IO-Link Safety System Extensions

Specification

Version 1.0
April 2017

Order No: 10.092

File name: **IO-Link_Safety_System-Extensions_10092_V10_Apr17.doc**

This specification has been prepared by the IO-Link Safety technology subgroup**.** It achieved a concept approval by TÜV-SÜD.

Any comments, proposals, requests on this document are appreciated through the IO-Link CR database www.io-link-projects.com. Please provide name and email address.
Login: *IOL-Safety10*
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:
may: indicates flexibility of choice with no implied preference.
should: indicates flexibility of choice with a strongly preferred implementation.
shall: indicates a mandatory requirement. Designers **shall** implement such mandatory requirements to ensure interoperability and to claim conformity with this specification.

Publisher:
IO-Link Community
Haid-und-Neu-Str. 7
76131 Karlsruhe
Germany
Phone: +49 721 / 96 58 590
Fax: +49 721 / 96 58 589
E-mail: info@io-link.com
Web site: www.io-link.com

CONTENTS

0 Introduction

0.1 General

The base technology of IO-Link™[1] is subject matter of the international standard IEC 61131-9 (see [2]). IEC 61131-9 is part of a series of standards on programmable controllers and the associated peripherals and should be read in conjunction with the other parts of the series.

It specifies a single-drop digital communication interface technology for small sensors and actuators – named SDCI, which extends the traditional switching input and output interfaces as defined in IEC 61131-2 towards a point-to-point communication link using coded switching. This technology enables the cyclic exchange of digital input and output process data between a Master and its associated Devices (sensors, actuators, I/O terminals, etc.). The Master can be part of a fieldbus communication system or any stand-alone processing unit. The technology enables also the acyclic transfer of parameters to Devices and the propagation of diagnosis information from the Devices to the upper-level automation system (controller, host) via the Master.

Physical topology is point-to-point from each Device to the Master using 3 wires over distances up to 20 m. The SDCI physical interface is backward compatible with the usual 24 V I/O signalling specified in IEC 61131-2. Transmission rates of 4,8 kbit/s, 38,4 kbit/s and 230,4 kbit/s are supported.

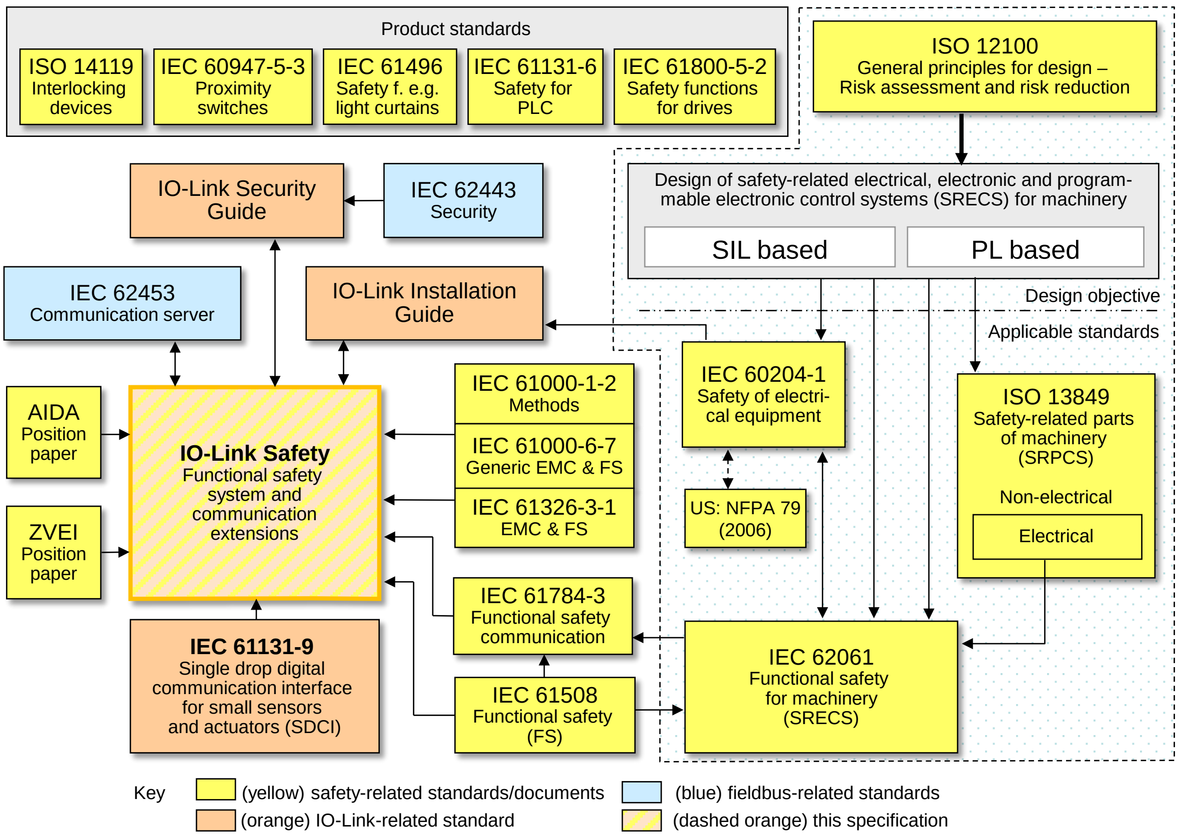


Figure 1 – Relationship of this document to standards

The main advantages of the IO-Link technology are:

- international standard for dual use of either switching signals (DI/DO) or coded switching communication respectively;

[1] IO-Link™ is a trade name of the "IO-Link Community". This information is given for the convenience of users of this specification and does not constitute an endorsement by the IO-Link Community of the trade name holder or any of its products. Compliance to this standard 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".

- traditional switching sensors and actuators now providing alternatively single drop digital communication within the same Device;
- one thin, robust, very flexible cable without shielding for power supply and signalling;
- lowest-cost digital communication down to the lowest end sensors and actuators.

As a consequence, the market demand for the extension of this technology towards functional safety has been raised.

This document provides the necessary extensions to the basic IO-Link interface and system standard for *functional safety communication* including compatibility to OSSDe based sensors and the necessary configuration management. Figure 1 shows its relationships to international fieldbus and safety standards as well as to relevant specifications.

This document does not yet provide the necessary specifications for a functional safety interface ("Combi") for actuators based on Port class B and for optional features such as functional safety signal processing as required in [11]. This part has been postponed to a later release.

The design objective for IO-Link Safety is up to SIL3 according to IEC 61508 and/or up to PLe according to ISO 13849.

Parameterization within the domain of safety for machinery requires a "Dedicated Tool" per FS-Device or FS-Device family. The Device Tool Interface (DTI) technology has been chosen for the links between FS-Master Tool, FS-Device, and its "Dedicated Tool" (Device Tool).

The structure of this document is described in 4.9.

Conformity with this document cannot be claimed unless the requirements of Annex I are met.

Terms of general use are defined in IEC 61131-1 or in the IEC 60050 series. More specific terms are defined in each part.

0.2 Patent declaration

The IO-Link Community draws attention to the fact that compliance with this document may involve the use of patents concerning the functional safety point-to-point serial communication interface for small sensors and actuators.

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.

The IO-Link Community maintains on-line data bases of patents relevant to their standards. Users are encouraged to consult the databases for the most up to date information concerning patents.

