

IO-Link Common Profile

Specification

**Version 1.2
January 2024**

Order No: 10.072

File name: IO-Link_CommonProfile_V1.2_Jan2024.pdf

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: *IOL-SM-Profile*

Password: *Report*

Important notes:

NOTE 1 The IO-Link Community Rules shall be observed 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 with this specification, its related IODD, and test documents, 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 can 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

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

© No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the publisher.

CONTENTS

- 1 Scope7
- 2 Normative references7
- 3 Terms, definitions, symbols, abbreviated terms and conventions7
 - 3.1 CommonProfile: Additional terms and definitions7
 - 3.2 Symbols and abbreviated terms8
 - 3.3 Conventions.....8
 - 3.3.1 Behavioral descriptions.....8
- 4 Objectives for Device profiles9
 - 4.1 Purpose of Device profiles9
 - 4.2 Interoperability9
 - 4.3 Common Profile Specification structure..... 11
- 5 Device profiles related to IEC 61131-9 11
 - 5.1 SDCI technology specified in IEC 61131-9..... 11
 - 5.2 Profile classification 11
 - 5.3 Concept of profiles..... 12
 - 5.3.1 Basic requirements for profile Devices 12
 - 5.3.2 Distinct profiles..... 13
 - 5.4 Profile characteristics 13
 - 5.5 Concept of FunctionClasses 14
 - 5.6 User benefits 15
- 6 Rules and constraints for developing IO-Link profile Devices 15
 - 6.1 Constraints for developing IO-Link Devices..... 15
 - 6.2 How to select the appropriate Profile functions 16
 - 6.3 Identification of supported Profiles 16
- 7 Identification and Diagnosis (I&D)..... 16
 - 7.1 Overview 16
 - 7.2 Identification and Diagnosis Profile (I&D) 16
 - 7.3 Extension of Identification and Diagnosis 17
 - 7.4 Proxy Function Block for Identification and Diagnosis 17
- Annex A (normative) FunctionClasses 18
 - A.1 Overview 18
 - A.2 Device identification objects [0x8000] 18
 - A.3 Process Data mapping (PDV) [0x8002] 18
 - A.3.1 Overview 18
 - A.3.2 Process data description 18
 - A.4 Diagnosis [0x8003] 19
 - A.5 Extended Identification [0x8100] 19
 - A.6 Locator [0x8101]..... 19
 - A.7 ProductURI [0x8102]..... 21
 - A.8 TeachRecommended [0x8103]..... 21
- Annex B (normative) Profile relevant Device parameters..... 23
 - B.1 Overview 23
 - B.2 System Commands 24

B.3	Identification parameters.....	24
B.4	ProfileCharacteristics parameter	24
B.5	Process data structure descriptors	25
B.5.1	Coding of PVinD and PVoutD	25
B.5.2	PDInputDescriptor	26
B.5.3	PDDescriptionDescriptor.....	26
B.6	Extended Identification parameters	26
B.7	Diagnosis parameters	27
B.8	ProductURI parameter	27
Annex C	(normative) Function block definitions	28
C.1	Overview	28
C.2	Proxy function block (FB) for identification and diagnosis.....	28
Annex D	(normative) IODD definition and rules	31
D.1	Overview	31
D.2	Name definitions	31
D.2.1	Profile type characteristic names	31
D.3	IODD Menu definitions	31
D.3.1	Overview	31
D.3.2	Explanation of used object layout	31
D.3.3	Menu structure of the Device Diagnosis parameter	31
D.3.4	Menu structure of the Device Identification parameters.....	32
D.3.5	Menu structure of the Locator functionality	32
Annex E	(normative) Profile testing and conformity	34
E.1	General.....	34
E.1.1	Overview	34
E.1.2	Test extension for profile Devices	34
E.1.3	Business rule extensions for the IODD Checker.....	34
Annex F	(normative) Testing Identification and Diagnosis	35
F.1	Test case extension for static parameter design	35
F.1.1	Ordering of Profile characteristics	35
F.1.2	Hiding FunctionClasses by ProfileIDs	36
F.1.3	Minimum required profile support.....	37
F.1.4	Extensions of profiles	38
F.1.5	PDInput-, PDDescriptionDescriptor parameter	39
F.2	Test case extension for dynamical behavior	40
F.2.1	Device localization commands	40
Bibliography	41
Figure 1	– Compatibility levels based on IEC 62390	9
Figure 2	– Domain of the SDCI technology within the automation hierarchy	11
Figure 3	– Overview of SDCI technologies and profiles	12
Figure 4	– Overview of typical FunctionClasses	14
Figure A.1	– State machine for optical indication	20
Figure A.2	– Device optical indicator timing for localization	21
Figure B.1	– Indication rules for ProfileIdentifiers	25
Figure C.1	– Proxy FB for Device Identification and Diagnosis	28
Figure D.1	– IODD object layout description	31

Figure D.2 – Menu Profile Diagnosis 32

Figure D.3 – Menu Profile Identification 32

Figure D.4 – Menu Profile Locator 33

Table 1 – Explanation of compatibility levels 10

Table 2 – Explanation of Device features 10

Table 3 – Example of the profile identification of a distinct switching sensor 13

Table 4 – Example of the profile identification of an extended Profile 13

Table 5 – Tag notation for BDC and PDV access of a PLC client 15

Table 6 – Prefixes for IODD ID elements 16

Table 7 – Identification and Diagnosis Device profile 16

Table 8 – Associated SDCI artefacts for Identification and Diagnosis 17

Table 9 – Extension for I&D 17

Table A.1 – Overview of FunctionClasses 18

Table A.2 – DeviceStatus priority 19

Table A.3 – State transition tables for optical indication 20

Table A.4 – Timing for the optical indication 21

Table B.1 – General profile relevant Device parameters 23

Table B.2 – Conditional "SystemCommand" 24

Table B.3 – Definitions for identification data objects 24

Table B.4 – Parameter "ProfileCharacteristic" 25

Table B.5 – Coding of PVinD or PVoutD 25

Table B.6 – Structure of "PDInputDescriptor" 26

Table B.7 – Structure of "PDOOutputDescriptor" 26

Table B.8 – Parameter Extended Identification 27

Table B.9 – Structure of "DetailedDeviceStatus" 27

Table B.10 – Definitions for ProductURI parameter 27

Table C.1 – Variables of "IOL_IdentificationAndDiagnosis" FB 29

Table C.2 – Elements of the IdentificationObjects 30

Table F.1 – Ordering of Profile characteristics 35

Table F.2 – Hiding FunctionClasses of I&D 36

Table F.3 – Minimum required profile support 37

Table F.4 – Extension of Identification and Diagnosis 38

Table F.5 – PDInput-, PDOOutputDescriptor parameter 39

Table F.6 – Device localization commands 40

1 **0 Introduction**

2 **0.1 General**

3 The Single-drop Digital Communication Interface (SDCI) and system technology (IO-Link™¹)
4 for sensors and actuators is standardized within IO-Link Interface and System Specification [1].
5 The technology is an answer to the need of these digital/analog sensors and actuators to ex-
6 change process data, diagnosis information and parameters with a controller (PC or PLC) using
7 a digital communication technology while maintaining backward compatibility with the current
8 DI/DO signals as defined in IEC 61131-2.

9 Tools allow the association of Devices with their corresponding electronic I/O device descrip-
10 tions (IODD) and their subsequent configuration to match the application requirements [2].

11 This document describes the common parts of sensor and actuator models to be used in all
12 Device profiles as well as the base profile "Identification & Diagnosis" which is mandatory for
13 all profiled Devices.

14 This document follows the IEC 62390 [3] to a certain extent.

15 Terms of general use are defined in IEC 61131-1 or in [4]. Specific SDCI terms are defined in
16 this part.

17 **0.2 Patent declaration**

18 There are no known patents related to the content of this document.

19 Attention is drawn to the possibility that some of the elements of this document may be the
20 subject of patent rights. The IO-Link Community shall not be held responsible for identifying
21 any or all such patent rights.

22

¹ IO-Link™ is a trade name of the "IO-Link Community". This information is given for the convenience of users of this specification. Compliance to this specification does not require use of the registered logos for IO-Link™. Use of the registered logos for IO-Link™ requires permission of the "IO-Link Community".

Common Profile — Related to IO-Link Interface and System

23
24
25

26 **Scope**

27 The single-drop digital communication interface (SDCI) technology described in part 9 of the
28 IEC 61131 series focuses on simple sensors and actuators in factory automation, which are
29 nowadays using small and cost-effective microcontrollers. With the help of the SDCI technology,
30 the existing limitations of traditional signal connection technologies such as switching 0/24 V,
31 analog 0 to 10 V, etc. can be turned into a smooth migration. Classic sensors and actuators are
32 usually connected to a fieldbus system via input/output modules in so-called remote I/O periph-
33 erals. The (SDCI) Master function enables these peripherals to map SDCI Devices onto a
34 fieldbus system or build up direct gateways. Thus, parameter data can be transferred from the
35 PLC level down to the sensor/actuator level and diagnosis data transferred back in turn by
36 means of the SDCI communication. This is a contribution to consistent parameter storage and
37 maintenance support within a distributed automation system. SDCI is compatible to classic sig-
38 nal switching technology according to part 2 of the IEC 61131 series.

39 This specification contains general explanations on how Profiles are defined in the context of
40 SDCI. It also defines the general Identification and Diagnosis profile, which is the base for all
41 SDCI based profile Devices.

42 **Normative references**

43 The following referenced documents are indispensable for the application of this document. For
44 dated references, only the edition cited applies. For undated references, the latest edition of
45 the referenced document (including any amendments) applies.

46 IEC 61131-3, *Programmable controllers – Part 3: Programming languages*

47 IEC 61131-9, *Programmable controllers – Part 9: Single-drop digital communication interface
48 for small sensors and actuators (SDCI)*

49 **Terms, definitions, symbols, abbreviated terms and conventions**

50 For the purposes of this document, the following terms and definitions in addition to those given
51 in IEC 61131-1, IEC 61131-2, and IEC 61131-9 apply.

