1], F.3.3	The following data types shall always be aligned with octet boundaries: Float32T, StringT, OctetStringT, TimeT, and TimeSpanT.	X				X
0091	RecordItems - Alignment - Integer58	[1], F.3.3	UIntegerT and IntegerT with a length of ≥ 58 bit shall always be aligned with one side of an octet boundary.	X				X
0092	RecordItems – Alignment - Integer10	[1], F.3.3	It is highly recommended for UIntegerT and IntegerT with a length of ≥ 8 bit to align always with one side of an octet boundary.	X		X		X
0093	RecordItems - Alignment - Integer6	[1], F.3.3	It is highly recommended for UIntegerT and IntegerT with a length of < 8 bit not to cross octet boundaries.	X		X		X
0094	RecordItems - Alignment - Integer	7.5.3.2.3	For variables, UIntegerT and IntegerT shall begin on an octet boundary.	X				
0095	RecordItems - Alignment - Boolean	7.5.3.2.3	For variables, it is highly recommended to group BooleanT together into one or few octets.	X		X	X	
0096	RecordItems - AccessRightRestriction	7.5.3.2.3	The ‘accessRightRestriction’ of the RecordItem shall include the accessRights of the variable.	X				X
0097	ProcessData - AccessRightRestriction	7.5.3.2.3	The attribute ‘accessRightRestriction’ is only applicable for ISDU parameter, not for RecordItem for process data.	X				X

IODD_TC No	Rule name	Reference	Definition	M	L	W	H	V
0098	BitOffset	[1], F.3.3	RecordItem shall not overlap	X				X
0099	Connection - MinDef	7.6	Connection: If OtherConnectionT is used, there shall be wires with the function L+, L- and C/Q.	X				
0100	MaxLength – Variables	[1], 4.4	The length of a variable shall not exeed 232 octets.	X				X
0101	MaxLength - ProcessData	[1], 4.4	The length of a ProcessDataIn/Out shall not exeed 32 octets.	X				X
0102	ExcludedFromDataStorage	7.5.4.1	This attribute may only be set to “true” for variables with accessRights = “rw”.	X				
0103	ModifiesOtherVariables	7.5.4.2	This attribute may only be set to “true” for variables with accessRights = “rw” or “wo”.	X				
0104	Dynamic	7.5.4.2	This attribute may only be set to “true” for variables with accessRights = “rw” or “ro”.	X				X
0105	Quantity	B	Quantities shall not be exceeded	X	X			
0106	DeviceAccessLocks - RecordItemRef	7.5.8.4	V_DeviceAccessLocks shall only be referenced via RecordItemRef.	X				
0107	DeviceAccessLocks - Features	7.5.8.5	A RecordItemRef with variableId=”V_DeviceAccessLocks” is only allowed for those subindices which refer to an access lock that is supported, i.e. where the respective attribute in Features/SupportedAccessLocks is set to “true”.	X				
0108	Features – DataStorage - DevSpecISDU	7.5.1	If attribute Features/@dataStorage = “false” and Device specific ISDU variables with accessRights = “rw” are present, issue a warning: “Warning: Support of Data storage is highly recommended” Result: Warning	X		X	X	
0109	Features – DataStorage - StdISDU	7.5.1	If attribute Features/@dataStorage = “false”, all Standard Variables with accessRights = “rw” shall not be implemented. (StdVariableRef to V_ApplicationSpecificTag, V_DeviceAccessLocks, V_OffsetTime) Result: Error	X				
0110	Variable - StdVariableRef	7.5.4.1	For a StdVariableRef to V_DeviceAccessLocks, StdRecordItemRef elements may only be present for supported access locks. Supported means, that the corresponding attribute in Features/SupportedAccessLocks is set to “true”.	X				
0111	Variable - StdVariableRef-ProductID_1	7.5.4.1	If there is only one DeviceVariant present, the value of DeviceVariant/@productId shall be used as default value of V_ProductID.	X				
0112	Variable - StdVariableRef-ProductID_2	7.5.4.1	For references to V_ProductID the attribute ‘defaultValue’ shall not be specified if more than one DeviceVariant is defined.	X				
0113	Menu - VariableRef	7.5.8.4	For VariableRef, the attribute accessRightRestriction shall not be set to “wo”.	X				
0114	Menu - RecordItemRef	7.5.8.5	For RecordItemRef, the attribute accessRightRestriction shall not be set to “wo”.	X				