IO-Link Safety – Functional safety communication and system extensions – based on IEC 61131-9 (SDCI)

1 Scope

For the design of functional safety communication on IO-Link there exist mainly three options:

- existing functional safety communication profiles (FSCP) specified within the IEC 61784-3-x series, *tunnelling* across IO-Link;
- a *new universal FSCP* suitable for all fieldbuses standardized in IEC 61158, also tunnelling across IO-Link;
- a *new lean dedicated functional safety communication interface* (IO-Link Safety) solely between Device and Master requiring a safety gateway for the connection to FSCPs.

This document specifies only the new lean functional safety communication interface including connectivity of OSSDe type safety sensors (FS-Devices).

Figure 2 shows four typical fieldbus/FSCP configurations A to D with remote I/Os (RIO) and attached FS-DIs as well as gateways to IO-Link Safety ("IOL-S"). The gateways contain FSCP-specific FS-Masters. FS-Devices with OSSDe can be connected to FS-DIs or FS-Masters. All IO-Link safety sensors (FS-Device) can communicate with any IO-Link Safety Master (FS-Master) using the IO-Link Safety protocol regardless of the upper level FSCP-system. The same is true for IO-Link safety actuators (FS-Devices) such as drives with integrated safety. This means the largest component commonality① for sensors and actuators similar to the DI and DO interfaces standardized within IEC 61131-2.

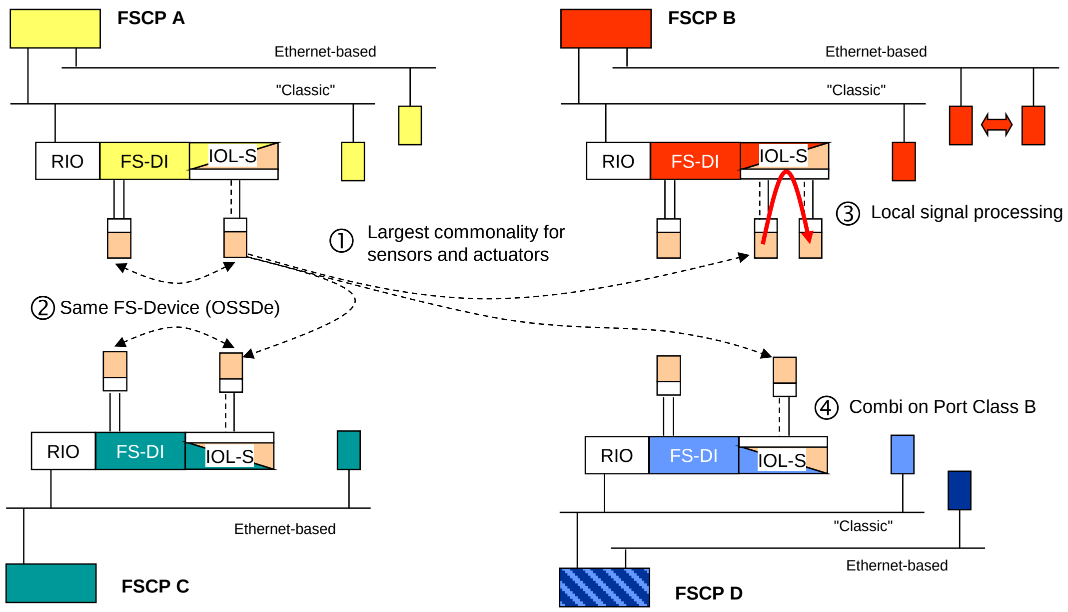


Figure 2 – IO-Link Safety on single platform

Safety sensors with OSSDe interfaces – equipped with IO-Link communication – can be parameterized via auxiliary tools such as "USB-Masters", then connected to an FS-DI and operated in OSSDe mode. They also can be operated in OSSDe mode on an FS-Master supporting OSSDe. In case these safety sensors are equipped with IO-Link Safety communication in addition, they can be operated in both modes②, either OSSDe or IO-Link Safety. This corresponds to the IO-Link SIO paradigm.

The concept of IO-Link Safety allows for local safety signal processing (safety functions) if the FS-Master provides a local safety controller③. This document specifies the interfaces if required.

The IO-Link specifications [1] and [2] define a Master Port class B with an extra 24 V power supply for actuators using a 5 pin M12 connector. The list of requirements in [11] suggests an extension – called "Combi-Port" –, where the power-down of the extra power supply can be controlled by the FS-Master itself④. This document does not yet specify this kind of Master Port class B. It is postponed until a later version.

NOTE The illustrations ① to ④ be valid for all FSCPs.

This document does not cover communication interfaces or systems incorporating multi-point or multi-drop linkages, or integration of IO-Link Safety into upper level systems such as fieldbuses.

2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 60947-5-3, *Low-voltage switchgear and controlgear – Part 5-3: Control circuit devices and switching elements – Requirements for proximity devices with defined behaviour under fault conditions (PDDB)*

IEC 61000-1-2, *Electromagnetic compatibility (EMC) - Part 1-2: General - Methodology for the achievement of functional safety of electrical and electronic systems including equipment with regard to electromagnetic phenomena*

IEC 61000-6-7, *Electromagnetic compatibility (EMC) - Part 6-7: Generic standards - Immunity requirements for equipment intended to perform functions in a safety-related system (functional safety) in industrial locations*

IEC 61131-2, *Programmable controllers – Part 2: Equipment requirements and tests*

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

IEC 61496-1, *Safety of machinery – Electro-sensitive protective equipment – Part 1: General requirements and tests*

IEC 61508-2:2010, *Functional safety of electrical/electronic/programmable electronic safety-related systems - Part 2: Requirements for electrical/electronic/programmable electronic safety-related systems*

IEC 61508-3:2010, *Functional safety of electrical/electronic/programmable electronic safety-related systems - Part 3: Software requirements*

IEC 61784-3:2016, *Industrial communication networks - Profiles - Part 3: Functional safety fieldbuses - General rules and profile definitions*

IEC 62061, *Safety of machinery – Functional safety of safety-related electrical, electronic and programmable electronic control systems*

IEC 62443 all*, Security for industrial automation and control systems*

IEC 62453, *Field device tool (FDT) interface specification*

ISO 12100:2010*, Safety of machinery – General principles for design – Risk assessment and risk reduction*

ISO 13849-1:2015, *Safety of machinery – Safety-related parts of control systems – Part 1: General principles for design*

ISO 14119:2013, *Safety of machinery – Interlocking devices associated with guards – Principles for design and selection*

3 Terms, definitions, symbols, abbreviated terms and conventions

3.1 Common terms and definitions

For the purposes of this document, the terms and definitions given in IEC 61131-1 and IEC 61131-2, as well as the following apply.

3.1.1
address
part of the M-sequence control to reference data within data categories of a communication channel

3.1.2
application layer
AL
<SDCI>[2] part of the protocol responsible for the transmission of Process Data objects and On-request Data objects

3.1.3
block parameter
consistent parameter access via multiple Indices or Subindices

3.1.4
checksum
<SDCI> complementary part of the overall data integrity measures in the data link layer in addition to the UART parity bit

3.1.5
CHKPDU
integrity protection data within an ISDU communication channel generated through XOR processing the octets of a request or response

3.1.6
coded switching
SDCI communication, based on the standard binary signal levels of IEC 61131-2

3.1.7
COM1
SDCI communication mode with transmission rate of 4,8 kbit/s

3.1.8
COM2
SDCI communication mode with transmission rate of 38,4 kbit/s

3.1.9
COM3
SDCI communication mode with transmission rate of 230,4 kbit/s

3.1.10
COMx
one out of three possible SDCI communication modes COM1, COM2, or COM3

3.1.11
communication channel
logical connection between Master and Device

Note 1 to entry: Four communication channels are defined: process channel, page and ISDU channel (for parameters), and diagnosis channel.