52 **3.1 CommonProfile: Additional terms and definitions**

53 **3.1.1**

54 **BinaryDataChannel**

55 BDC

56 binary information as switching or control signal

57 **3.1.2**

58 **Function block**

59 software functional element comprising an individual, named copy of a data structure and as-
60 sociated operations specified by a corresponding function block type

61 [SOURCE: IEC 62390, 3.1.13]

62 **3.1.3**

63 **FunctionClass**

64 particular function within a Device profile

65 Note 1 to entry: A profile Device can use one or several FunctionClasses once or several times.

66 **3.1.4**

67 **not applicable**

68 n/a

69 this entry cannot be applied within this context

70 **3.1.5**
71 **ProfileIdentifier**
72 unique identifier for Device Profile, CommonApplicationProfile, or FunctionClass

73 **3.2 Symbols and abbreviated terms**

PD	Process Data
PLC	Programmable logic controller
SDCI	Single-drop digital communication interface
CP	CommonProfile

74

75 **3.3 Conventions**

76 **3.3.1 Behavioral descriptions**

77 For the behavioral descriptions, the notations of UML2 [7] are used, mainly state diagrams. The
78 layout of the associated state-transition tables is following IEC 62390 [3].

79 The state diagrams shown in this document are entirely abstract descriptions. They do not
80 represent a complete specification for implementation.

81 **Objectives for Device profiles**

82 **4.1 Purpose of Device profiles**

83 In factory automation, sensors nowadays are using a broad spectrum of transducers based on
 84 many different physical or chemical effects. They are converting one or more physical or chem-
 85 ical quantities (for example position, pressure, temperature, substance, etc.) and propagate
 86 them in an appropriate form to data processing units such as for example PLCs.

87 Also actuators like lamps, locks, valves, motors, and so on are not only actuators. The internal
 88 states are going to be important for the customers. Even the acknowledged control and config-
 89 uration is of increasing importance and not covered by simple digital output signals.

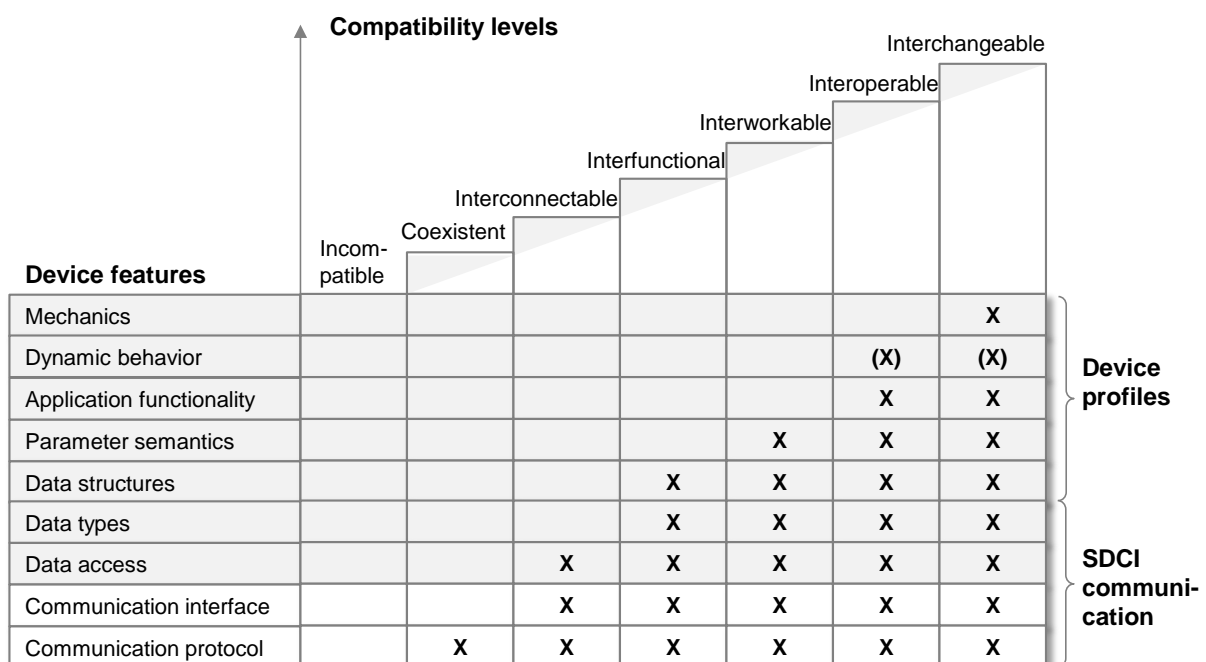
90 It is the purpose of SDCI to overcome the limitations of the classic Device interfaces DI, DO,
 91 AI, and AO via a point-to-point digital communication that allows transmitting digitally not only
 92 binary and/or analog information but additional information also. Very often, the changes to the
 93 core sensor or actuator application ("sensor/actuator technology") are very little during the mi-
 94 gration to SDCI. However, the user realizes a dramatic increase in comfort and flexibility
 95 through the identification, parameterization, and diagnosis features.

96 As a consequence of the digitization, the Devices can also provide many more technology fea-
 97 tures and data structures to the user for processing within for example a PLC user program
 98 than before with the classic interfaces.

99 Device profiles define terminologies, features, behaviours, commands, responses, correspond-
 100 ing data structures, and other things common to particular Device families and thus prevent the
 101 user from a confusing variety.

102 **4.2 Interoperability**

103 The major parts of the Device profiles deal with process data structures and behavior as well
 104 as parameter data structures and dynamic parameterization at runtime. These features stream-
 105 line the functions of comparable Devices though requiring more sophisticated and powerful PLC
 106 user programs. Thus, interoperability between existing user programs and Devices of a corre-
 107 sponding family is the goal of profile Device design and testing (see [3]). Figure 1 shows com-
 108 patibility levels based on IEC 62390.



109
 110 **Figure 1 – Compatibility levels based on IEC 62390**

111 The different compatibility levels are described in Table 1 and the different Device features are
 112 described in Table 2, based on [3].

113

Table 1 – Explanation of compatibility levels

Compatibility level	Definition
Incompatible	Two or more devices are incompatible if they cannot exist together in the same distributed system
Coexistent	Two or more devices coexist on the same communications network if they can operate independently of one another in a physical communication network or can operate together using some or all of the same communication protocols, without interfering with the application of other devices on the network
Interconnectable	Two or more devices are interconnectable if they are using the same communication protocols, communication interface and data access
Interfunctional	Two or more devices are interfunctional if they can exchange data for specific purposes without manual configuration, the parameter semantics are defined and the devices provide the necessary identifier
Interworkable	Two or more devices are interworkable if they can transfer parameters between them, i.e. in addition to the communication protocol, communication interface, and data access, the parameter data types are the same
Interoperable	Two or more devices are interoperable if they can work together to perform a specific role in one or more distributed applications. The parameters and their application related functionality fit together both syntactically and semantically. Interoperability is achieved when the devices support complementary sets of parameters and functions belonging to the same profile
Interchangeable	Unlike the other compatibility levels (which refer to two or more devices working in the same system) interchangeability refers to the replacement of one device with another. Devices are interchangeable for a given role in a distributed application if the new device has the functionality to meet the application requirements

114

115

Table 2 – Explanation of Device features

Device features	Definition
Mechanics	This feature is defined by the mechanical outline, process connector, and/or electrical connection
Dynamic behavior	This feature is defined by time constraints that influence the parameter update or the general device behavior. For example, the update rate of a process value can influence block algorithms.
Application functionality	This feature is defined by specifying the dependencies and consistency rules within the functional element. This is done in the parameter description characteristics or in the device behavior section
Parameter semantics	This feature is defined by the parameter characteristics: parameter name, parameter description, parameter range, substitute value of the parameter, default value, persistence of the parameter after power loss and deployment
Data structures	This feature is defined by the combination of data types, such as records of simple data types
Data types	This feature is defined by characteristics such as "data type", see Note
Data access	This feature is defined by characteristics such "parameter timing" and "access direction", see Note
Communication interface	This feature is defined by the protocols of layer 5 to 7 of the OSI reference model including the services and the service parameters. Additional mapping mechanisms can be necessary. The dynamic performance of the communication system is part of this feature
Communication protocol	This feature is defined by all protocols of layer 1 to 4 of the OSI reference model, i.e. from the physical medium access to the transport layer protocol

Device features	Definition
<p>Note: "parameter timing": in the context of SDCI this refers to the used data channels like acyclic or process data "access direction": specification whether the parameter can be read and/or written "data type": IEC 61131-3 data types are preferred</p>	

116

117 **4.3 Common Profile Specification structure**

118 This specification covers the base understanding of profiles for any Device designer in clause 0.
 119 Clause 0 describes the SDCI related view on profiles with some examples for better understanding.
 120

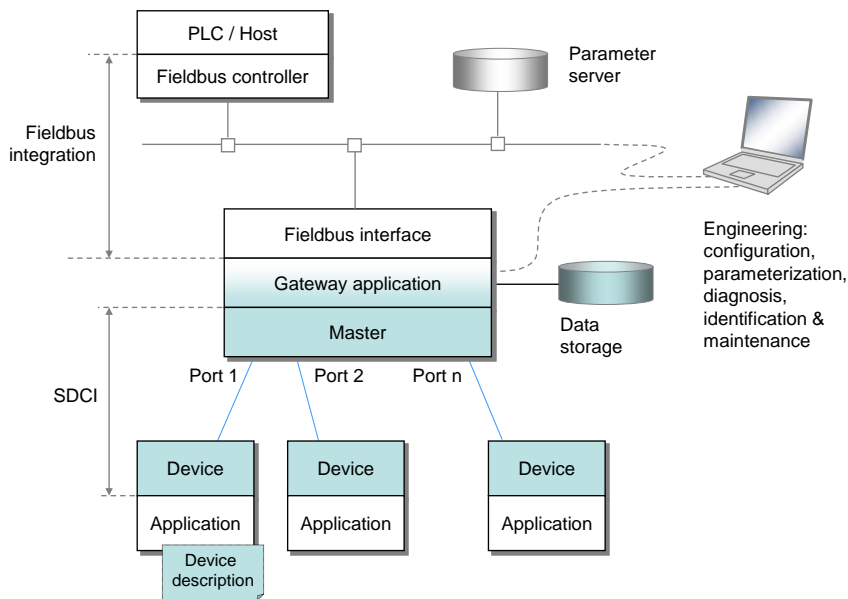
121 Clause 0 contains general rules and constraints for the device designer when profiles are integrated into Devices.
 122

