IO-Link

IO-Link
Common Profile

Specification

Draft Version 1.1.05
July 2023

Order No: 10.072

IO-Link

File name: **IO-Link_CommonProfile_10072_d1105_Jul23**

The IO-Link community created this update to cover the change requests on V1.1 of December 2021.

Deadline for this review is September 30, 2023.

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.

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
Haid-und-Neu-Str. 7
76131 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

0 Introduction

0.1 General

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

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

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

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

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

0.2 Patent declaration

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

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

---

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

Scope

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

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

Normative references

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.

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

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

Terms, definitions, symbols, abbreviated terms and conventions

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

3.1 CommonProfile: Additional terms and definitions

3.1.1
BinaryDataChannel
BDC

binary information as switching or control signal

3.1.2
Function block
software functional element comprising an individual, named copy of a data structure and associated operations specified by a corresponding function block type

[SOURCE: IEC 62390, 3.1.13]

3.1.3
FunctionClass
particular function within a Device profile

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

3.1.4
not applicable
n/a
this entry cannot be applied within this context

3.1.5
ProfileIdentifier
unique identifier for Device Profile, CommonApplicationProfile, or FunctionClass

3.2 Symbols and abbreviated terms

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

3.3 Conventions

3.3.1 Behavioral descriptions

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

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

Objectives for Device profiles

4.1 Purpose of Device profiles

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

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

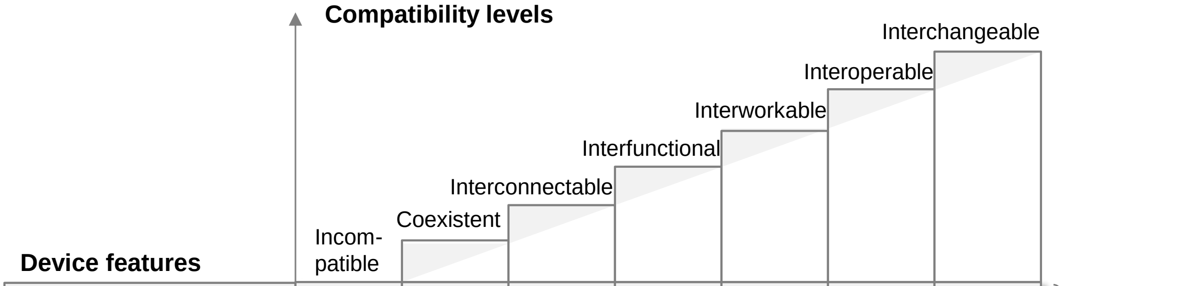
It is the purpose of SDCI to overcome the limitations of the classic Device interfaces DI, DO, AI, and AO via a point-to-point digital communication that allows transmitting digitally not only binary and/or analog information but additional information also. Very often, the changes to the core sensor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

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

Device profiles define terminologies, features, behaviours, commands, responses, corresponding data structures, and other things common to particular Device families and thus prevent the user from a confusing variety.

4.2 Interoperability

The major parts of the Device profiles deal with process data structures and behavior as well as parameter data structures and dynamic parameterization at runtime. These features streamline the functions of comparable Devices though requiring more sophisticated and powerful PLC user programs. Thus, interoperability between existing user programs and Devices of a corresponding family is the goal of profile Device design and testing (see [3]). Figure 1 shows compatibility levels based on IEC 62390.



Device features	Incom-patible	Coexistent	Interconnectable	Interfunctional	Interworkable	Interoperable	Interchangeable	
Mechanics							X	Device profiles
Dynamic behavior						(X)	(X)	
Application functionality						X	X	
Parameter semantics					X	X	X	
Data structures				X	X	X	X	
Data types				X	X	X	X	SDCI communi-cation
Data access			X	X	X	X	X	
Communication interface			X	X	X	X	X	
Communication protocol		X	X	X	X	X	X	

Figure 1 – Compatibility levels based on IEC 62390

The different compatibility levels are described in Table 1 and the different Device features are 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 servie 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
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	

4.3 Common Profile Specification structure

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

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

Clause 0 specifies the Identification and Diagnosis profile. The Annex A defines the FunctionClasses, 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.