3.1.12
communication error
unexpected disturbance of the SDCI transmission protocol

---

[2] Angle brackets indicate validity of the definition for the SDCI (IO-Link) technology

3.1.13
cycle time
time to transmit an M-sequence between a Master and its Device including the following idle time

3.1.14
Device
single passive peer to a Master such as a sensor or actuator

Note 1 to entry: Uppercase "Device" is used for SDCI equipment, while lowercase "device" is used in a generic manner.

3.1.15
Direct Parameters
directly (page) addressed parameters transferred acyclically via the page communication channel without acknowledgement

3.1.16
dynamic parameter
part of a Device's parameter set defined by on-board user interfaces such as teach-in buttons or control panels in addition to the static parameters

3.1.17
Event
instance of a change of conditions in a Device

Note 1 to entry: Uppercase "Event" is used for SDCI Events, while lowercase "event" is used in a generic manner.

Note 2 to entry: An Event is indicated via the Event flag within the Device's status cyclic information, then acyclic transfer of Event data (typically diagnosis information) is conveyed through the diagnosis communication channel.

3.1.18
fallback
transition of a port from coded switching to switching signal mode

3.1.19
inspection level
degree of verification for the Device identity

3.1.20
interleave
segmented cyclic data exchange for Process Data with more than 2 octets through subsequent cycles

3.1.21
ISDU
indexed service data unit used for acyclic acknowledged transmission of parameters that can be segmented in a number of M-sequences

3.1.22
legacy (Device or Master)
Device or Master designed in accordance with [8]

3.1.23
M-sequence
sequence of two messages comprising a Master message and its subsequent Device message

3.1.24
M-sequence control
first octet in a Master message indicating the read/write operation, the type of the communication channel, and the address, for example offset or flow control

3.1.25
M-sequence error
unexpected or wrong message content, or no response

3.1.26
M-sequence type
one particular M-sequence format out of a set of specified M-sequence formats

3.1.27
Master
active peer connected through ports to one up to n Devices and which provides an interface to the gateway to the upper level communication systems or PLCs

Note 1 to entry: Uppercase "Master" is used for SDCI equipment, while lowercase "master" is used in a generic manner.

3.1.28
message
<SDCI> sequence of UART frames transferred either from a Master to its Device or vice versa following the rules of the SDCI protocol

3.1.29
On-request Data
acyclically transmitted data upon request of the Master application consisting of parameters or Event data

3.1.30
physical layer
first layer of the ISO-OSI reference model, which provides the mechanical, electrical, functional and procedural means to activate, maintain, and de-activate physical connections for bit transmission between data-link entities

Note 1 to entry: Physical layer also provides means for wake-up and fallback procedures.

[SOURCE: ISO/IEC 7498-1, 7.7.2, modified – text extracted from subclause, note added]

3.1.31
port
communication medium interface of the Master to one Device

3.1.32
port operating mode
state of a Master's port that can be either INACTIVE, DO, DI, FIXEDMODE, or SCANMODE

3.1.33
Process Data
input or output values from or to a discrete or continuous automation process cyclically transferred with high priority and in a configured schedule automatically after start-up of a Master

3.1.34
Process Data cycle
complete transfer of all Process Data from or to an individual Device that may comprise several cycles in case of segmentation (interleave)

3.1.35
single parameter
independent parameter access via one single Index or Subindex

3.1.36
SIO
port operation mode in accordance with digital input and output defined in IEC 61131-2 that is established after power-up or fallback or unsuccessful communication attempts

3.1.37
static parameter
part of a Device's parameter set to be saved in a Master for the case of replacement without engineering tools

3.1.38
switching signal
binary signal from or to a Device when in SIO mode (as opposed to the "coded switching" SDCI communication)

3.1.39
system management
SM
<SDCI> means to control and coordinate the internal communication layers and the exceptions within the Master and its ports, and within each Device

3.1.40
UART frame
<SDCI> bit sequence starting with a start bit, followed by eight bits carrying a data octet, followed by an even parity bit and ending with one stop bit

3.1.41
wake-up
procedure for causing a Device to change its mode from SIO to SDCI

3.1.42
wake-up request
WURQ
physical layer service used by the Master to initiate wake-up of a Device, and put it in a receive ready state

3.2 IO-Link Safety: Additional terms and definitions

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

3.2.1
error
discrepancy between a computed, observed or measured value or condition and the true, specified or theoretically correct value or condition

Note 1 to entry: Errors may be due to design mistakes within hardware/software and/or corrupted information due to electromagnetic interference and/or other effects.

Note 2 to entry: Errors do not necessarily result in a *failure* or a *fault*.

SOURCE: [IEC 61508-4:2010], [IEC 61158]

3.2.2
failure
termination of the ability of a functional unit to perform a required function or operation of a functional unit in any way other than as required

Note 1 to entry: The definition in IEC 61508-4 is the same, with additional notes.

Note 2 to entry: Failure may be due to an error (for example, problem with hardware/software design or message disruption)

SOURCE: [IEC 61508-4:2010, modified], [ISO/IEC 2382-14.01.11, modified]

3.2.3
fault
abnormal condition that may cause a reduction in, or loss of, the capability of a functional unit to perform a required function

Note 1 to entry: IEV 191-05-01 defines “fault” as a state characterized by the inability to perform a required function, excluding the inability during preventive maintenance or other planned actions, or due to lack of external resources.

SOURCE: [IEC 61508-4:2010, modified], [ISO/IEC 2382-14.01.10, modified]

3.2.4
FS-Device
single passive peer such as a functional safety sensor or actuator to a Master with functional safety capabilities

3.2.5
FS-Master
active peer with functional safety capabilities connected through ports to one up to n Devices or FS-Devices and which provides an interface to the gateway to the upper level communication systems (NSR or SR) or controllers with functional safety capabilities

3.2.6
FSP parameter
parameter set for the administration and operation of the IO-Link Safety protocol

3.2.7
FST parameter
parameter set for the safety-related technology of an FS-Device, for example light curtain

3.2.8
Safety Protocol Data Unit
SPDU
protocol data unit transferred through the safety communication channel

[SOURCE: IEC 61784-3:2015 modified]

3.3 Symbols and abbreviated terms

AIDA	Automatisierungsinitiative Deutscher Automobilhersteller	
AL	application layer	
BEP	bit error probability	
C/Q	connection for communication (C) or switching (Q) signal (SIO)	
CRC	cyclic redundancy check	
DDO	Device data object	
DI	digital input	
DL	data link layer	
DO	digital output	
DTI	Device Tool Interface	
FDI	Field Device Integration	[IEC 62769]
FDT	Field Device Tool	[IEC 62453]
FS	functional safety	
FSCP	functional safety communication profile (for example IEC 61784-3-x series)	
FS-AI	functional safety analog input	
FS-DI	functional safety digital input	
I/O	input / output	
IODD	IO Device Description	
IOPD	IO-Link Parameterization & Diagnostic tool	
IOL-S	IO-Link Safety	
L-	power supply (-)	
L+	power supply (+)	
N24	24 V extra power supply (-); Port class B	
NSR	non safety-related	
OD	On-request Data	