123 Clause 0 specifies the Identification and Diagnosis profile. The Annex A defines the Function-Classes, Annex B specifies the profile parameter, Annex C specifies the associated PLC function block. Annex D specifies the IODD related section, Annex E contains the test case description as extension to the protocol test system.
 124
 125
 126

127 **Device profiles related to IEC 61131-9**

128 **5.1 SDCI technology specified in IEC 61131-9**

129 Figure 2 shows the domain of the SDCI technology within the automation hierarchy.



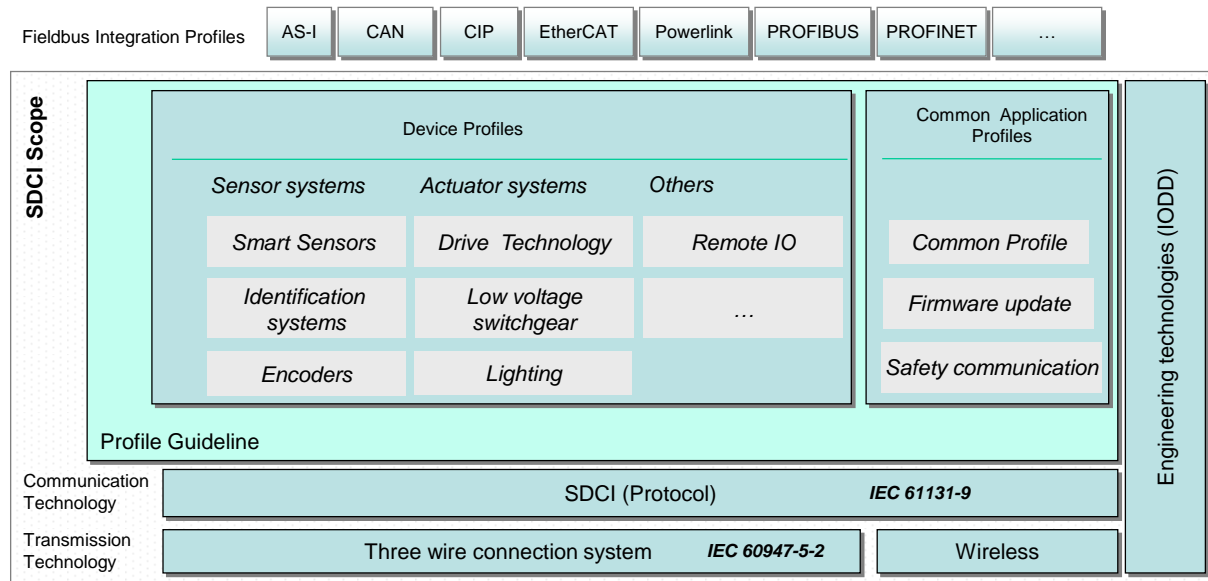
130

131 **Figure 2 – Domain of the SDCI technology within the automation hierarchy**

132 The SDCI technology defines a point-to-point digital communication interface for connecting
 133 "digital" or "analog" type sensors and actuators to a Master unit, which can be combined with
 134 gateway capabilities to become a fieldbus remote I/O node. The technology is specified in [1]
 135 and [2].

136 **5.2 Profile classification**

137 Figure 3 shows an overview of the SDCI technologies and profiles.



138

139

Figure 3 – Overview of SDCI technologies and profiles

140 The "Device Profiles" represent specifications of common functionality of particular Device type
141 families/classes such as

- 142 • smart sensors,
143 • smart actuators,
144 • lighting
145 • etc.

146 These profiles primarily focus on the structure and behavior of the Device technology and sec-
147 ondarily on the data structure mapping on SDCI. Thus, the user experiences a well-known De-
148 vice behavior to a certain extent even when he uses different Devices or switches from one
149 brand to another.

150 The "Common Application Profiles" represent specifications that may be used by several Device
151 type families/ classes such as

- 152 • Common Profile
153 • Firmware update
154 • Safety communication
155 • etc.

156 The "Fieldbus Integration Profiles" specify the adaptation of the SDCI technology to particular
157 fieldbuses. These specifications are outside the responsibility of the organization listed on the
158 last page of this document. However, this organization is interested in harmonizing the "views"
159 of SDCI users through the different fieldbuses.

160 5.3 Concept of profiles

161 The approach for profiling SDCI Devices follow the guideline to enforce as many equal func-
162 tionality or behavior in all SDCI Devices. It is a stacked approach by defining a common subset
163 which are highly recommended for all Devices like defined in the basic requirements. Further-
164 more the application specific profiles such as smart sensors or specific actuators define appli-
165 cation specific requirements and solutions.

166 5.3.1 Basic requirements for profile Devices

167 As [1] defines only a few parameter as mandatory, the profile Devices shall support more pa-
168 rameters to allow standardized handling of these devices.

169 The following base features shall be supported by all profile devices :

- 170 • IO-Link Version 1.1
- 171 • ISDU support
- 172 • DataStorage for all parameters which a spare part needs for full functionality
- 173 • Support of block parameter handling
- 174 • Profile "Identification and Diagnosis"

175

176 5.3.2 Distinct profiles

177 The distinct profiles consist of a defined combination of one or multiple FunctionClasses identified by ProfileIdentifier. The mandatory FunctionClasses are defined in the related specifications and may contain FunctionClasses from different specifications. The Device functionality can be extended by manufacturer specific parameters or additional FunctionClasses. The user can always and only rely on the functions defined by the ProfileIdentifier of the Device, this provides a higher level of interchangeability even between different manufacturers.

183 To sharpen the distinct profiles and reduce the amount of similar profiles with minor differences, some FunctionClasses may be accompanied to profiles when they are supplements to the profile functionality.

186 5.4 Profile characteristics

187 The profiles are based on the definition of FunctionClasses. These FunctionClasses are combined to ProfileIDs such as

- 189 • DeviceProfileIDs for particular classes of Devices, or
- 190 • CommonApplicationProfileIDs for generic use in all Devices.

191 The supported functionality of a Device shall be listed within an array of ProfileIdentifier. It is also possible for a Device to support several DeviceProfiles, CommonApplicationProfiles as well as FunctionClasses as extensions for specific ProfileIDs (see 5.5).

194 The CommonApplicationProfile Identification and Diagnosis (I&D) is mandatory for Devices which provide at least one other profile and highly recommended for all other Devices.

196 An overview of the defined ProfileIdentifier is available on www.io-link.com.

197 The parameter object "ProfileCharacteristic" provides a list of the supported profiles.

198 Table 3 shows the example content of the "ProfileCharacteristic" of an adjustable switching sensor.

200 **Table 3 – Example of the profile identification of a distinct switching sensor**

Index	ProfileIdentifier type	Referenced Profile ID
0x000D	DeviceProfileID	0x0005: SSP 2.2
		0x4000: I&D

201

202 Table 4 shows the example content of the "ProfileCharacteristic" of a Smart Sensor with an additional FunctionalClass.

204 **Table 4 – Example of the profile identification of an extended Profile**

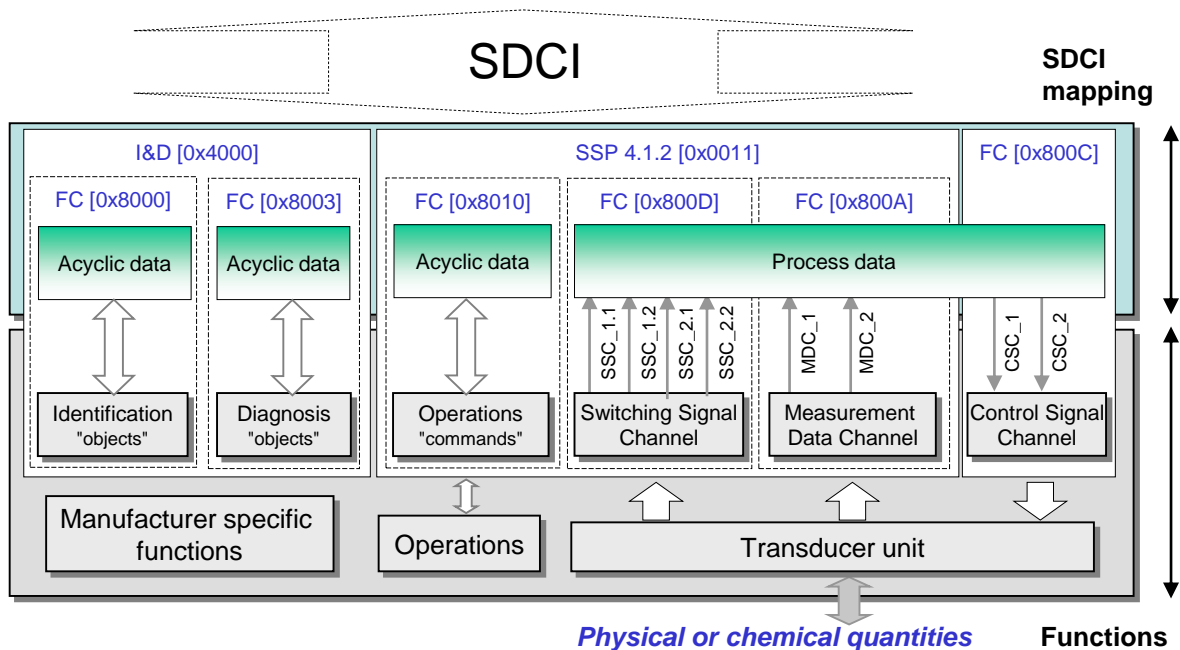
Index	ProfileIdentifier type	Referenced ProfileID
0x000D	DeviceProfileID	0x0005: SSP 2.2
		0x4000: I&D
	FunctionClassID	0x8101: Locator

205 For further details, see B.4.

206 5.5 Concept of FunctionClasses

207 So far only a so-called function-driven Device model instead of for example an architectural
 208 model is defined. That means it only defines independent and consistent functions (Function-
 209 Classes) that are available via the communication channels. This allows the community to cre-
 210 ate a variety of combinations from basic switching sensors/actuators using only the Function-
 211 Class Switching Signal Channel (SSC) up to complex sensors/actuators with several measure-
 212 ment values adding FunctionClasses like the Measurement Data Channel.