Device profiles related to IEC 61131-9

5.1 SDCI technology specified in IEC 61131-9

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

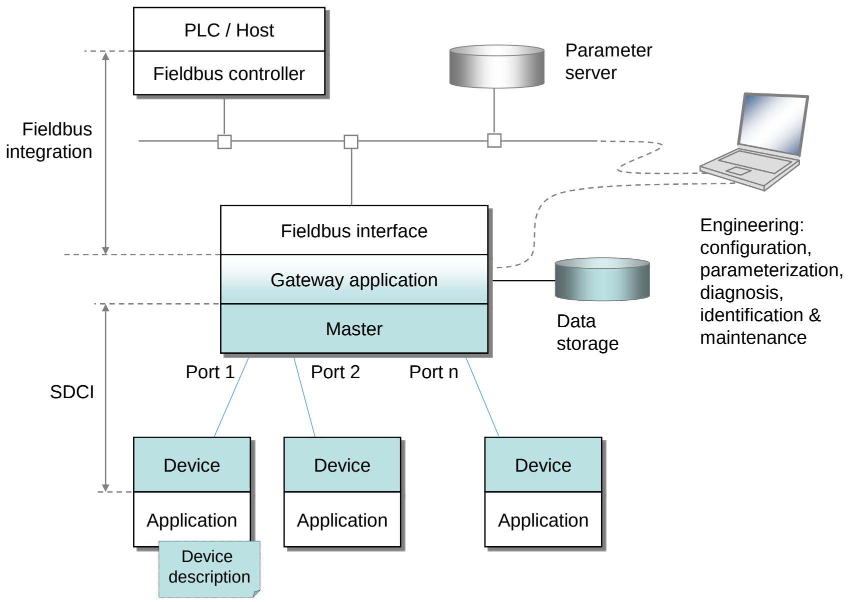


Figure 2 – Domain of the SDCI technology within the automation hierarchy

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

5.2 Profile classification

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

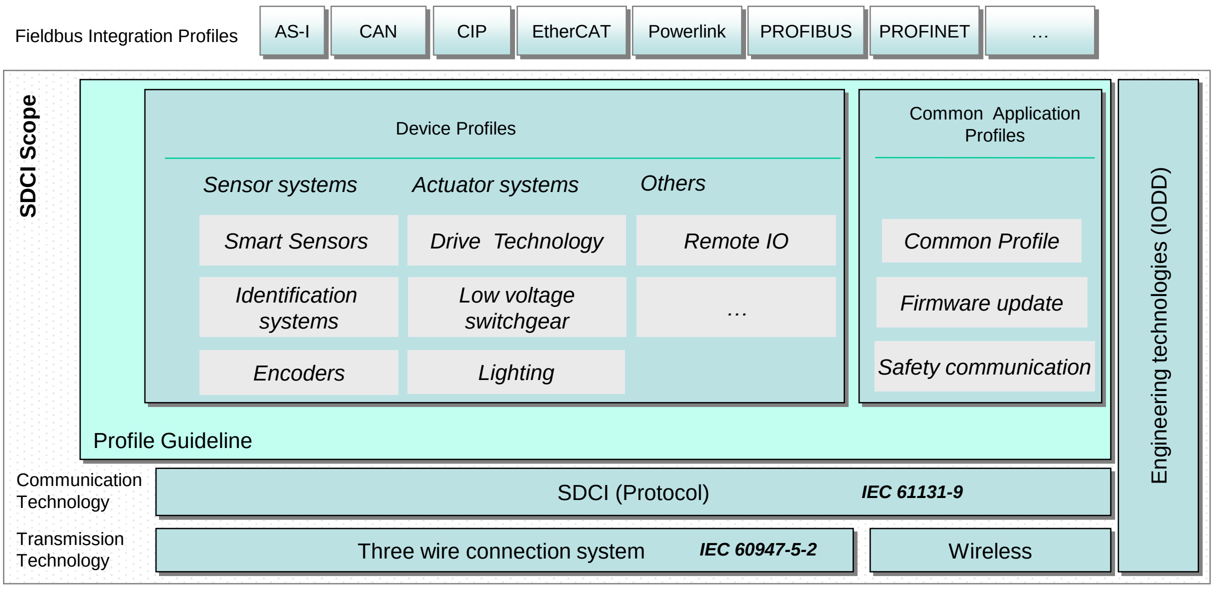


Figure 3 – Overview of SDCI technologies and profiles

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

- smart sensors,
- smart actuators,
- lighting
- etc.