OK	"OK", values or state correct	
OSSD	output signal switching device (self-testing electronic device with built-in OSSD)	[IEC 61496-1]
OSSDe	output signal switching device (self-testing electronic device with built-in OSSD)	[This document]
OSSD1/2e	pin assigment of both OSSDe signals according to [14]	
OSSDm	output signal switching device (relay and solid state outputs)	[IEC 60947-5-5]
P24	24 V extra power supply (+); Port class B	
PD	Process Data	
PDin	functional safety input process data (from an FS-Master's view)	
PDout	functional safety output process data (from an FS-Master's view)	
PDCT	port and Device configuration tool	
PFH	(average) probability of a dangerous failure per hour	
PID	program interface description	
PL	physical layer	
PLC	programmable logic controller	
PS	power supply (measured in V)	
RIO	remote I/O	
SCL	safety communication layer	
SDCI	single-drop digital communication interface	[IEC 61131-9]
SIO	standard input output (digital switching mode)	[IEC 61131-2]
SM	system management	
SPDU	safety protocol data unit	
SR	safety-related	
SSI	synchronous serial interface (usually for encoders)	
TAF	temporary acknowledgment file	
TBF	temporary backchannel file	
TPF	temporary parameter file	
UART	universal asynchronous receiver transmitter	
UML 2	unified modeling language, edition 2	[ISO/IEC 19505-2]
WURQ	wake-up request pulse	
XML	extensible markup language	

3.4 Conventions

3.4.1 Behavioral descriptions

For the behavioral descriptions, the notations of UML 2 are used, mainly for state and sequence diagrams (see [3], [6], or [7]).

Events to trigger a transition usually can be a signal, service call, or timeout. Logic conditions (true/false) shall be the result of a [guard]. To alleviate the readability and the maintenance of the state machines, the diagrams do not provide the actions associated with a transition. These actions are listed within a separate state-transition table according to IEC 62390 [8].

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

3.4.2 Memory and transmission octet order

Figure 3 demonstrates the order that shall be used when transferring WORD based data types from memory to transmission and vice versa.

NOTE Existing microcontrollers can differ in the way WORD based data types are stored in memory: "big endian" and "little endian". If designs are not taking into account this fact, octets can be erroneously permuted for transmission.

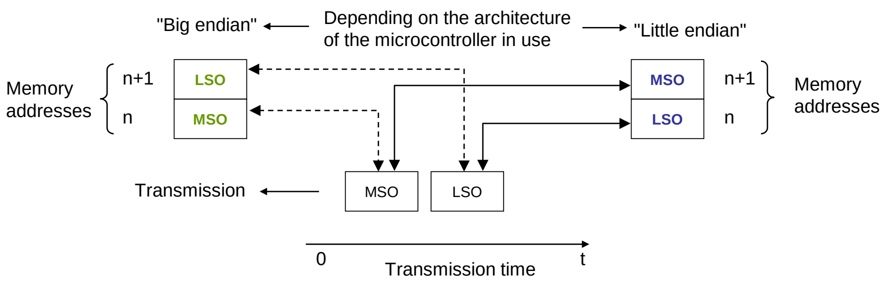


Figure 3 – Memory and transmission octet order

4 Overview of IO-Link Safety

4.1 Purpose of the technology and feature levels

4.1.1 Base IO-Link Safety technology

This document specifies a new lean functional safety communication protocol on top of the existing IO-Link transmission system specified in [1] or within the international standard IEC 61131-9 [2]. Figure 4 illustrates how the corresponding IO-Link Safety communication layers are located within the architectural models of Master and Device such that they become FS-Master and FS-Device. Most of the original IO-Link design remains unchanged for this specification.

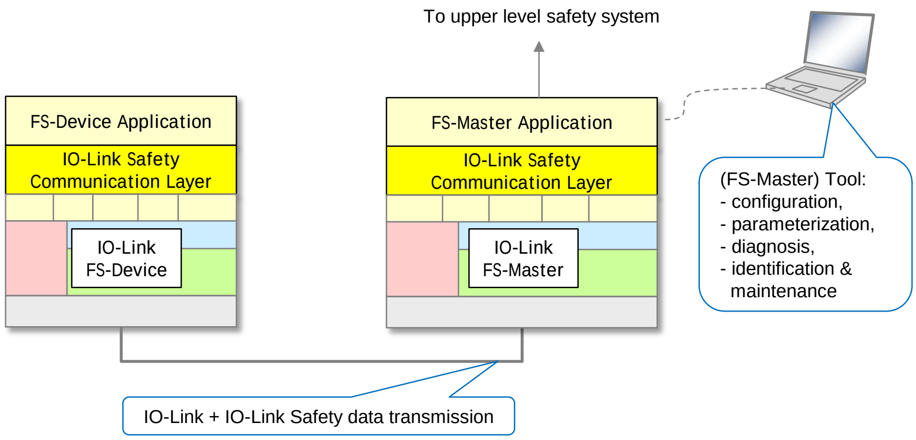


Figure 4 – IO-Link Safety communication layer model

The IO-Link Safety communication layer accommodates the functional safe transmission protocol. This protocol generates a safety PDU consisting of the FS-I/O data, protocol control or status data, and a CRC signature. The safety PDU together with optionally non-safety-related data is transmitted as IO-Link Process Data between an FS-Master and one single FS-Device (point-to-point).

IO-Link Safety increases the number of Port modes and thus requires changes to the Physical Layer and System Management.

Changes are required for the Master-(Software)-Tool to provide the necessary safety-related configuration and parameterization of the protocol (FSP-Parameter) as well as of the particular FS-Device technology (FST-Parameter).

IO-Link Safety comprises not only the digital communication; it also supports OSSDe (class A) in this version, similar to the SIO mode.

IO-Link Safety does not support

- wireless connections between FS-Master and FS-Device (see Annex H.2);
- cascaded FS-Master/FS-Device systems.

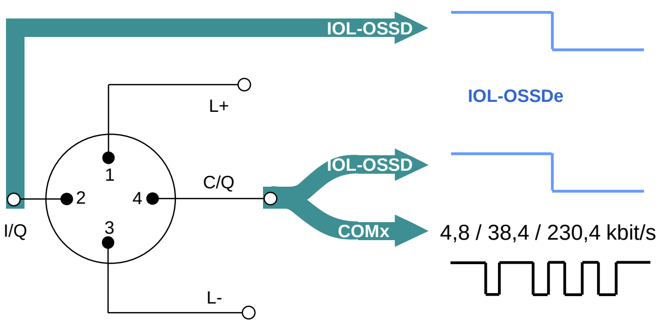
4.1.2 From "analog" and "switching" to communication

In "Safety-for-Machinery", usually the switch states (on/off) of relays or sensors are transmitted similar to standard IO-Link (SIO) as a 24 V or 0 V signal to FS-DI-Modules within remote I/Os. In contrast to standard IO-Link, due to safety requirements, these signals are redundant, either equivalent (OSSDe = 11→00) or antivalent (OSSDm = 01→10) switching.

NOTE OSSDe stands for IEC 61496-1 and OSSDm for IEC 60947-5-5 concepts.