IODD_TC No	Rule name	Reference	Definition	M	L	W	H	V
0115	Variables - ComplexDatatypes-accessRights	7.5.4.3	For records and arrays the attribute accessRights=”wo” is not permitted.	X				
0116	Variables – Record-RecordItem	7.5.3.2.3	The attribute accessRightRestriction shall not be set to “wo”. An exception to this rule is the V_DirectParameters_1.	X				
0117	PhysicalLayer - mSequenceCapability	[1], A.2.6	Checker shall warn if M-Sequence value leads to Type_0. Checker shall use following text: “Warning – It is highly recommended for Devices not to use TYPE_0 in Preoperate or Operate states”	X		X	X	
0118	VariableRef – Button - ActionStartedMessage	7.5.8.4	ActionStartedMessages shall be omitted for buttons referencing StdSingleValueRefs [@id=”128” or “129” or “130” or “131”].	X	X			
0119	VariableRef – Button - Description	7.5.8.4	For button values described as StdSingleValueRefs [@id=”128” or “129” or “130” or “131”], the Description element is mandatory. The definition in template IODD-SystemCommandDefinitions_V113.xml shall be used. Checker shall check IODD for correct content of button description within each supported language.	X	X			
0120	SupportedAccessLocks - Parameter	7.5.1 [1], B.2.4	It is highly recommended, that Features/SupportedAccessLocks/@parameter is set to “false”.	X		X	X	
0121	SupportedAccessLocks - dataStorage	7.5.1 [1], B.2.4	It is highly recommended, that Features/SupportedAccessLocks/@dataStorage is set to “false”.	X		X	X	
0122	RecordItemRef – DeviceAccessLocks - Parameter	7.5.1	IODD Menu shall not contain reference to V_DeviceAccessLocks subindex=”1”	X				
0123	RecordItemRef – DeviceAccessLocks - DataStorage	7.5.1	IODD Menu shall not contain reference to V_DeviceAccessLocks subindex=”2”	X				

Annex D Profile conformity and testing

D.1 General business rule extensions for the IODD Checker

To achieve consistency and conformity of the profiled Devices to the claimed Profiles, the business rules of the IODD checker are extended covering the Profile requirements.

The rule extensions are generic to suit the Profile requirements and based on IODD snippets which are provided together with the corresponding Profile specifications.

Each Profile provides XML based files containing IODD related snippets, which may be copied and adapted to create well-formed Device IODDs. These XML files contain XML elements following the rules of this IODD Specification which are extended by the following elements and attributes. These specific extensions must be removed when copying the parts into a specific Device IODD.

D.2 Rules for IODD snippet files

This clause defines the layout und and content rules which apply to the IODD snippet files which support the design and test of profiled Devices.

The base rules specify the layout and strategy when an IODD snippet file is generated.

The extensions by attributes are specified in the later clauses, together with the applicable checking rules.

D.2.1 Base rules

The following rules apply for a Profile describing IODD snippet file:

- The XML-file shall be compliant to the schema file defining the snippet files
- The XML-file shall be formatted with "Pretty-Print" to provide a common layout
- The attribute "excludedFromDataStorage" shall be predefined whenever applicable
- ′ids′ shall be unique in the scope of the specific Profile predefinition and use unique prefixes throughout a Profile specification
- Predefined elements are associated to specific ProfileIdentifiers via ProfileConstraints, this allows the reuse of identical or selection of similar items.
- Any elements like datatypes or texts which are referenced by another element do not provide ProfileConstraints, the reference is used to derive the data from snippet and IODD, which has to match according the check rules
- White spaces may improve the readability, but will be ignored by the parser

D.2.2 Menu appearance

The appearance of the profiled Devices is defined in the according Profile specification and enforced via the IODD snippets. The rules defined in "IO-Device-Desc_Guideline" must be considered.

The following rules on menu collections and role menu sets shall apply.

D.2.2.1 Menu collections

Menus or parts of menus may be predefined by a Profile, in this case the parts shall be defined as a menu collection and the allowed deviations shall be marked.

Each Profile parameter shall be defined in the corresponding section. In case of predefined single values, these shall also be defined. If parameters or single values allow multiple different representations, these shall be listed and identified by a common prefix.