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

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

- Common Profile
- Firmware update
- Safety communication
- etc.

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

5.3 Concept of profiles

The approach for profiling SDCI Devices follow the guideline to enforce as many equal functionality or behavior in all SDCI Devices. It is a stacked approach by defining a common subset which are highly recommended for all Devices like defined in the basic requirements. Furthermore the application specific profiles such as smart sensors or specific actuators define application specific requirements and solutions.

5.3.1 Basic requirements for profile Devices

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

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

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

5.3.2 Distinct profiles

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 ProfileIdentifer of the Device, this provides a higher level of interchangeablility even between different manufacturers.

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.

5.4 Profile characteristics

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

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

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).

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.

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

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

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

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

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

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

For further details, see B.4.

5.5 Concept of FunctionClasses

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

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

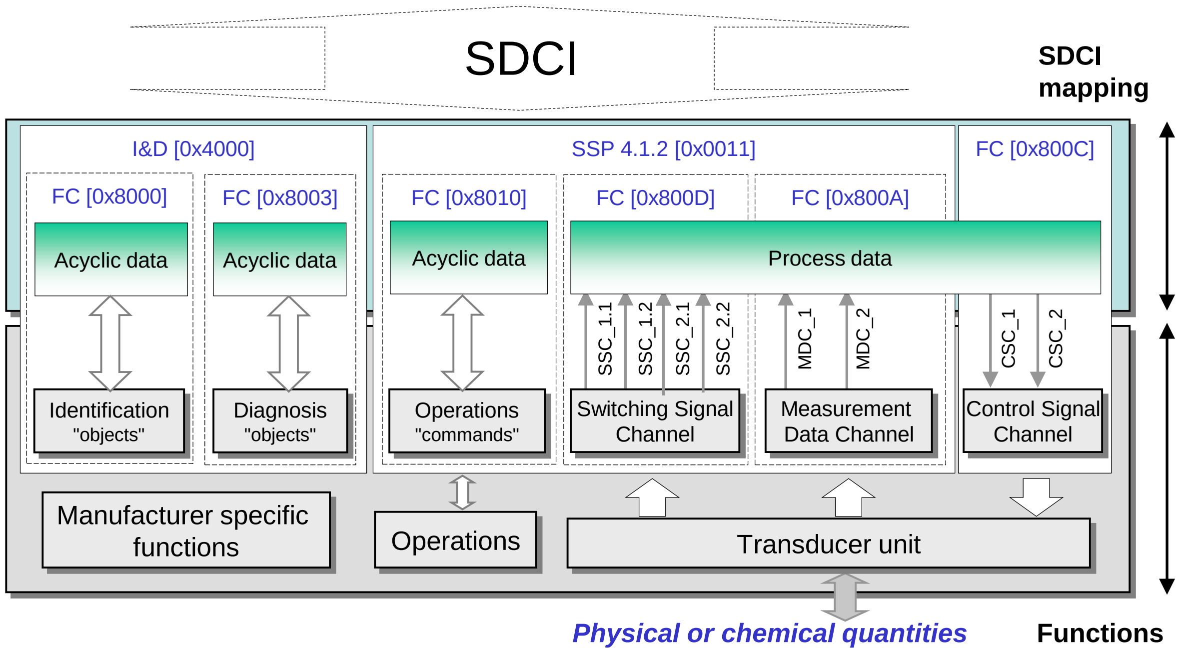


Figure 4 – Overview of typical FunctionClasses

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

The FunctionClasses DeviceIdentification, DeviceDiagnosis, ProcessDataVariable, and ExtendedIdentification are combined to the DeviceProfile [0x4000] in this document. This CommonApplicationProfile is mandatory by definition of this specification.

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

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

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

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

The parameter "ProfileCharacteristic" contains at least one ProfileIdentifier or an array of ProfileIdentifier.

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

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

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

5.6 User benefits

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

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

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

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

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

Rules and constraints for developing IO-Link profile Devices

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

6.1 Constraints for developing IO-Link Devices

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

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

6.2 How to select the appropriate Profile functions

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

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

As a next step the specific DeviceProfiles should be considered, they define specific functionalities for the specific types of Device like sensor or actuator. DeviceProfiles or CommonApplicationProfiles may define specific FunctionClasses as possible extensions.