The electrical characteristics for the OSSDe interface are following IEC 61131-2, type 1 (see Figure 5).

"Single-platform":



Pin	Signal	Definition	Standard
1	L+	24 V	IEC 61131-2
2	I/Q	Not connected, DI / **IOL-OSSD2e**, or DO	IEC 61131-2
3	L-	0 V	IEC 61131-2
4	Q	DI / **IOL-OSSD1e**, or DO	IEC 61131-2
	C	"Coded switching" COM1…3 + new IOL-Safety profile	IEC 61131-9 + "Upgrade" + "Profile"

Key: IOL-OSSDe = Equivalent switching redundant signals

Figure 5 – Port interface extensions for IO-Link Safety

Measurement of physical quantities such as temperature, pressure, position, or strain (FS-AI-Modules) has several interface solutions such as 4 to 20 mA, 0 to 10 V, or SSI, but no common signal transmission technology (see Figure 6, left).

Actuators such as motors can be de-energized via FS-DO-Modules and connected relays as shown in Figure 6 (left).

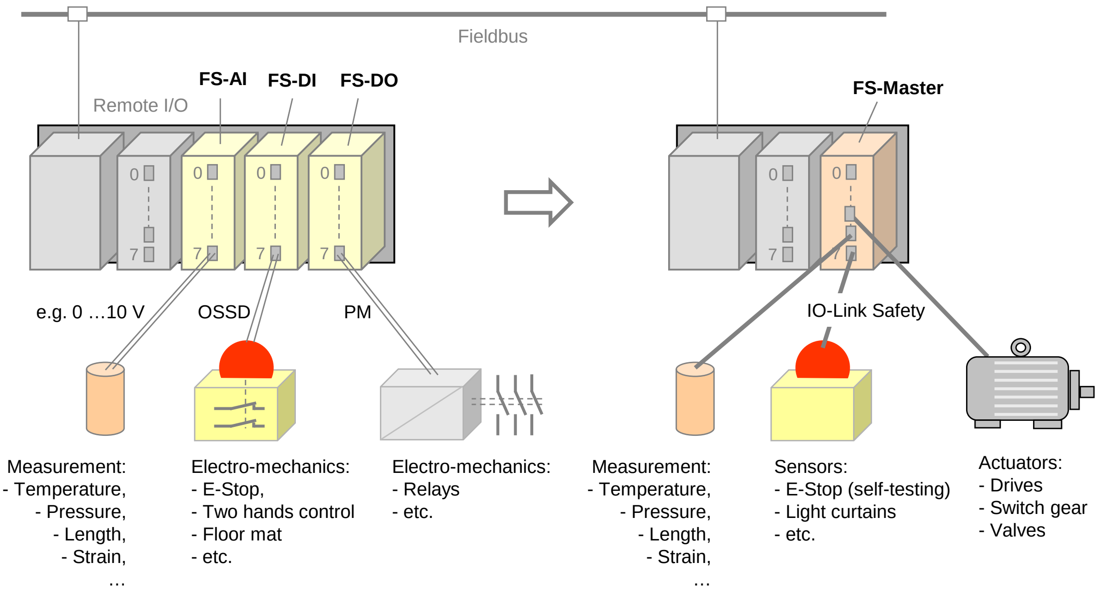


Figure 6 – Migration to IO-Link Safety

Without additional interfaces, it was not possible in all cases to configure or parameterize the safety devices or to receive diagnosis information.

IO-Link Safety can now provide a functional safe and reliable solution for process data exchange (signal states and measurement values) via single drop digital communication (SDCI), as well as parameterization and diagnosis (see Figure 6, right).

4.1.3 Minimized paradigm shift from FS-DI to FS-Master

Similar to nowadays safety devices for FS-DI modules (see Figure 7) and in contrast to FSCP-based safety devices, it is not necessary to

setup an *authenticity code switch or adequate software solution*;

assign a *watchdog time*;

use any software tool in case of *FS-Device replacement*.

Authenticity is guaranteed through checking of the correct FS-Device to the assigned FS-Master Port during commissioning similar to FS-DI modules. However, IO-Link Safety provides means to discover any incorrect plugging.

IO-Link Safety uses a watchdog timer for the transmission of safety data in time (Timeliness). The system is able to calculate the required watchdog time automatically due to the point-to-point nature of the transmission.

FS-Device replacement without tools can be achieved using the original IO-Link Data Storage mechanism.

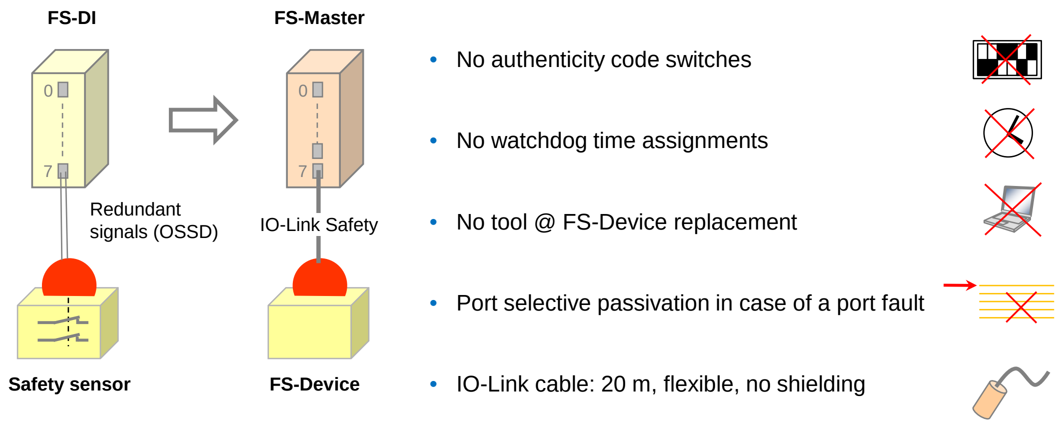


Figure 7 – Minimized paradigm shift from FS-DI to FS-Master

The FS-Master supports *port selective passivation* in case of a port fault and *signal granular passivation* in case of a channel fault within for example a remote I/O terminal ("Hub") connected to an FS-Master Port.

Cables are the same as with IO-Link, i.e. unshielded with a maximum of 20 m. However, due to the higher permitted power supply current of 1000 mA per Port, the overall loop resistance RL_{eff} can only be 1,2 Ohm (see Table 9).

NOTE Compliance to AIDA rules requires cable color to be any except yellow. However, the connector color shall be yellow (RAL 1004).

4.1.4 Following the IO-Link paradigm (SIO vs. OSSDe)

Standard IO-Link supports a port type A (4 pin) without extra power supply and a port type B (5 pin) with extra 24 V power supply (see [1] or [2]). IO-Link Safety takes care of several specification levels "a" to "d" (see Figure 8). The number of pins refers to the possible FS-Master pins.