Profiles are not obliged to predefine the complete menu structure, but may define sub parts to be incorporated into vendor specific structures. Each menu collection shall be assigned to one of the predefined top level menus.

Top level menu collections may be provided, but shall allow vendor specific extensions in this case.

D.2.2.2 Role menu sets

Role menu sets shall reference at least the top level menus with predefined menu collections assigned by ProfileContext entries. Extensions to the Profile defined entries shall be possible on this level.

D.3 Snippet specific elements

The top-level-element <IODDProfileDefinitions> provides necessary common attributes accompanied by the single <DocumentInfo> element with the release information.

D.3.1 Supported Profiles

The single element “SupportedProfiles” contains general information about the Profiles by this snippet file. This information can be used by the IODD checker or any IODD generating tool chain.

D.3.2 Attributes of SupportedProfiles

The attributes defined in Table 8 provide general information like ProfileIDs, names, and reserved ranges for parameter, events, or commands.

Table 8 – Attributes of SupportedProfiles

Attibute name	Content type	Content definition
profileCharacteristic	Enumeration of integer	List of ProfileIDs covered by this snippet file
profileClassName	Text	Name of the related Profile
profilePrefixes	Text	Prefix associated to the snippet artefacts
reservedIndexRange	List of integer, ranges possible	List of indices covered by this Profile
reservedSystemCommands	List of integer, ranges possible	List of SystemCommands covered by this Profile
reservedEvents	List of integer, ranges possible	List of Events covered by this Profile
requiredProfile	List of integer, ranges possible	List of mandatory additional ProfileIDs

D.3.3 Elements of SupportedProfiles

The elements defined in Table 9 and Table 10 provide general information for each variant of Profile or FunctionClass by their attributes.

Table 9 – ProfileVariant

Attibute name	Content type	Content definition
id	Text	Name of the Profile variant, used as reference in profileConstraints, prefixed by PR_
profileId	Integer	ProfileID of this Profile variant
name	Text reference	Profile variant name to be used by user interfaces
profileOption	List of integers, XOR indicates the exclusiveness of ProfileIDs	Enumeration of allowed additional Profile or FunctionClass extensions defined in one Profile specification
info	Text reference	Text to be shown as IODD Checker output

Table 10 – FunctionClass

Attibute name	Content type	Content definition
id	Text	Name of the FunctionClass, used as reference in profileConstraints, prefixed by FC_
profileId	Integer	ProfileID of this FunctionClass
name	Text reference	Profile variant name to be used by user interfaces
profileContext	List of integer	Enumeration of associated Profile variants

D.4 IODD extensions by snippet files

Some Device IODD artefacts can be predefined by a Profile in its availability, content, or omission. The following clauses define the extension and their interpretation by the IODD checker.

D.4.1 Attribute profileConstraints

The attribute profileConstraints specifies in which Profile context the Device IODD element is applicable and controls the scope of check.

The following syntax and behavioral rules apply:

- Syntax: profileContraints="<id1>"
- An omitted attribute profileConstraints defines a positive matching, means this element is enforced by this snippet file without any exclusion
- The attribute may contain several ids defined for ProfileVariants or FunctionClasses
- Dependencies between two ProfileIDs or FunctionClasses are indicated by the logical expression 'AND'
- The list of ids or expressions are separated by comma
- The attribute profileConstraints applies to the entire element including sub-elements
- The attribute profileConstraints can be used in a sub-element in order to filter the sub-elements before checking. However, only a subset of the ids of the next higher hierarchical element is allowed

D.4.2 Attribute checkAttributes

The attribute checkAttributes is an optional attribute which defines the checking rules for the Device IODD attributes of this element, the possible values are specified in Table 11.

Table 11 – Rules of checkAttributes

Value name	Rule description
exact	All predefined attributes shall exist as predefined. Additional attributes are prohibited
atLeast	All predefined attributes shall exist as predefined. Additional attributes are allowed
option <attribute>	The listed attributes with according values may be referenced, but are not mandatory. Several attributes may be referenced by multiple instantiations in the form "option <attribute1>, option <attribute2>, …"
startsWith	This rule enforces a predefined beginning of an attribute’s value.
contains	This rule enforces the coverage of predefined values within an attribute’s value.
notEmpty	This rule enforces any string content, optionally a minimum length can be requested. The provided value content is a proposal.