213 Figure 4 shows the structure of the function-driven Device model, its combination of DevicePro-
 214 files, CommonApplicationProfiles and FunctionClasses based on the example of a Measuring
 215 and Switching Sensor, 2 channel defined in [6].



216

217

Figure 4 – Overview of typical FunctionClasses

218 Each and every FunctionClass consists of a communication dependent function and an associ-
 219 ated mapping on the SDCI communication. FunctionClasses are represented and referenced
 220 by ProfileIdentifiers, for example FunctionClass [0x8000], as shown in Figure 4.

221 The FunctionClasses DeviceIdentification, DeviceDiagnosis, ProcessDataVariable, and Ex-
 222 tendedIdentification are combined to the DeviceProfile [0x4000] in this document. This Com-
 223 monApplicationProfile is mandatory by definition of this specification.

224 A Switching Signal Channel (e.g. FunctionClass [0x800D]) uses the measurement values out
 225 of the transducer unit and creates switching information (SSC_n), whenever certain thresh-
 226 olds are passed. These thresholds are defined via parameters.

227 In case of a combined sensor/actuator Device, the FunctionClass [0x800C] is used for switching
 228 the transducer ON or OFF.

229 The operation commands like FunctionClasses [0x8010] allow the user for example to remotely
 230 adjust a Device in the automation process via the user program in a controller (PLC).

231 The mapping of SSCs, CSCs and PDVs into SDCI communication messages is specified in the
 232 corresponding FunctionClass definitions. These data structures are designed for simplicity and
 233 highest efficiency.

234 The parameter "ProfileCharacteristic" contains at least one ProfileIdentifier or an array of Pro-
 235 fileIdentifier.

236 The objects SSC, CSC and MDC can be used once or more times depending on the complexity
237 of the sensor/actuator.

238 The parameter set of a FunctionClass are classified into two groups:

- 239 • Operating parameters, which are modified during production, and
- 240 • Configuration parameters (static data), which are only set/modified during commissioning.

241

242 5.6 User benefits

243 As already mentioned in 5.2 the user recognizes from the application point of view a "generic"
244 Device through the communication interface even though he switches from one brand to an-
245 other. The customer experiences the following advantages of profile Devices at different points
246 in time:

- 247 • At commissioning time the process data can be easily configured due to reduced sets of
248 process data structures. In future this could be extended by explicit support of profile De-
249 vices by the system provider.
- 250 • At programming time the process data and common parameters can be used with expected
251 behavior and without checking the IODD of the specific Device, just based on the profile
252 defined behavior. This will be supported by specific Proxy Function Blocks for the defined
253 Profiles.
- 254 • At runtime the Devices represent their process data in an equal manner and can be replaced
255 by Devices with the same ProfileIdentifier and the same physical measurement or actuator
256 behavior. For the replacement only the configured Device Identification has to be updated.

257 However, due to the objectives for the individual Device profiles, the interoperability levels can
258 be different and the compatibility between the profile Devices can be partly limited. For example
259 the measurement range of a sensor or actuator strength can be different and not suitable for
260 the specific application. It is the responsibility of system maintenance to check this prior to a
261 replacement of the Device.

262 A user program ("client") for example in a PLC can access the objects via corresponding func-
263 tions or methods respectively. Table 5 shows an example.

264 **Table 5 – Tag notation for BDC and PDV access of a PLC client**

Read/Write access	Description
Read Sensor1.AppSpecTag	Readout of the parameter "ApplicationSpecificTag" of the Device
Read Sensor1.DeviceStatus	Readout of the parameter "DeviceStatus"
Write Sensor1.switch point1.SetPointValueSP1	Write parameter "SetPointValueSP1"
Write Sensor1.TeachTP1	Start teach procedure for TP1

265 Rules and constraints for developing IO-Link profile Devices

266 Within this clause the general rules and constraints for the design of IO-Link Devices with sup-
267 port of profiles are defined.

268 6.1 Constraints for developing IO-Link Devices

269 When designing a new IO-Link Device the designer should consider the existing Device Profiles
270 available on www.io-link.com. To offer a potential user a maximum of benefits, as defined in
271 5.6, a maximum of CommonApplicationProfiles, DeviceProfiles or at least FunctionClasses
272 should be supported by the new Device.

273 If the Device does not fit exactly into the restricted Profiles, the designer has to consider be-
274 tween an easy usage with DeviceProfiles and on the other hand special capabilities which are
275 not commonly used. In simple words: does the special characteristic or behavior justify a device
276 without standardized or profiled functionalities.

277 6.2 How to select the appropriate Profile functions

278 It is mandatory to support the Identification & Diagnosis profile according 7.2 to harmonize the
279 core functionality of all IO-Link Profile Devices.

280 At first the CommonApplicationProfiles should be considered, they define base functionalities
281 corresponding to the IO-Link system itself.

282 As a next step the specific DeviceProfiles should be considered, they define specific function-
283 alities for the specific types of Device like sensor or actuator. DeviceProfiles or CommonAppli-
284 cationProfiles may define specific FunctionClasses as possible extensions.

285 It is not allowed to use FunctionClasses without the related DeviceProfiles or CommonAppli-
286 cationProfiles for which the FunctionClasses are defined as extensions.

287 It is not allowed to support parameters, commands, or events of any FunctionClass without
288 claiming the ProfileID for which the functionality is defined.

289 6.3 Identification of supported Profiles

290 It is highly recommended to provide the supported Device Profiles by mentioning them with their
291 associated Profile Characteristic Name or FunctionClass Name in the technical documentation
292 and in marketing brochures. This enables the customer to identify these profiled Devices
293 amongst others and allows to identify the standardized features of a particular Device.

294 Identification and Diagnosis (I&D)

295 7.1 Overview

296 It is very important to provide all necessary identification and diagnosis information in a unified
297 manner and with the same contents to interpret.

298 As [1] specifies the required objects as optional, this CommonApplicationProfile specifies these
299 parameters as mandatory for the profile Devices

300 The profile specific abbreviation for all artefacts associated with the CommonProfile is defined
301 in Table 6.

302 **Table 6 – Prefixes for IODD ID elements**

Profile name	Context identifier
CommonProfile	CP

303

304 7.2 Identification and Diagnosis Profile (I&D)

305 Table 7 provides an overview of the FunctionClasses for Identification and Diagnosis.

306 **Table 7 – Identification and Diagnosis Device profile**

ProfileID	Profile characteristic name	Function Classes		
0x4000	Identification and Diagnosis	0x8000	Device Identification	See A.2
		0x8003	Device Diagnosis	See A.4
		0x8002	Process Data Mapping	See A.3
		0x8100	Extended Identification	See A.5

307

308 The associated parameters of the Identification and Diagnosis profile are listed in Table 8.
309 These are already defined in [1], the profile states them as mandatory.

310

Table 8 – Associated SDCI artefacts for Identification and Diagnosis

Profil type	Associated parameter	Functional description
I&D	ProfileCharacteristic	See B.4 and clause B.2.5 in [1]
	PDInputDescriptor	See B.5 and clause B.2.6 in [1]
	PDOutputDescriptor	See B.5 and clause B.2.7 in [1]
	ProductID	See clause B.2.11 in [1]
	SerialNumber	See clause B.2.12 in [1]
	HardWareRevision	See clause B.2.14 in [1]
	FirmwareRevision	See clause B.2.15 in [1]
	ApplicationSpecifictag	See B.2 and clause B.2.16 in [1]
	LocationTag	See B.6
	FunctionTag	See B.6
	DeviceStatus	See B.7 and clause B.2.20 in [1]
	DetailedDeviceStatus	See B.7 and clause B.2.21 in [1]

311

312 7.3 Extension of Identification and Diagnosis

313 The DeviceProfile Identification and Diagnosis may be accompanied by the FunctionClasses
 314 Locator, ProductURI optionally. The FunctionClass TeachRecommended is mandatory for "Data
 315 Storage class 2" Devices according 12.2.3 in [1]. The possible extensions are defined in Table
 316 9.

317

Table 9 – Extension for I&D

ProfileType	Possible extension	Associated parameter
I&D	Locator (0x8101), see A.6	SystemCommand, see B.2
	ProductURI (0x8102), see A.7	ProductURI, see B.8
	TeachRecommended (0x8103), see A.8	n.a.

318

319 7.4 Proxy Function Block for Identification and Diagnosis

320 To ease the integration in Run-Time systems like PLCs, an appropriate Function Block is spec-
 321 ified in C.1. The Function Block reads or writes identification or diagnosis data from the Device
 322 and shows the status of the Function Block. The information is provided in a way an operator
 323 can use directly in any PLC program for further handling. All specific action is taken without any
 324 required specific knowledge of the operator.

Annex A (normative)

FunctionClasses

A.1 Overview

Table A.1 provides an overview of the defined FunctionClasses within this document.

Table A.1 – Overview of FunctionClasses

FunctionClass	Name	Reference / Clause
[0x8000]	Device Identification	A.2
[0x8002]	Process Data Mapping (PDV)	A.3
[0x8003]	Device Diagnosis	A.4
[0x8100]	Extended Identification	A.5
[0x8101]	Locator	A.6
[0x8102]	ProductURI	A.7
[0x8103]	TeachRecommended	A.8

A.2 Device identification objects [0x8000]

The FunctionClass 0x8000 defines some optional parameters as mandatory for profile Devices. These are

- ProductID
- FirmwareRevision
- ApplicationSpecificTag

The ProductID and the FirmwareRevision are unchanged to the definition in clause B.2.11 and B.2.15 of [1]. The parameter ApplicationSpecificTag defined in clause B.2.16 in [1], is defined in this FunctionClass with the maximum size of 32 octets to get a maximum reusability over all profile Devices.

A.3 Process Data mapping (PDV) [0x8002]

A.3.1 Overview

Depending on the particular profile type, a Device arranges binary information and/or more complex data structures for the cyclic transmission to and/or from the Master via SDCI in a so-called "PDinput data stream" and/or "PDoutput data stream".

A.3.2 Process data description

The profile Device provides an input Process Data description (PDInputDescriptor) indicating the composition (mapping) in the "PDinput data stream" and/or a similar output Process Data description (PDOOutputDescriptor). In case multiple process data representations are supported, the description shall represent the currently selected process data structure.

The content of the process variable descriptors PVinD or PVoutD shall be available via the corresponding Index. The coding of the corresponding parameters is defined in B.5.