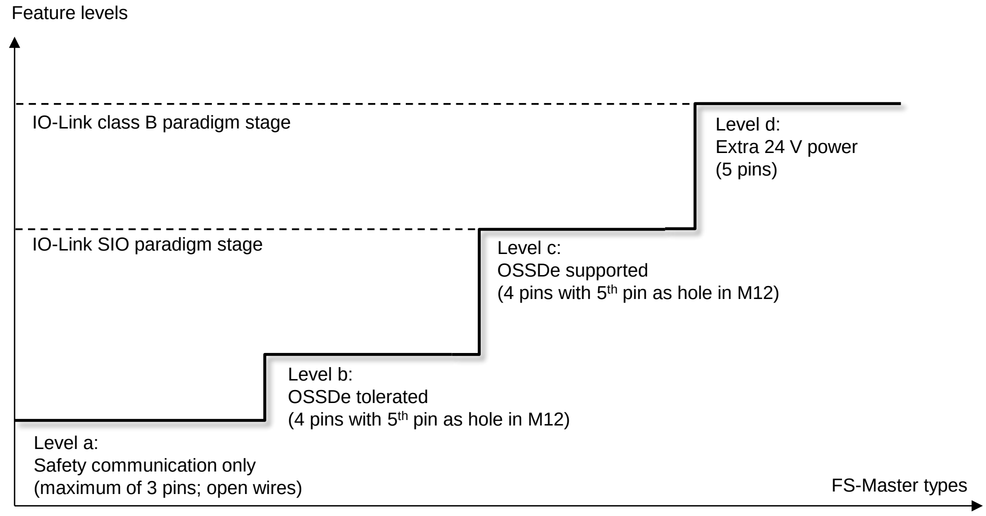


Figure 8 – FS-Master types and feature levels

The original pin layouts of IO-Link for port class A are shown in Figure 9 together with the extensions for level "a" through "c". Table 1 shows the details of these levels.

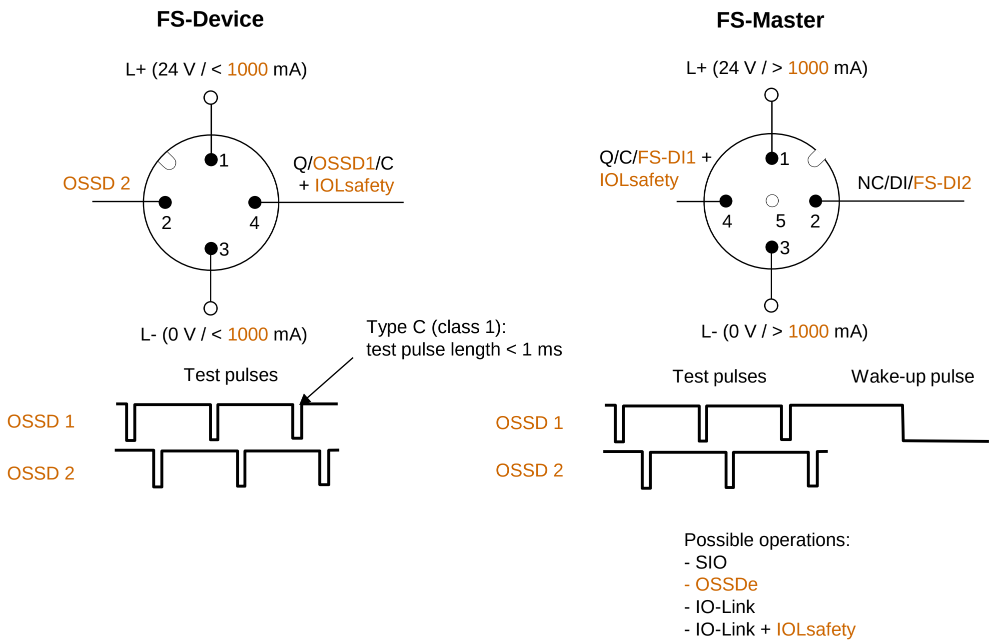


Figure 9 – Original pin layout of IO-Link (port class A)

Level "a" provides communication only (Pin 1, 3, and 4). That means support for sensor-type FS-Devices and actuator-type FS-Devices.

Due to the redundant nature of most of the safety device interfaces, IO-Link Safety considers pin 2 for the redundant signal path (e.g. OSSD2e) besides pin 4 for the primary signal path (e.g. OSSD1e)[3]. Thus, level "b" allows FS-Devices to provide OSSDe outputs besides the IO-Link Safety communication capability. They can be parameterized with the help of a "USB-Master" and be connected to any FS-DI module in switching mode. When connected to an FS-Master, safety and standard non-safety communication is possible.

Level "c" corresponds to the SIO level of standard IO-Link Master. In this case, the FS-Master supports an OSSDe mode besides communication (Pin 1, 3, 4 and 2).

3 FS-Devices are based on electronics and not on relays. Thus, the electronic version OSSDe is considered.

Table 1 shows the pin layout and possible operational modes for the feature levels "a" to "c" of the port class A FS-Device and FS-Master.

Table 1 – Operational modes of feature level "a" to "c" (port class A)

Feature level	FS-Device		FS-Master	
	Pin 2	Pin 4	Pin 2	Pin 4
"a"	- NC, DI, DO	- DI, DO - IO-Link - IO-Link + IOL-S	- NC, DI, DO	- DI, DO - IO-Link - IO-Link + IOL-S
"b"	- NC, DI, DO - OSSD2e	- DI, DO - OSSD1e - IO-Link - IO-Link + IOL-S	- NC, DI, DO	- DI, DO - IO-Link - IO-Link + IOL-S
"c"	- NC, DI, DO - OSSD2e	- DI, DO - OSSD1e - IO-Link - IO-Link + IOL-S	- NC, DI, DO - FS-DI	- DI, DO - FS-DI - IO-Link - IO-Link + IOL-S
Key IOL-S = IO-Link Safety				

Figure 10 shows the optimized OSSDe commissioning with FS-Masters:

- No filter adjustments due to fixed maximum test pulse length of 1 ms according to type C and class 1 in [12], and
- No discrepancy time adjustments due to fixed maximum discrepancy.

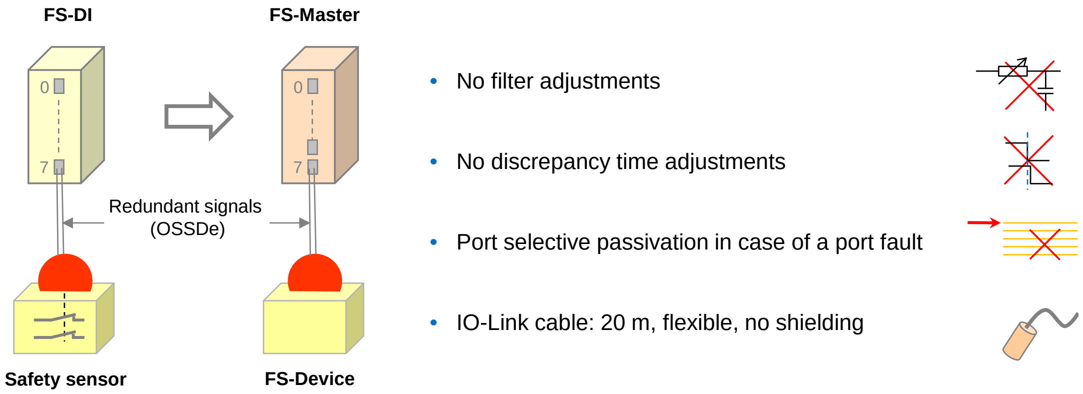


Figure 10 – Optimized OSSDe commissioning with FS-Master

4.1.5 Port class B (Classic and Combi)

The original strategy for a port class B provides for an extra 24 V power supply for actuators supplementing the main 24 V power supply of IO-Link (see [1]). This extra power supply was already considered in external functional safety concepts. According to these concepts, it is possible to switch off the extra power supply via FSCP controls and thus de-energize the actuator [11]. Annex Jspecifies details for this "classic" approach.