The following additional syntax and behavioral rules apply:

- Syntax: checkAttributes="<rule1>, <rule2>, …"
- The list of checking rules are separated by comma
- The rule "exact" is predefined when the attribute is omitted
- The rules "exact", "atLeast", and "option" shall not be combined
- The check does not only cover the attribute’s presence, but also performs a check of the content.
    - A predefined value of "#tbd#" indicates a wildcard for any allowed content
    - An allowed range can be indicated by "#tbd n..m#"

Hint for Device designer regarding extended reference names by startsWith:

- It is mostly allowed to extend the reference itself
- In this case the reference has to be extended in the same way
- In this case any tooling (e.g. JSON or OPC UA) which uses the id as internal reference will experience this difference

D.4.3 Attribute checkElement

The attribute checkElement is an optional attribute in each element to check the order or presence of subelements, the possible values are specified in Table 12.

Table 12 – Rules of checkElement

Value name	Rule description
exact	All predefined sub elements shall exist as predefined. Additional sub elements are prohibited
atLeast	All predefined sub elements shall exist as predefined. Additional sub elements are allowed
atLeastSequence	The predefined sub elements shall exist as predefined and within the defined order without gaps. This rule is only applicable for the elements ′ProcessDataRef′ and ′Menu′ within the section ′UserInterface′
maxOccurs <n>	This type of element is allowed with a maximum number of instances of n
minOccurs <n>	This type of element is required with a minimum number of instances of n

The following additional syntax and behavioral rules apply:

- Syntax: checkElement="<rule1>, <rule2>, …"
- The list of checking rules are separated by comma
- The rule "exact" is predefined when the attribute is omitted
- The rules "exact", "atLeast", and "atLeastSequence" shall not be combined

D.4.4 Attribute contextConstraints

The attribute contextConstraints is an optional attribute used within the section ′UserInterface′ only. It enforces a reference to an element within the indicated menu group. The attribute has no default value when omitted.

Syntax: contextConstraints="<menugroup>"

Permissible values for menugroup are:

- IdentificationMenu
- ParameterMenu
- ObservationMenu
- DiagnosisMenu

D.5 Test requirements

D.5.1 Test sequence

The following test sequence is performed by the IODD checker as an extended business rule whenever the profileCharacteristic are not empty.

The sequence of test steps as extension of the IODD checker is specified in Table 13.

Table 13 – Test sequence

Test step	Description
1	Read profileCharacteristic from Device-IODD and decompose ProfileIDs
2	Iterate over all available Profile snippets files and perform the following tests
2.1	Read Profile snippets file
2.2	Perform the following tests on each matching element including the references in /IODevice/ProfileBody/DeviceFunction/Features, /IODevice/ProfileBody/DeviceFunction/VariableCollection, /IODevice/ProfileBody/DeviceFunction/ProcessDataCollection, /IODevice/ProfileBody/DeviceFunction/UserInterface/ProcessDataRefCollection, /IODevice/ProfileBody/DeviceFunction/UserInterface/EventCollection, and /IODevice/ProfileBody/DeviceFunction/UserInterface/MenuCollection Check the presence of the corresponding element in the Device-IODD. In case of failure, create an error log entry and skip tests 2.3 to 2.6 on this element
2.3	Read checkAttribute Perform test on specific element regarding attribute rules. In case of failures, create an error log entry
2.4	Read checkElement Perform test on "minOccurs" and "maxOccurs" of this type of element against elements on the same level. In case of failures, create an error log entry Perform test on structure rules "exact", "atLeast", and "atLeastSequence" against sub-elements (on each level). In case of failures, create an error log entry
2.5	Read contextConstraints Check presence for menu, variable or recorditem reference within the indicated menu group. In case of failures, create an error log entry
2.6	For each reference as "Name textId", "Description textId", "DatatypeRef datatypeId", and "ProcessDataRef processDataId" Get matching elements from snippet and Test-IODD and perform check based on the rules for checkAttributes and checkElement (see 2.3 and 2.4). In case of failure, create an error log entry
3	Perform action for each snippet file from 2.1 on

The following rules specify the check of the attribute’s value constraints:

- any predefined value shall be provided as predefined
- exception: an entry of “#tbd#” is used as a wildcard for any allowed content, it’s on behalf of the standard IODD Checker rules to perform any further contextual check
- exception: an entry of #tbd n..m# is used to restrict the allowed value to a range of numbers from n to m
- when checking the attribute excludedFromDataStorage, the presetting of "false" is assumed when the checked IODD does not contain the attribute

D.5.2 Test on exclusive use of Profile parameters

The IODD checker shall check that no Profile parameter, SystemCommand, or Event is used without reference to an appropriate ProfileID. The necessary information shall be provided by the snippet files.