It is not allowed to use FunctionClasses without the related DeviceProfiles or CommonApplicationProfiles for which the FunctionClasses are defined as extensions.

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

6.3 Identification of supported Profiles

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

Identification and Diagnosis (I&D)

7.1 Overview

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

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

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

Table 6 – Prefixes for IODD ID elements

Profile name	Context identifier
CommonProfile	CP

7.2 Identification and Diagnosis Profile (I&D)

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

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

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

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]

7.3 Extension of Identification and Diagnosis

The DeviceProfile Identification and Diagnosis may be accompanied by the FunctionClasses Locator, ProductURI, and TeachRecommended [CR105]. The possible extensions are defined in Table 9.

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) [CR105], see A.8	n.a.

7.4 Proxy Function Block for Identification and Diagnosis

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

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 (PDOutputDescriptor). 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:

- the data type (DataType) of the particular process variable. "Set of BoolT" describes combined independent Boolean values
- the length of the data type (TypeLength) in bit, for example 6 for UIntegerT6
- the bit offset (Bit offset) as the beginning of the variable in the data stream

The user program within a controller (e.g. PLC) can thus read this information.

A.4 Diagnosis [0x8003]

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

- DeviceStatus
- DetailedDeviceStatus

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

As already described in [1], the Events, the DetailedDeviceStatus and the DeviceStatus are interconnected. Whenever an Event appears, the DetailedDeviceStatus contains this Event until it disappears, see B.2.21 in [1]. The resulting DeviceStatus of each predefined Event is defined in Table D.1 in [1], the highest DeviceStatus value of all current sources determines the content of the DeviceStatus.

A.5 Extended Identification [0x8100]

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

The parameter

- FunctionTag
- LocationTag

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

Additionally the standardized parameter SerialNumber and HardwareRevision are set to mandatory.

A.6 Locator [0x8101]

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

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

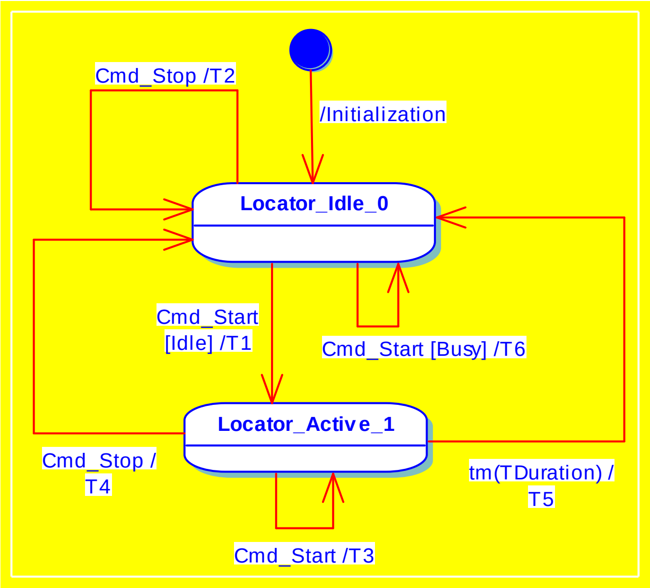


Figure A.1 – State machine for optical indication

Table A.2 shows the state transition tables for the optical indication.

Table A.2 – 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.3
timer	Variable	Timeout timer
Idle	Variable	Local user interface is in in idle state, no interaction with user
Busy	Variable	Local user interface is in interaction with user

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

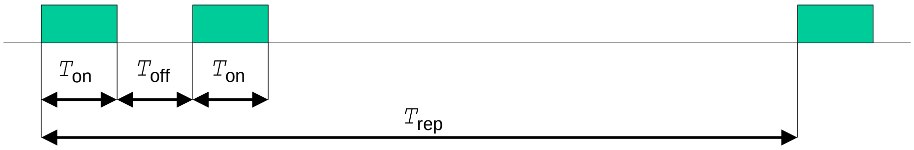


Figure A.2 – Device optical indicator timing for localization

Table A.3 defines the timing for the optical indication of Devices.