The new strategy suggests incorporating the P24- and N24-safety switches into the FS-Master port and controlling them via signals within the FSCP message or by local safety controls. The required technology corresponds to level "d" in Figure 8.

It is intended to specify the additional port electronics and control features in a later version of this document.

Figure 11 shows the pin layout, signal, and power supply assignment as well as the internal switches for L+, P24, and N24.

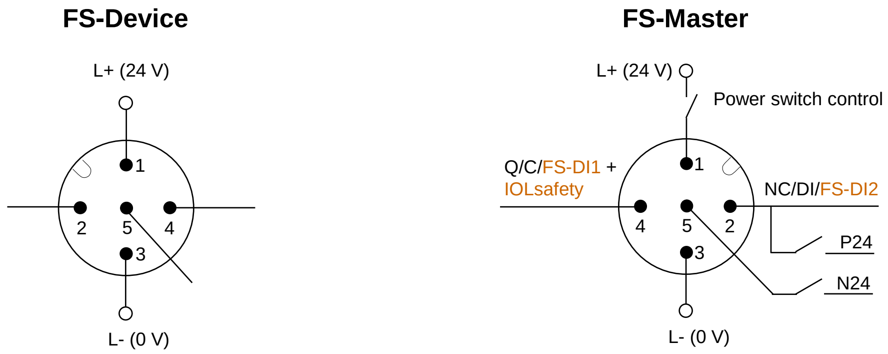


Figure 11 – Level "d" of an FS-Master (Combi – class B)

4.1.6 "USB-Master" with safety parameterization

It is possible to use upgraded "USB-Masters" for off-site configuration, parameterization and test as shown in Figure 12. Due to functional safety requirements, it will be necessary to extend the Master-Tool software for the functional safe configuration and parameterization of the FS-Device technology (FST-Parameters).

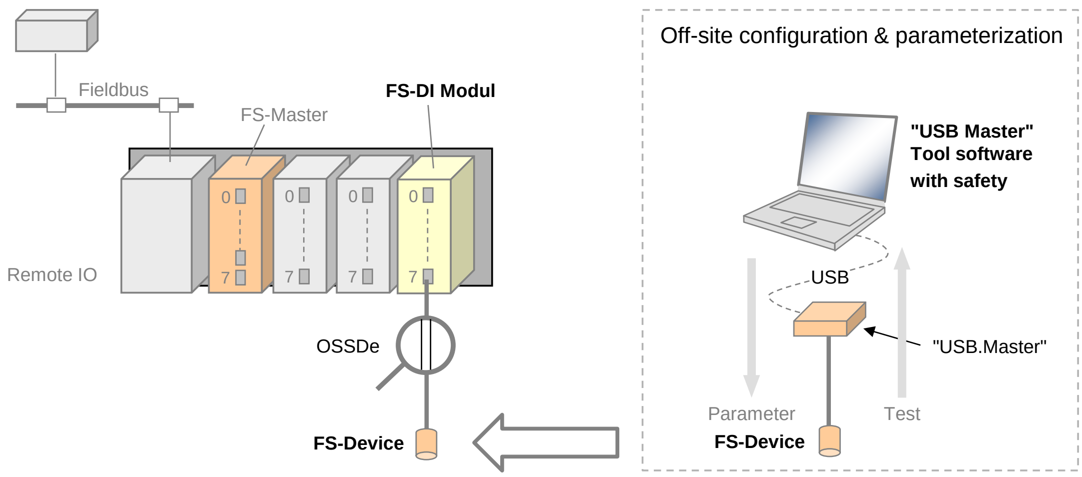


Figure 12 – Off-site configuration and parameterization

Table 2 shows the device types that can be supported by such a "USB-Master".

4.1.7 Interoperability matrix of safety devices

Table 2 provides an overview of typical safety sensors and actuators and their interoperability with FS-Masters of different feature levels, a "USB-Master" upgraded to safety parameterization, and conventional FS-DI modules connected to FSCPs.

Table 2 – Interoperability matrix of safety devices

Device type	FS-Master			"USB-Master" with safety parameterization	FS-DI module (FSCP)
	Communication "a"	OSSDe tolerated "b"	OSSDe supported "c"		
Sensor with OSSDe [a]	-	-	OSSDe	-	OSSDe
Sensor with OSSDe and IO-Link	-	-	OSSDe	IO-Link [b]	OSSDe
Sensor with OSSDe and IOL-S	IOL-S	IOL-S	OSSDe or IOL-S	IO-Link	OSSDe
Sensor with IOL-S communication only, e.g. light curtain	IOL-S	IOL-S	IOL-S	IO-Link	-
Sensor with OSSDm, e.g. E-Stop	-	-	-	-	OSSDm

Device type	FS-Master			"USB-Master" with safety parameterization	FS-DI module (FSCP)
	Communication "a"	OSSDe tolerated "b"	OSSDe supported "c"		
Actuator with IOL-S, e.g. 400 V power drive, low voltage switch gear	IOL-S	IOL-S	IOL-S	IO-Link	-
Key IOL-S = IO-Link Safety including non-safety a Pin layout according to [14]. b Pin layout may differ	USB = Universal Serial Bus, currently the most common interface amongst possible others for offsite parameterization tools due to fast communication combined with power supply				

4.2 Positioning within the automation hierarchy

Figure 13 shows the positioning of IO-Link Safety within the automation hierarchy.