Each part of the process data stream is described unambiguously via its coding in PVinD and/or PVoutD. Subsequent Boolean variables are described within one descriptor. The following information shall be provided within a PVinD or PVoutD respectively:

- 359 • the data type (DataType) of the particular process variable. "Set of BoolT" describes com-
360 bined independent Boolean values
 - 361 • the length of the data type (TypeLength) in bit, for example 6 for UIntegerT6
 - 362 • the bit offset (Bit offset) as the beginning of the variable in the data stream
- 363 The user program within a controller (e.g. PLC) can thus read this information.

364 **A.4 Diagnosis [0x8003]**

365 The FunctionClass 0x8003 defines some optional parameters as mandatory for profile Devices.
366 These are

- 367 • DeviceStatus
- 368 • DetailedDeviceStatus

369 Both parameters are unchanged to the definition in clause B.2.20 and B.2.21 in [1].

370 As already described in [1], the Events, the DetailedDeviceStatus and the DeviceStatus are
371 interconnected. Whenever an Event appears, the DetailedDeviceStatus contains this Event until
372 it disappears, see B.2.21 in [1].

373 The priority column in Table A.2 defines which DeviceStatus value is signalled in case of mul-
374 tiple active events, the lowest priority value dominates higher priority values.

375

Table A.2 – DeviceStatus priority

Priority	Value	Definition
5	0	Device is operating properly
4	2	Out-of-Specification
3	1	Maintenance-Required
2	3	Functional-Check
1	4	Failure

376

377 **A.5 Extended Identification [0x8100]**

378 The FunctionClass 0x8100 defines extended identification which can be used e.g. for localiza-
379 tion in a plant, machine, etc in any readable location format. Another parameter can contain a
380 detailed description of the specific Device like "Hot water valve", etc. Both parameter provide
381 only a sequence of characters without any interpretation within the Device itself.

382 The parameter

- 383 • FunctionTag
- 384 • LocationTag

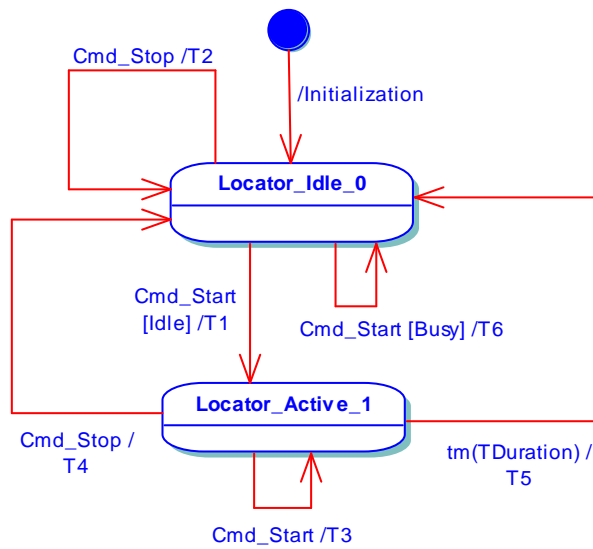
385 defined in B.6 provide the necessary non-volatile memory space.

386 Additionally the standardized parameter SerialNumber and HardwareRevision are set to man-
387 datory.

388 **A.6 Locator [0x8101]**

389 The FunctionClass 0x8101 defines the visual localization via an optical indicator within the De-
390 vice which can be used during setup of the installation to localize a specific Device among other
391 Devices.

392 Figure A.1 shows the state machine of the optical indication of a Device.



393

394

Figure A.1 – State machine for optical indication

395 Table A.3 shows the state transition tables for the optical indication.

396

Table A.3 – State transition tables for optical indication

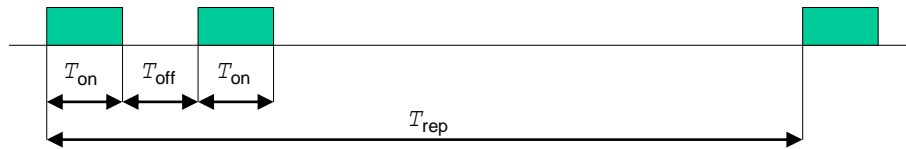
STATE NAME		STATE DESCRIPTION	
Locator_Idle_0		In this state the Device is waiting for a SystemCommand LocatorStart and performs no specific optical indication, timer is inactive	
Locator_Active_1		In this state the Device performs the specific optical indication according to Figure A.2 to allow easy identification of this Device until the LocatorStop command is received or the timeout elapses	
TRANSITION	SOURCE STATE	TARGET STATE	ACTION
Initialization	-	0	-
T1	0	1	Start optical indication, start timer
T2	0	0	-
T3	1	1	Restart timer
T4	1	0	Stop optical indication, stop timer
T5	1	0	Stop optical indication
T6	0	0	Indicate unavailability by responding "Function temporarily unavailable"
INTERNAL ITEMS	TYPE	DEFINITION	
Cmd_Start	Service	Reception of ISDU with SystemCommand containing LocatorStart	
Cmd_Stop	Service	Reception of ISDU with SystemCommand containing LocatorStop or SM_DeviceMode_IDLE (communication stopped) detected	
TDuration	Time	See Table A.4	
timer	Variable	Timeout timer	
Idle	Variable	Local user interface is in idle state, no interaction with user	
Busy	Variable	Local user interface is in interaction with user	

397

398

399 The indication of the Device localization follows the timing shown in Figure A.2.

400



401

402

Figure A.2 – Device optical indicator timing for localization

403 Table A.4 defines the timing for the optical indication of Devices.

404

Table A.4 – Timing for the optical indication

Timing	Minimum	Typical	Maximum	Unit
T_{rep}	900	1000	1100	ms
T_{on}	90	100	110	ms
T_{off}	90	100	110	ms
$T_{Duration}$	9	10	11	min

405

406 The sequence of repeated double flashing is started after reception of the SystemCommand
 407 "LocatorStart", see Table B.2, and lasts for the time $T_{Duration}$. After this time or by reception of
 408 a SystemCommand "LocatorStop" the sequence will stop. The timeout can be retriggered to
 409 $T_{Duration}$ by reception of another SystemCommand "LocatorStart".

410 The sequence shows a double flashing to avoid interference with other indications like output
 411 indicators.

412 It is on behalf of the Device designer to select the means for the optical indication like LEDs or
 413 graphical displays. The intended effect should be as outstanding as possible to enable the user
 414 to identify the selected Device as easy as possible. The standardized recognition of the IO-Link
 415 Device visual localization is based on the double flashing, and not on any specific color or
 416 symbol on any display.

417 The implementation of Locator is recommended for all Device which have controllable optical
 418 indications like LEDs or means like displays of any kind.

419

420 **A.7 ProductURI [0x8102]**

421 The FunctionClass 0x8102 provides a globally biunique ID of the Device according
 422 DIN SPEC 91406 [8]. The content is provided by a Product Unique Ressource Identifier
 423 (ProductURI). The structure and content is not defined here, any rules on the content are de-
 424 fined in [8].

425 The IO-Link ProductURI is restricted to 100 octets, which follows the restriction on printable
 426 data matrix codes on a device.

427 The parameter is defined in B.8, as being a read-only value there is no dynamic behaviour
 428 defined.

429 **A.8 TeachRecommended [0x8103]**

430 This FunctionClass does not define any special functionality. It indicates to the customer, that
 431 the Device may behave correct after transmission of all parameters via DataStorage, but the
 432 vendor recommends to perform teach procedures after replacement or implementation. The
 433 vendor may choose this FunctionClass when

- 434 • The Device is not calibrated and therefore differs between several devices,

- 435 • The application is depending on mechanical reproduction of the installation, which results
436 in different results,
- 437 • The application needs any type of position calibration,
- 438 • Or other reasons apply, and a teach is recommended by the vendor.
- 439 The vendor is obliged to state this requirement in comprehensive manner in the user manual
440 by statements like

This Device performs best if a teach is applied after installation, even if DataStorage is performed.

Annex B (normative)

Profile relevant Device parameters

B.1 Overview

The manufacturer may provide Subindex access to objects with RecordItems, the Common Profile specification does not define this behaviour. Any overall usable software shall always use the Subindex 0 access instead as this access is granted by any Device.

The persistence or volatility of the objects is stated for each object.

The Device reset option rules defined in clause 10.7.1 in [1] shall be considered and reset all Device parameters to their default value.

The profile relevant Device parameters are specified in [1]. An overview is shown in Table B.1.

Table B.1 – General profile relevant Device parameters

Index (dec)	Object name	Access	Length	Data type	M/C	Remark
...						
0x0002 (2)	SystemCommand	W	1 octet	UIntegerT	M	See B.2
...						
0x000D (13)	ProfileCharacteristic	R	variable	ArrayT of UIntegerT16	M	See B.4 See clause B.2.5 in [1]
0x000E (14)	PDInputDescriptor	R	variable	ArrayT of OctetStringT3	C	Conditional on availability of PDIn See B.5 and clause B.2.6 in [1]
0x000F (15)	PDOOutputDescriptor	R	variable	ArrayT of OctetStringT3	C	Conditional on availability of PDOOut See B.5 and clause B.2.7 in [1]
0x0010 (16)	VendorName	R	max. 64 octets	StringT	M	See clause B.2.8 in [1], a default value shall be provided in the IODD
...						
0x0012 (18)	ProductName	R	max. 64 octets	StringT	M	See clause B.2.10 in [1], a default value shall be provided in the IODD
0x0013 (19)	ProductID	R	max. 64 octets	StringT	M	See clause B.2.11 in [1]
...						
0x0015 (21)	SerialNumber	R	max. 16 octets	StringT	M	See clause B.2.13 in [1]
0x0016 (22)	HardwareRevision	R	max. 64 octets	StringT	M	See clause B.2.14 in [1]
0x0017 (23)	FirmwareRevision	R	max. 64 octets	StringT	M	See clause B.2.15 in [1]
0x0018 (24)	ApplicationSpecific-Tag	R/W	32	StringT	M	See B.2 See clause B.2.16 in [1]
0x0019 (25)	FunctionTag	R/W	32	StringT	M	See B.6
0x001A (26)	LocationTag	R/W	32	StringT	M	See B.6
0x001B (27)	ProductURI	R	100	StringT (US-ASCII)	C	Conditional on support of FunctionClass 0x8102. See B.8
...						