Table A.3 – 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

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

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

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

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

A.7 ProductURI

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

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

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

A.8 TeachRecommended [0x8017] [CR105]

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

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

- The application is depending on mechanical reproduction of the installation, which results in different results,
- The application needs any type of position calibration,
- Or other reasons apply, and a teach is recommended by the vendor.

The vendor is obliged to state this requirement in comprehensive manner in the user manual 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)	PDOutputDescriptor	R	variable	ArrayT of OctetStringT3	C	Conditional on availability of PDOut 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.2.12 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 Func-tionClass 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					

B.2 System Commands

This clause describes the SystemCommands which are used by the CommonProfile. The availability of the commands specified in Table B.2 is depending on the support of the corresponding FunctionClass.

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	

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

B.3 Identification parameters

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

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

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					

B.4 ProfileCharacteristics parameter

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

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

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					

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

Figure B.1 – Indication rules for ProfileIdentifiers

B.5 Process data structure descriptors

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

B.5.1 Coding of PVinD and PVoutD

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

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

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

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

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

B.5.2 PDInputDescriptor

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

The descriptors shall be sorted in ascending bit offset order, means "PVinD 1" contains the PVinD entry describing the lowest bit offset.

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

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					

B.5.3 PDOutputDescriptor

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

The descriptors shall be sorted in ascending bit offset order, means "PVoutD 1" contains the PVoutD entry describing the lowest bit offset.

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

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					

B.6 Extended Identification parameters

This clause defines the extended identification parameters which can be used for overall localization and identification of any Device.

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

Table B.8 defines the structure of the parameters.

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					

B.7 Diagnosis parameters

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

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

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					

B.8 ProductURI parameter

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

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

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

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					

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), DeviceDiagnosis (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).

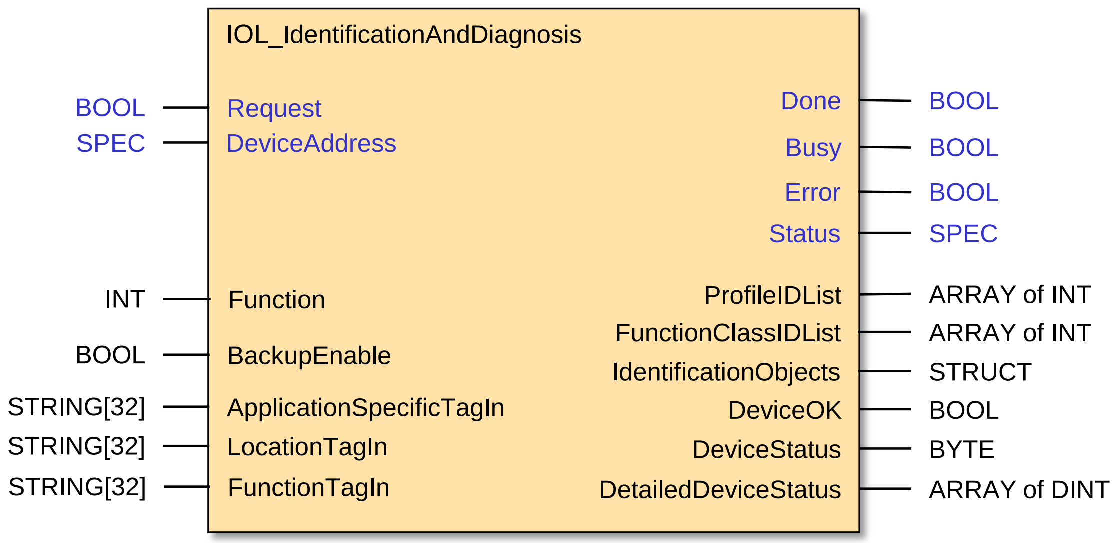


Figure C.1 – Proxy FB for Device Identification and Diagnosis

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

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		

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

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

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

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

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 definied 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.