Bibliography

The following referenced documents are indispensable for the application 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.

Related documents

[1] IO-Link Community, *IO-Link Interface and System* Specification Version 1.1.4, June 2024, Order No. 10.002

[2] IO-Link Community, IO-Link Communication Specification Version 1.0, January 2009, Order No. 10.002

[3] IO-Link Community, IO-Link Device Description Specification Version 1.0.1, March 2010, Order No. 10.012

[4] IO-Link Community, IO-Link Test Specification Version 1.1.4, June 2024 Order No: 10.032

[5] ANSI/IEEE Std 754, *IEEE Standard for Binary Floating-Point Arithmetic*

[6] IETF RFC 2083, *PNG (Portable Network Graphics) Specification Version 1.0, available at http://tools.ietf.org/html/rfc2083*

[7] ISO 639:2023, *Codes for the representation of names of languages: Alpha-2 code*

[8] ISO/IEC 646:1991, Information technology – ISO 7-bit coded character set for information interchange

[9] ISO 15745-1:2003, Industrial automation systems and integration – Open systems application integration framework – Part 1: Generic reference description

[10] ISO 15745-1 Amd 1:2007, *Industrial automation systems and integration – Open systems application integration framework – Part 1: Generic reference description, Amendment 1*

[11] IEC 60757, *Code for designation of*

[12] The Unicode Standard, *available at http://www.unicode.org/*

[13] ITU-T recommendation V.42 (03/2002), *Error-correcting procedures for DCEs using asynchronous-to-synchronous conversion, available at http://www.itu.int/rec/T-REC-V.42-200203-I/en*

[14] REC-xml-20081126, *Extensible Markup Language (XML) 1.0 (Fifth Edition) – W3C Recommendation 26 November 2008, available at http://www.w3.org/TR/xml/*

[15] REC-xmlschema-1-20041028, *XML Schema Part 1: Structures Second Edition – W3C Recommendation 28 October 2004, available at http://www.w3.org/TR/xmlschema-1/*

[16] REC-xmlschema-2-20041028, *XML Schema Part 2: Datatypes Second Edition – W3C Recommendation 28 October 2004, available at http://www.w3.org/TR/xmlschema-2/*

[17] ANSI INCITS 4-1986 (R2007), *Information Systems – Coded Character Sets – 7-Bit American National Standard Code for Information Interchange (7-Bit ASCII) (predecessor of ISO/IEC 646)*

[18] IEC 60559, *Information technology – Microprocessor Systems – Floating-Point arithmetic*

[19] IETF RFC 3629, *UTF-8, a transformation format of ISO 10646*, available at http://tools.ietf.org/html/rfc3629

[20] IETF RFC 5905, *Network Time Protocol Version 4: Protocol and Algorithms Specification*, available at http://tools.ietf.org/html/rfc5905

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

[22] ISO 8601-1:2019, *Data elements and interchange formats – Information interchange – Representation of dates and times*

[23] ISO/IEC 10646: , *Information technology – Universal Multiple-Octet Coded Character Set (UCS)*

[24] ISO/IEC 15948:2004, *Information technology – Computer graphics and image processing – Portable Network Graphics (PNG): Functional specification*

[25] REC-xslt-19991116, *XSL Transformations (XSLT), Version 1.0, W3C Recommendation 16 November 1999*, available at http://www.w3.org/TR/xslt

IO-Link Community
c/o PROFIBUS Nutzerorganisation e.V.
Ohiostrasse 8
76149 Karlsruhe
Germany
Phone: +49 (0) 721 / 98 61 97 0
Fax: +49 (0) 721 / 98 61 97 11
e-mail: info@io-link.com
http://www.io-link.com/