Index (dec)	Object name	Access	Length	Data type	M/C	Remark
0x0024 (36)	DeviceStatus	R	1 octet	UIntegerT	M	See B.7 See clause B.2.20 in [1], default value is "0".
0x0025 (37)	DetailedDeviceStatus	R	variable	ArrayT of OctetStringT3	M	See B.7 See clause B.2.21 in [1], default values are "0". Contains a minimum of one Event entry.
Keys	M = mandatory C = conditional R = read W = write					

454

455 B.2 System Commands

456 This clause describes the SystemCommands which are used by the CommonProfile. The avail-
457 ability of the commands specified in Table B.2 is depending on the support of the corresponding
458 FunctionClass.

459 **Table B.2 – Conditional "SystemCommand"**

Command (hex)	Command (dec)	Command name	Definition
0x7E	126	Locator Start	Applicable for Locator, see A.6
0x7F	127	Locator Stop	

460

461 The reaction to the SystemCommands "FlashStart" and FlashStop" shall meet the requirement
462 in clause 10.3.7 in [1] with an immediate return after checking the availability of the command.

463 B.3 Identification parameters

464 As identification parameters ProductID, FirmwareRevision, and ApplicationSpecificTag are de-
465 fined as mandatory, the structure and coding is defined in clauses B.2.11, B.2.15 and B.2.16 in
466 [1].

467 As a difference the parameter ApplicationSpecificTag is defined with the maximum size of 32
468 octets as defined in Table B.3. The object shall be stored persistent, follows the Device reset
469 option rules defined in clause 10.7.1 in [1]. and handled by the DataStorage mechanism.

470 **Table B.3 – Definitions for identification data objects**

Index (dec)	Subindex	Offset	Access	Object name	Length (octets)	Data Type
0x0018 (24)	n/a	n/a	R/W	ApplicationSpecificTag	32	StringT
Keys	R = read W = write					

471

472 B.4 ProfileCharacteristics parameter

473 This clause describes the parameter which contains the ProfileIdentifier of the supported Device
474 profiles and FunctionClasses.

475 Table B.4 defines the structure of the parameter ProfileCharacteristics.

476

Table B.4 – Parameter "ProfileCharacteristic"

Index (dec)	Subindex (dec)	Offset	Access	Parameter Name	Length	Data type
0x000D (13)	1	(n-1) * 2	R	ProfileIdentifier 1	16 bit	UIntegerT16
	...					
	n	0	R	ProfileIdentifier n		
Keys n = number of supported ProfileIdentifier R = read						

477

478 Figure B.1 specifies the rules which apply to the ProfileIdentifier in the ProfileCharacteristic
 479 parameter.

- 1) Whenever 1 to n Device profiles are supported, they shall be indicated via 1 to n DeviceProfileID entries
- 2) Whenever 1 to n common application profiles are supported, they shall be indicated via 1 to n CommonApplicationProfileIDs
- 3) Additionally supported FunctionClasses which are not covered by DeviceProfileIDs or CommonApplicationProfileIDs shall be indicated by 1 to n FunctionClassIDs
- 4) The IDs shall be listed in ascending order

480

481 **Figure B.1 – Indication rules for ProfileIdentifiers**

482 **B.5 Process data structure descriptors**

483 This clause describes the parameters which contain the structure information of the process
 484 data input and output. Each part of the process data is described with an PVinD or PVoutD.
 485 The generic rules for defining the structures are described in A.3, specific process data struc-
 486 ture definitions for ProfileIDs are defined in the corresponding profile specification like [6].

487 **B.5.1 Coding of PVinD and PVoutD**

488 Table B.5 shows the coding of each process variable to be placed in the descriptors using
 489 PVinD or PVoutD.

490

Table B.5 – Coding of PVinD or PVoutD

Location	Item	Coding
Octet 1	DataType	0: OctetStringT 1: Set of BoolT 2: UIntegerT 3: IntegerT 4: Float32T 5: StringT 6: TimeT 7: TimeSpanT 8 to 127: reserved: 128 to 255 reserved for profiles
Octet 2	TypeLength	0: 256 Bit 1 to 255: 1 to 255 Bit
Octet 3	Bit offset	0 to 255 Bit

491

492 NOTE The abstract notation of for example a PVinD is: DataType.TypeLength.Bit_offset
 493 Set of BoolT describes a combination of one or more BooleanT without gaps

494 Any profile may define their own complex DataTypes if necessary. Reuse of same DataType
 495 definitions is mandatory.

496 In case a currently selected process data representation does not provide any content, the
 497 PVinD or PVoutD shall return a reponse with length 'zero' (empty octet string).

508 B.5.2 PDInputDescriptor

509 Profile Devices with process input data shall use the standard Device parameter "PDIn-
500 putDescriptor" in Index 0x000E to provide the description information according to Table B.5.

501 The PVinD descriptors shall be sorted in ascending "Bit offset" order.

502 Table B.6 defines the structure of the PDInputDescriptor regarding the offset and Subindex
503 layout.

504 **Table B.6 – Structure of "PDInputDescriptor"**

Index (dec)	Subindex (dec)	Offset	Access	Parameter Name	Length	Data type
0x000E (14)	1	$(n-1) * 3$	R	PVinD 1	24 bit	OctetStringT3
	...					
	n	0	R	PVinD n	24 bit	OctetStringT3
Keys	n = number of provided descriptors R = read					

505

506 B.5.3 PDOutputDescriptor

507 Profile Devices with process data output shall use the standard Device parameter "PDOut-
508 putDescriptor" in Index 0x000F to provide the description information according to Table B.5.

509 The PVoutD descriptors shall be sorted in ascending "Bit offset" order.

510 Table B.7 defines the structure of the PDOutputDescriptor regarding the offset and Subindex
511 layout.

512 **Table B.7 – Structure of "PDOutputDescriptor"**

Index (dec)	Subindex (dec)	Offset	Access	Parameter Name	Length	Data type
0x000F (15)	1	$(n-1) * 3$	R	PVoutD 1	24 bit	OctetStringT3
	...					
	n	0	R	PVoutD n	24 bit	OctetStringT3
Keys	n = number of provided descriptors R = read					

513

514 B.6 Extended Identification parameters

515 This clause defines the extended identification parameters which can be used for overall local-
516 ization and identification of any Device.

517 The content is not predefined, the customer can provide any visible string conform to his own
518 naming rules. The R/W parameters "FunctionTag" and "LocationTag" shall be stored persistent,
519 follows the Device reset option rules defined in clause 10.7.1 in [1], and handled by the DataS-
520 torage mechanism. As default it is recommended to fill the parameter "FunctionTag" and "Lo-
521 cationTag" with "****".

522 Table B.8 defines the structure of the parameters.

523

Table B.8 – Parameter Extended Identification

Index (dec)	Subindex (dec)	Offset	Access	Parameter Name	Length	Data type
0x0015 (21)	n/a	n/a	R	SerialNumber	Max 16 octets	StringT
0x0016 (22)	n/a	n/a	R	HardwareRevision	Max 64 octets	StringT
0x0019 (25)	n/a	n/a	R/W	FunctionTag	32 octets	StringT32
0x001A (26)	n/a	n/a	R/W	LocationTag	32 octets	StringT32
Keys	n/a = not applicable R = read W = write					

524

B.7 Diagnosis parameters

525 The structure and coding is defined in clauses B.2.20 and B.2.21 in [1].

526 Table B.9 defines the structure of the DetailedDeviceStatus regarding the offset and Subindex layout.

527

Table B.9 – Structure of "DetailedDeviceStatus"

Index (dec)	Subindex (dec)	Offset	Access	Parameter Name	Length	Data type
0x0025 (37)	1	$(n-1) * 3$	R	Event 1	24 bit	OctetStringT3
	...					
	n	0	R	Event n	24 bit	OctetStringT3
Keys	n = number of provided Event entries R = read					

528

B.8 ProductURI parameter

529 This clause defines the parameter containing the globally biunique ID according [8]. The following restrictions on the content shall be considered:

- 530 • Content structure according rules in [8]
- 531 • Max length 100 octets
- 532 • Coding in US-ASCII, consider restrictions defined in table A.1 in [8]

533 Table B.10 defines the structure of the ProductURI regarding the offset and Subindex layout.

534

Table B.10 – Definitions for ProductURI parameter

Index (dec)	Subindex	Offset	Access	Object name	Length (octets)	Data Type
0x001B (27)	n/a	n/a	R	ProductURI	100	StringT (US-ASCII)
Keys	n/a = not applicable R = read					

535

Annex C (normative)

Function block definitions

C.1 Overview

This annex contains the proxy Function Blocks supporting the CommonApplicationProfileID.

The specification is based on IEC 61131-3 definitions.

As there are still some differences between the existing systems regarding the PLC system or fieldbus, the system dependent features are marked and have to be defined for each system separately.

The proxy Function Block is asynchronous, which means that the Function Block is triggered and after accomplishing the functionality the results are available.

The access of acyclic parameters in IO-Link Devices requires the usage of BlockParametrization according to 10.3.5 in [1] by following these steps

- Hidden SystemCommand "ParamUploadStart" or "ParamDownloadStart" depending on direction
- Perform acyclic parameter access
- SystemCommand "ParamUploadEnd" or "ParamDownloadEnd" depending on direction
- SystemCommand "ParamDownloadStore" if parameters were written and BackupEnable = "true"

C.2 Proxy function block (FB) for identification and diagnosis

The layout of the proxy function block for the CommonApplicationProfile Identification and Diagnosis (0x4000) which supports the FunctionClasses DeviceIdentification (0x8000), Device-Diagnosis (0x8003), and ExtendedIdentification (0x8100) is shown in Figure C.1.

The input and output data types of the proxy function block correspond to those of IEC 61131-3 (PLC programming languages).