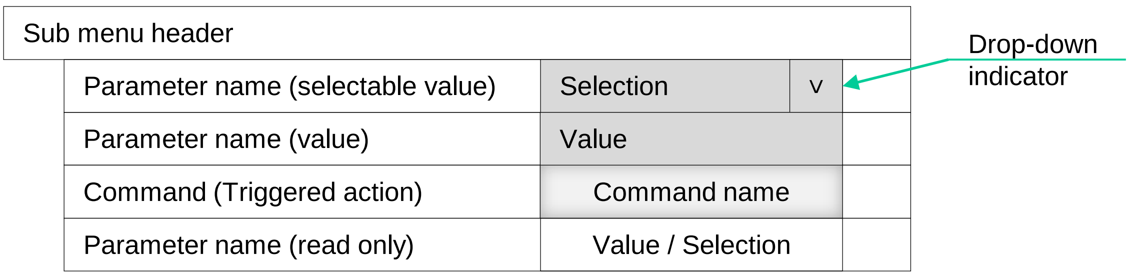


Figure D.1 – IODD object layout description

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]

Figure D.2 – Menu Profile Diagnosis

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

D.3.4 Menu structure of the Device Identification parameters

In Figure D.3 the menu structure Device Identification according to 7.2, A.2, and A.5 is specified, 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

Figure D.3 – Menu Profile Identification

D.3.5 Menu structure of the Locator functionality

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

- Identification

…

Standard Command	Locator Start
Standard Command	Locator Stop

…

623

624 **Figure D.4 – Menu Profile Locator**

Annex E
(normative)
Profile testing and conformity

E.1 General

E.1.1 Overview

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

E.1.2 Test extension for profile Devices

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

E.1.3 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 checker are extended covering the profile requirements. This predefinitions are defined in so-called IODD snippet files, which provide all necessary information to cover the ceration and the test of the profile Device's IODD.

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

This profile provides xml based files containing IODD related snippets, which can be copied and adapted to create well formed Device IODDs. These xml files contain xml elements following the rules of [2] which are extended by test related attributes. This specific extensions must 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 [CR104]
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 >

F.1.2 Hiding FunctionClasses by ProfileIDs

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

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 [CR104]
Configuration / setup	Device-Tester-Unit

TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	All already by the CommonApplicationProfileID 0x4000 incorporated Function-Classes as 0x8000, 0x8002, 0x8003, and 0x8100 shall not be listed in the Profile-Characteric.
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 >

F.1.3 Minimum required profile support

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

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 >

F.1.4 Extensions of profiles

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

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 >

F.1.5 PDInput-, PDOutputDescriptor parameter

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

Table F.5 – PDInput-, PDOutputDescriptor parameter

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	CP_TC_0005
Name	TCD_CMPR_ID_PDOUTDESCR
Purpose (short)	Test correct description of PDInput- and PDoutputDescriptor
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 PDInput Descriptor, zero length indicates no content within the process data b) Read PDOutputDescriptor, 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
Input parameter	-
Post condition	-

TEST CASE RESULTS	CHECK / REACTION
Evaluation	1. if /IODevice/ProfileBody/DeviceFunction/ProcessDataCollection/ProcessDataIn is existent then check against PD_InputDescriptor from a) 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 then check against PD_OutputDescriptor from b) 4. if /IODevice/ProfileBody/DeviceFunction/ProcessDataCollection/ProcessDataOut is not existent then check if read access at b) returns ErrorType 0x80110x8011 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) PDOut description available < ok/nok >

F.2 Test case extension for dynamical behavior

F.2.1 Device localization commands

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

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 7) without failure
Test failed (examples)	Any failure in 1) to 7)
Results	a) Locator control < ok/nok > b) Localization visible < ok/nok >

Bibliography

[1] IO-Link Community, *IO-Link Interface and System*, V1.1.3, June 2019, Order No. 10.002

[2] IO-Link Community, *IO Device Description (IODD)*, V1.1.3, January 2021, Order No. 10.012

[3] IEC/TR 62390:2005, *Common automation device profile guideline*

[4] IEC 60050 (all parts), *International Electrotechnical Vocabulary*

[5] IO-Link Community, *IO-Link Test Specification*, V1.1.3, January 2021, Order No. 10.032

[6] IO-Link Community, *IO-Link Smart Sensor Profile Ed.2*, V1.1, September 2021, Order No. 10.042

[7] ISO/IEC 19505-2:2012, Information technology – Object Management Group Unified Modeling Language (OMG UML) – Part 2: Superstructure

[8] DIN SPEC 91406:2019-12, Automatic identification of physical objects and information on physical objects in IT systems, particularly IoT systems;