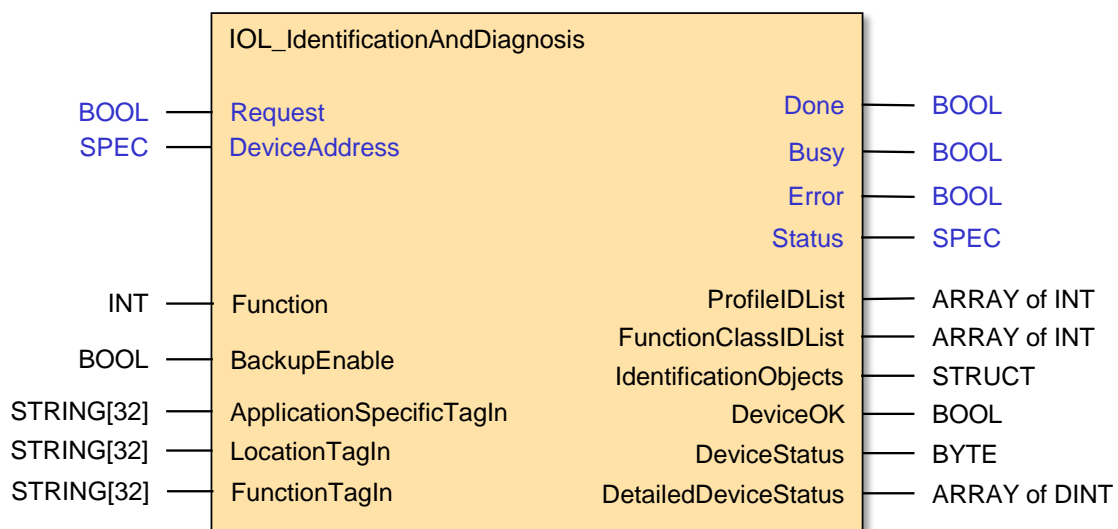


Figure C.1 – Proxy FB for Device Identification and Diagnosis

Table C.1 defines the variables of this proxy FB.

570

Table C.1 – Variables of "IOL_IdentificationAndDiagnosis" FB

Variable	PLC Type	Description
Inputs		
Request ^a	BOOL	A trigger causes the function selected with variable Function to be executed
DeviceAddress ^a	SPEC ^b	This variable depends on the individual fieldbus address mechanism of an SDCI Device at an SDCI Master port (see SDCI integration specification of a particular fieldbus)
Function	INT	This variable selects the functionality to be triggered by a Request 0 = no_func A Request is neglected, no function is executed 1 = rd_all A Request starts the read back of current identification and diagnostic parameter values from the Device. 2 = rd_diag A Request starts the read back of current diagnostic parameter values by reading DeviceStatus and DetailedDeviceStatus from the Device. 3 = wr_ident A Request causes a previously applied value for ApplicationSpecificTagIn, LocationTagIn, and FunctionTagIn to be written to the Device
BackupEnable	BOOL	This variable configures the behavior of the FB in case of the requested function wr_ident. "true" = enabled The backup mechanism is triggered by the FB by issuing the SystemCommand ParamDownloadStore after wr_ident. "false" = disabled The backup mechanism is not triggered by the FB
ApplicationSpecificTagIn	STRING[32]	See Device parameter in clause B.2.16 in [1]
LocationTagIn	STRING[32]	See B.6
FunctionTagIn	STRING[32]	See B.6
Outputs		
Done ^a	BOOL	The signal is set, if the FB has completed a requested operation.
Busy ^a	BOOL	The signal is set, if the FB is executing a requested operation
Error ^a	BOOL	The signal is set, if an error occurred during execution of a requested operation.
Status ^a	SPEC ^b	The value represents the current status of the FB operation and executed functions. The content is system specific and contains the status information
ProfileIDList	ARRAY of INT	List of ProfileIDs supported by the Device
FunctionClassIDList	ARRAY of INT	List of FunctionClassIDs supported by the Device
IdentificationObjects	STRUCT	Structured list of identification objects, see Table C.2 for further details
DeviceOK	BOOLEAN	The signal is set when no further diagnosis info is available, it is false when further information is available at DeviceStatus and DetailedDeviceStatus
DeviceStatus	BYTE	See Device parameter in clause B.2.20 in [1]
DetailedDeviceStatus	ARRAY of DWORD	This parameter contains the type casted values from the Device parameter defined in clause B.2.21 in [1]
Keys a: This variable name may be adapted to the PLC specific naming guide lines b: SPEC represents the applicable data type for this specific parameter, this can vary over different PLC systems		

571
572

The lists ProfileIDList, FunctionClassIDList, and DetailedDeviceStatus are set to 0 by default and are overwritten by data read from the Device.

573 The structured information in the variable IdentificationObjects is specified in Table C.2.

574 The default value is provided when the corresponding parameter is not already read from the
575 Device or not available in the Device.

576 **Table C.2 – Elements of the IdentificationObjects**

Name	PLC Type	Default	Remark
VendorID	WORD	00 00	See clause B.1.8 in [1]
DeviceID	DWORD	00 00 00 00	See clause B.1.9 in [1]
VendorName	STRING[64]	"na"	See clause B.2.8 in [1]
VendorText	STRING[64]	"na"	See clause B.2.9 in [1]
ProductName	STRING[64]	"na"	See clause B.2.10 in [1]
ProductID	STRING[64]	"na"	See clause B.2.11 in [1]
ProductText	STRING[64]	"na"	See clause B.2.12 in [1]
SerialNumber	STRING[16]	"na"	See clause B.2.13 in [1]
HardwareRevision	STRING[64]	"na"	See clause B.2.14 in [1]
FirmwareRevision	STRING[64]	"na"	See clause B.2.15 in [1]
ApplicationSpecificTag	STRING[32]	"na"	See clause B.2.16 in [1]
LocationTag	STRING[32]	"na"	See B.6
FunctionTag	STRING[32]	"na"	See B.6
ProductURI	STRING[100]	"na"	See B.8

577

Annex D
(normative)

IODD definition and rules

D.1 Overview

The objective of the Common Profile specification is to ease the integration of Devices and to provide additional information in a uniformed manner. The integration is part of the specialised profile specifications, the uniformed information about profile support is part of this clause. As the parameter and the behavior is specified, the look and feel of the Devices should also be harmonized, otherwise the appearance of the same profile is different between different manufacturers.

To achieve a common look and feel, the IODD content of the Identification and Diagnosis profile with its extensions has to be defined as well. This clause contains the IODD predefinitions.

D.2 Name definitions

D.2.1 Profile type characteristic names

The profile characteristic name defined in 7.2, A.1, and in separated profile specifications shall be used whenever any profile functionality is referenced in the IODD.

D.3 IODD Menu definitions

D.3.1 Overview

Examples for layouts of Port and Device configuration tools are shown in clause 13.5.3 in [1].

Within these examples the IODD defines the parameter layout of the connected device. In this clause the object and parameter layout of the ProfileIdentifier is specified.

It is mandatory to provide all defined parameters in the defined order, it is on behalf of the manufacturer to group them by menu groups or extend by further parameters.

D.3.2 Explanation of used object layout

Figure D.1 shows the basic layout objects to describe the look of the profile parameters in any IODD based tooling.

The content description is placed at the corresponding positions.

Sub menu header		
Parameter name (selectable value)	Selection	v
Parameter name (value)	Value	
Command (Triggered action)	Command name	
Parameter name (read only)	Value / Selection	

Figure D.1 – IODD object layout description

D.3.3 Menu structure of the Device Diagnosis parameter

In Figure D.2 the menu structure Device Diagnosis according to 7.2 and A.4 is specified, it shall be located in the Diagnosis section directly or in a sub menu part of this section.

- Diagnosis	
Device Status	Status Text ¹
- Detailed Device Status	
[1]	Detailed Status Text ²
[2]	Detailed Status Text ²
...	
[3]	Detailed Status Text ²

Note 1 = One of the texts according to STD_TN_DeviceStatus_xxx in [3]

2 = One of the texts according to STD_TN_0xnnnn in [3]

612

613

Figure D.2 – Menu Profile Diagnosis

614 The texts are already defined in [2] as standard parameter texts.

615 D.3.4 Menu structure of the Device Identification parameters

616 In Figure D.3 the menu structure Device Identification according to 7.2, A.2, and A.5 is speci-
617 fied, it shall be located in the Identification section directly or in a sub menu part of this section.

- Identification	
Vendor Name	Text
VendorText ¹	Text
Product Name	Text
Product Text ¹	Text
Product ID	Text
Serial Number	Text
ProductURI	Text
Hardware Version	Text
Firmware Version	Text
Application Specific Tag	Text
Function Tag	Text
Location Tag	Text

Note 1 = optional parameter

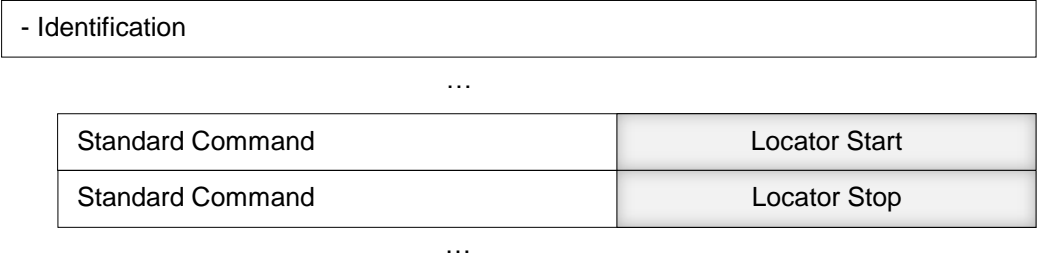
618

619

Figure D.3 – Menu Profile Identification

620 D.3.5 Menu structure of the Locator functionality

621 In Figure D.4 the menu structure Device localization according to A.6 is specified, it shall be
622 located in the "Diagnose/ServiceFunctions" section of the menu.



623

624

Figure D.4 – Menu Profile Locator

625
626
627

Annex E (normative) **Profile testing and conformity**

628 **E.1 General**

629 **E.1.1 Overview**

630 It is the responsibility of the vendor/manufacture of a profile Device to perform a conformity
631 testing according to the test specification [5] and to provide a document similar to the manufac-
632 turer declaration defined in [1] or based on the template downloadable from the IO-Link website
633 (www.io-link.com).

634 **E.1.2 Test extension for profile Devices**

635 The standard test cases to achieve the conformity are extended by profile test cases specified
636 in Annex F.

637 **E.1.3 Business rule extensions for the IODD Checker**

638 To achieve consistency and conformity of the profiled Devices to the claimed profiles, the busi-
639 ness rules of the checker are extended covering the profile requirements. This predefinitions
640 are defined in so-called IODD snippet files, which provide all necessary information to cover
641 the ceration and the test of the profile Device's IODD.

642 The rule extensions are generic to suit the profile requirements and based on IODD snippets
643 which are provided together with this profile specifications.

644 This profile provides xml based files containing IODD related snippets, which can be copied
645 and adapted to create well formed Device IODDs. These xml files contain xml elements follow-
646 ing the rules of [2] which are extended by test related attributes. This specific extensions must
647 be removed when copying the parts into a specific Device IODD.

Annex F (normative)

Testing Identification and Diagnosis

F.1 Test case extension for static parameter design

F.1.1 Ordering of Profile characteristics

Table F.1 defines the test conditions for this test case.

Table F.1 – Ordering of Profile characteristics

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	CP_TC_0001
Name	TCD_CMPR_ID_ASCENDID
Purpose (short)	Consistence and ascending order of supported ProfileIDs
Equipment under test (EUT)	Device, IODD; ProfileCharacteristics not empty
Test case version	1.0
Category / type	Parameter verification test; test to pass (positive testing)
Specification (clause)	B.4
Configuration / setup	Device-Tester-Unit
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Consistency between Device and IODD of supported ProfileIDs and test of ascending order of the ProfileIDs
Precondition	Master and Device in Operate
Procedure	a) Read parameter ProfileCharacteristic and memorize b) Read from IODD /IODevice/ProfileBody/DeviceFunction/Features/@profileCharacteristic
Input parameter	-
Post condition	-
TEST CASE RESULTS	CHECK / REACTION
Evaluation	1. Check after step a) for positive result 2. Match ProfileIDs from Device against IODD content 3. Check ProfileIDs for ascending order
Test passed	Evaluation from 1) to 3) without failure
Test failed (examples)	Any failure in 1) to 3)
Results	Consistence and order of ProfileIDs < ok/nok >

659 **F.1.2 Hiding FunctionClasses by ProfileIDs**

660 Table F.2 defines the test conditions for this test case.

661 **Table F.2 – Hiding FunctionClasses of I&D**

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	CP_TC_0002
Name	TCD_CMPR_ID_HIDDEN_I_D
Purpose (short)	Already incorporated FunctionClasses by higher ProfileID 0x4000 shall not be listed
Equipment under test (EUT)	Device, IODD supporting ProfileID 0x4000
Test case version	1.0
Category / type	Parameter verification test; test to pass
Specification (clause)	Table 7, B.4
Configuration / setup	Device-Tester-Unit
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	All already by the CommonApplicationProfileID 0x4000 incorporated FunctionClasses as 0x8000, 0x8002, 0x8003, and 0x8100 shall not be listed in the ProfileCharacteristic.
Precondition	Master and Device in Operate
Procedure	a) Read parameter ProfileCharacteristic
Input parameter	–
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) Check for positive result b) Check for absence of entries of intrinsic FunctionClasses 0x8000, 0x8002, 0x8003, and 0x8100
Test passed	Evaluation from 1) to 2) without failure
Test failed (examples)	Any failure in 1) to 2)
Results	Intrinsic FunctionClasses of I&D hidden < ok/nok >

664

665

666 **F.1.3 Minimum required profile support**

667 Table F.3 defines the test conditions for this test case.

668 **Table F.3 – Minimum required profile support**

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	CP_TC_0003
Name	TCD_CMPR_ID_LEASTPROFILE
Purpose (short)	Test if required ProfileID 0x4000 is supported
Equipment under test (EUT)	Device, IODD; ProfileCharacteristic not empty
Test case version	1.0
Category / type	Parameter verification test; test to pass
Specification (clause)	Table 7
Configuration / setup	Device-Tester-Unit
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Check for availability of ProfileID 0x4000 (Identification and Diagnosis) when at least one other ProfileID is listed
Precondition	Master and Device in Operate
Procedure	a) Read parameter ProfileCharacteristic
Input parameter	–
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) Check for positive result b) Check for presence of 0x4000
Test passed	Evaluation from 1) to 2) without failure
Test failed (examples)	Any failure in 1) to 2)
Results	Identification and Diagnosis supported < ok/nok >

671

672 **F.1.4 Extensions of profiles**

673 Table F.4 defines the test conditions for this test case.

674 **Table F.4 – Extension of Identification and Diagnosis**

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	CP_TC_0004
Name	TCD_CMPR_ID_EXTENSION
Purpose (short)	Test correct extension for Identification and Diagnosis
Equipment under test (EUT)	Device supporting FunctionClasses Locator or ProductURI
Test case version	1.0
Category / type	Parameter verification test; test to pass
Specification (clause)	7.3
Configuration / setup	Device-Tester-Unit
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Test availability and correct relation of the extending FunctionClasses to the Identification and Diagnosis profile. The extensions are only allowed in combination with the CommonApplicationProfile 0x4000, Identification and Diagnosis
Precondition	Master and Device in Operate
Procedure	a) Read parameter ProfileCharacteristic
Input parameter	-
Post condition	-
TEST CASE RESULTS	CHECK / REACTION
Evaluation	1. Check for positive result in a) 2. Check if ProfileID 0x4000 is listed
Test passed	Evaluation from 1) to 2) without failure
Test failed (examples)	Any failure in 1) to 2)
Results	a) Extension available < ok/nok >

677

678 **F.1.5 PDInput-, PDOOutputDescriptor parameter**

679 Table F.5 defines the test conditions for this test case.

680 **Table F.5 – PDInput-, PDOOutputDescriptor parameter**

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	CP_TC_0005
Name	TCD_CMPR_ID_PDOUTDESCR
Purpose (short)	Test correct description of PDInput- and PDOOutputDescriptor
Equipment under test (EUT)	Device supporting Identification and Diagnosis profile
Test case version	1.0
Category / type	Parameter verification test; test to pass
Specification (clause)	A.3.1, B.5
Configuration / setup	Device-Tester-Unit
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Test availability and correct structure of provided process data description. This test checks for general compliance to the rules in B.5. If any specific profile requires a dedicated process data layout, this is tested by a specific test of this profile.
Precondition	Master and Device in Operate
Procedure	a) Read parameter PDInputDescriptor, zero length indicates no content within the process data b) Read parameter PDOOutputDescriptor, zero length indicates no content within the process data c) Read from IODD /IODevice/ProfileBody/DeviceFunction/ProcessDataCollection, read condition variable when available and perform condition setting for multiple process data layouts, compile ProcessDataIn and ProcessDataOut layout
Input parameter	-
Post condition	-
TEST CASE RESULTS	CHECK / REACTION
Evaluation	1. if /IODevice/ProfileBody/DeviceFunction/ProcessDataCollection/ProcessDataIn is existent and PDInDescriptor does not contain profile specific data types, then check PDInputDescriptor from a) against ProcessDataIn. 2. if /IODevice/ProfileBody/DeviceFunction/ProcessDataCollection/ProcessDataIn is not existent then check if read access at a) returns ErrorType 0x8011 or returns zero length 3. if /IODevice/ProfileBody/DeviceFunction/ProcessDataCollection/ProcessDataOut is existent and PDOOutDescriptor does not contain profile specific data types, then check against PDOOutputDescriptor from b) 4. if /IODevice/ProfileBody/DeviceFunction/ProcessDataCollection/ProcessDataOut is not existent then check if read access at b) returns ErrorType 0x8011 or returns zero length
Test passed	Evaluation from 1) to 4) without failure
Test failed (examples)	Any failure in 1) to 4)
Results	a) PDIn description available < ok/nok > b) PDOOut description available < ok/nok >

683

684 **F.2 Test case extension for dynamical behavior**685 **F.2.1 Device localization commands**

686 Table F.6 defines the test conditions for this test case.

687 **Table F.6 – Device localization commands**

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	CP_TC_0006
Name	TCD_CMPR_ID_LOCATOR
Purpose (short)	Test if localization commands are performed adequate.
Equipment under test (EUT)	Device, IODD supporting Locator (0x8101)
Test case version	1.0
Category / type	Parameter verification test; test to pass
Specification (clause)	A.6
Configuration / setup	Device-Tester-Unit
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Test if the SystemCommands for Locator are responded with the correct return code. A Write response (+) shall be returned.
Precondition	Master and Device in Operate, Locator and local display in Idle state
Procedure	a) Write request to SystemCommand with 0x7E "Locator Start" b) Wait 5 sec c) Write request to SystemCommand with 0x7E "Locator Start" d) Wait 5 sec e) Write request to SystemCommand with 0x7F "Locator Stop" f) Wait 2 sec g) Write request to SystemCommand with 0x7E "Locator Start" h) Wait 5 sec i) If SIO supported, perform communication stop via Fallback j) Wait 5 sec
Input parameter	-
Post condition	-
TEST CASE RESULTS	CHECK / REACTION
Evaluation	1. Check after step a) for positive response 2. Check after step c) for positive response 3. Check for visualization scheme on Device 4. Check after step e) for positive response 5. Check manually the visualization scheme on Device during steps b) to e) and h) 6. Check Locator idle during step f) 7. If SIO supported, check Locator idle during step j) 8. Check manually the "Locator" timeout of 10 min
Test passed	Evaluation in 1) to 8) without failure
Test failed (examples)	Any failure in 1) to 8)
Results	a) Locator control < ok/nok > b) Localization visible < ok/nok >

690

691

692

Bibliography

- 693 [1] IO-Link Community, *IO-Link Interface and System*, V1.1.3, June 2019, Order No.
694 10.002
- 695 [2] IO-Link Community, *IO Device Description (IODD)*, V1.1.3, January 2021, Order No.
696 10.012
- 697 [3] IEC/TR 62390:2005, *Common automation device profile guideline*
- 698 [4] IEC 60050 (all parts), *International Electrotechnical Vocabulary*
- 699 [5] IO-Link Community, *IO-Link Test Specification*, V1.1.3, January 2021, Order No.
700 10.032
- 701 [6] IO-Link Community, *IO-Link Smart Sensor Profile Ed.2*, V1.1, September 2021, Order
702 No. 10.042
- 703 [7] ISO/IEC 19505-2:2012, Information technology – Object Management Group Unified
704 Modeling Language (OMG UML) – Part 2: Superstructure
- 705 [8] DIN SPEC 91406:2019-12, Automatic identification of physical objects and information
706 on physical objects in IT systems, particularly IoT systems;

707

© Copyright by:

IO-Link Community

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/>



IO-Link