____

Originator		Company	Email
Gu, Jiejie		Hefei Onsoon Intelligent Electronics Co., Ltd.	gujj@onsoon.cn
Assignee		Found in Version	Fixed in Version
---		V1.1	---
ID	State	Creation Date	Last Changed
[CR101]	Review	14.06.2022 02:35:07	14.06.2022 02:35:07
Line	Clause / Subclause Number	Clause / Subclause Title	Page
491	Annex B	B.5.2	25

Abstract:
Structure of "PDInputDescriptor" - Subindex

Description:
There are doubts about the use of Subindex in Table B.6. If the ProcessDataIn is RecordT, the bitLength="16", there are 16 Subindex, and each Subindex is BoolT, then my PDInputDescriptor is 011000 or 010100, 010101, 010102, 010103, 010104 ,010105, 010106, 010107, 010108, 010109, 01010A, 01010B, 01010C, 01010D, 01010E, 01010F or something else?

Responses:

Test:

Compatibility: no impact

Attached Files:

No downloadable files available!

Originator		Company	Email
Moser, Johannes		TEConcept	johannes.moser@teconcept.de
Assignee		Found in Version	Fixed in Version
---		V1.1	---
ID	State	Creation Date	Last Changed
[CR102]	Review	11.08.2022 11:22:35	11.08.2022 11:22:35
Line	Clause / Subclause Number	Clause / Subclause Title	Page
676	F.2.1	---	39

Abstract:
CP_TC_0006: Handling of a negative response is not clearly specified

Description:
Purpose (detailed) of the testcase says, that the test system should retry 3 times when the Device answers with negative response 0x8036 to the system command. But it is neither defined that the Device may answer with negative response 0x8036, nor it is defined that the system command must succeed latest after three retries. --- Device behaviour should be clearly defined in Specification Annex A.6 and not in the test case. Either do not allow any negative response. Or define in the specification how/how often/in which time frame a system command may be answered with negative response.

Responses:

Test:

Compatibility: no impact

Attached Files:

No downloadable files available!

Originator		Company	Email
Engelbrecht, Michael		EGE-Elektronik Spezial-Sensoren GmbH	michael.engelbrecht@ege-elektronik.com
Assignee		Found in Version	Fixed in Version
---		V1.1	---
ID	State	Creation Date	Last Changed
[CR104]	Review	10.11.2022 07:00:53	10.11.2022 07:00:53
Line	Clause / Subclause Number	Clause / Subclause Title	Page
644	F.1	---	34

Abstract:
Missing reference

Description:
In table F.1 and F.2 there is no reference source for the specification (clause).

Responses:

Test:

Compatibility: no impact

Attached Files:

No downloadable files available!

Originator		Company	Email
Hackenstraß, Kai		ifm prover GmbH	kai.hackenstrass@ifm.com
Assignee		Found in Version	Fixed in Version
---		V1.1	---
ID	State	Creation Date	Last Changed
[CR105]	Review	12.04.2023 08:40:19	12.04.2023 08:40:19
Line	Clause / Subclause Number	Clause / Subclause Title	Page
315	7.3	---	17

Abstract:
Standardization of device behaviour when teach required with device exchange

Description:
Resulting from Requirement WG - CR123: Some IO-Link devices require actions to restart after the device has been replaced. This contradicts the expectation that you can exchange replacement devices with IO-Link without further measures. Depending on the device and application, the measures are mandatory. Then the device should only make its process data valid when the measures have been taken. A diagnostic message should also be sent. In other cases, the device can be used without any measures, the measures are only used for more precise setting. The behavior could be made adjustable for critical applications. The suggestion is to standardize this topic. One possible specification would be the Common Profile. **** Response from Requirement WG: Calibration (CR119) and Teach required after device exchange is very similar from both an application and a technical point of view and should be worked out together. Therefore CR119 was closed. The requirement was decided as important improvement of the IO-Link sytem common behaviour. The Smart Sensor Profile Sub-WG is already working on a solution. But it is not required for smart sensors only. Threfore it should be implemented in the Common Profile. The CR is assigned to the Core Team. Decision Requirement Team: 2023-04-06 assigned to the Core Team for implementation

Responses:

Test:

Compatibility: no impact

Attached Files:

No downloadable files available!

712

IO-Link Community
Haid-und-Neu-Str. 7
76131 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/