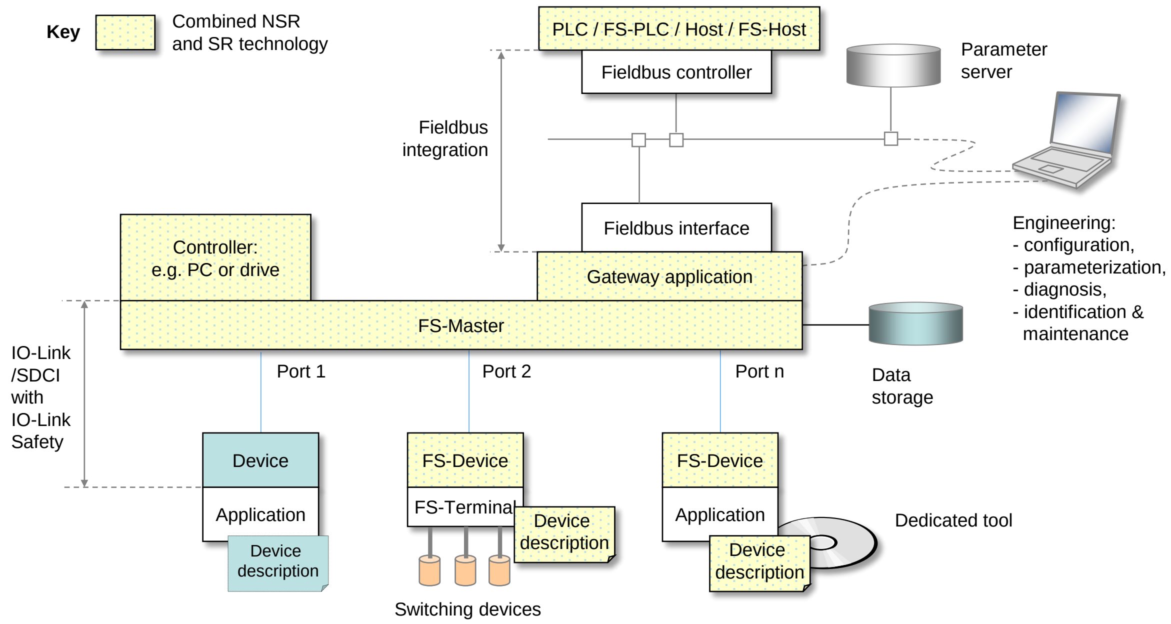


Figure 13 – IO-Link Safety within the automation hierarchy

Classic safety is relay based and thus seemed to be straightforward, easily manageable, and reliable. However, the same criteria that led to the success of fieldbuses, led to the success of functional safety communication profiles (FSCP) on top of the fieldbuses also: reduced wiring, variable parameterization, detailed diagnosis, and more flexibility. IO-Link is the perfect complement to the fieldbus communication and bridges the gap to the lowest cost sensors and actuators. It not only provides communication, but power supply on the same flexible and unshielded cable. One type of sensor can be used in the traditional switching mode or in the coded switching mode (communication). IO-Link Safety follows exactly this paradigm.

It aims for two main application areas. One is building up safety functions across the IO-Link Safety communications and the functional safety communications across fieldbuses. The other builds up safety functions "locally" between a safety controller and safety sensors/actuators using IO-Link Safety communication.

IO-Link Safety allows for building up power saving FS-Devices ("green-line"), for self-testing safety sensors in order to avoid yearly testing, for the reduction of interface types (e.g. 0 to 10 V, 4 to 20 mA, etc.), and for robust and reliable transmission of safety information.

Last but not least it is a precondition for new automation concepts such as Industry 4.0 or the Internet-of-Things (IoT).

4.3 Wiring, connectors, and power supply

Port class A types (3 to 4 wires): Cables and connectors as specified in [1] for Class A can be used for IO-Link Safety also. However, due to the higher permitted power supply current of up to 1000 mA per Port, the overall loop resistance RL_{eff} can only be 1,2 Ohm. No shielding is required.

Port class B types (5 wires): Cable, wire gauges, shielding, maximum switched currents, interference, signal levels, etc. are not specified within this document.

4.4 Relationship to IO-Link

The IO-Link communication and its SIO mode are used as the base vehicle ("black channel") for IO-Link Safety. Besides IO-Link Safety, any FS-Master Port can be configured for standard IO-Link operation also.

The independent signal inputs of the SIO mode on Pin 2 and Pin 4 are scanned by an FS-Master simultaneously to achieve an OSSDe interface. The result is propagated to the upper level safety system as one safety signal. A new Safety Layer Manager supports this feature.

Another new Port configuration mode enables the IO-Link Safety communication. Standard state machines are slightly extended to support

- detection of a Ready pulse from the FS-Device on Pin 4
- power supply (Pin 1) switching OFF/ON in case an FS-Device missed the Wake-up sequence and started its OSSDe operation
- transmission of functional safety protocol parameters (FSP) during PREOPERATE from FS-Master to the FS-Device
- activation of the IO-Link safety communication layer (SCL)
- activation of the FS Process Data Exchange within the Safety Layer Manager

4.5 Communication features and interfaces

FS Process Data from and to an FS-Device are always packed into a safety code envelop consisting of a safety PDU counter, protocol Control/Status information, and a 16/32 bit CRC signature. The minimum safety PDU size is 3 octets in case of no FS Process Data. IO-Link Safety uses M-Sequence TYPE_2_V.

Only a subset of the IO-Link data types is permitted: Boolean (packed as record), IntegerT(16), and IntegerT(32).

Parameterization within the domain of safety for machinery requires a "Dedicated Tool" per FS-Device or FS-Device family. The Device Tool Interface (DTI) based on proven technology has been chosen for the links between FS-Master Tool, FS-Device, and its "Dedicated Tool". The FS-Master Tool shall provide communication means for a "Dedicated Tool" to allow for the transmission of safety technology parameters (FST parameters) to and from an FS-Device. The "Dedicated Tool" and the FS-Device are both responsible for sufficient means to secure the transmitted data, for example via CRC signature or read-back.

4.6 Parameterization

IO-Link Safety comprises a three-tier concept. The first tier is IODD based and contains all basic non-safety parameters for a Device or FS-Device.

The second tier requires an extension of the IODD for the fixed set of protocol parameters (FSP). These parameters are safety-related and secured via CRC signature against unintended changes of the IODD file. The interpreter of the FS-Master Tool provides a safety-related extension for the handling of the FSP parameters. Usually, the FS-Master Tool is able to determine and suggest the FSP parameter assignments (instance values) automatically and thus relieves the user from assigning these values initially. He can check the plausibility of the values and modify them if required.

The third tier deals with technology specific safety parameters (FST) of an FS-Device. IO-Link Safety classifies two types of FS-Devices. Type "basic" requires only a few orthogonal FST parameters, whereas type "complex" can have a number of FST parameters requiring business rules and verification or validation wizards. Usually, the latter comes already with existing PC software ("Dedicated Tool") used for several functional safety communication profiles for fieldbuses.

The FST parameters for type "basic" are coded as any non-safety parameter within the IODD. They can be modified and downloaded to the FS-Device as usual. However, a diverse second path allows for checking these assignments for correctness. At the end of a parameterization session, the user launches a safety-related "Dedicated Tool" (FS-IOPD) for the calculation of a CRC signature across all FST instance values provided by the FS-Master Tool.

For both types of FS-Devices, the "Dedicated Tool" presents a CRC signature, which the user can copy into one of the FSP parameters. Upon reception of the FSP parameters at start-up, the FS-Device calculates a CRC signature across the locally stored instance values and compares it with the received CRC signature.

This method is used also for the check after using the IO-Link Data Storage mechanism.

4.7 Role of FS-Master and FS-Gateway

The role of the FS-Master is extended to safe monitoring of Process Data, transferred to and from FS-Devices with respect to timeliness, authenticity, and data integrity according to IEC 61784-3. Concerning authenticity, it uses the authenticity code assigned to the FS-Master by the upper level FSCP system and the port number. This prevents from local port related mis-connections and misconnections whenever several FS-Masters are located side by side.

An FS-Master can be equipped by a safety controller, for example according to IEC 61131-6, or vice versa, and thus build-up a stand-alone safety system with its own complete safety functions.

With the help of an FS-Gateway in conjunction with the FS-Master, safety functions can be build-up across the upper level FSCP system using the safety sensors and actuators connected to the FS-Master.

4.8 Mapping to upper level systems

Specification of the mapping to an upper level FSCP system is the responsibility of the particular fieldbus organization. IO-Link Safety made provisions to meet the majority of FSCPs for example via reduced number of data types, descriptions of safety IO data, port selective passivation, and operator acknowledgment signals to prevent from automatic restart of machines.

4.9 Structure of the document

The structure of this document complies mostly with the structure of [1]. Clause 5 specifies the extensions to the Physical Layer (PL), mainly the OSSDe issues, the wake-up behavior, and the additional Port modes. Extensions to SIO are specified in clause 6, those to data link layer (DL) in clause 7, those to system management (SM) in clause 8, those to the FS-Device in clause 9, and those to the FS-Master in clause 10.

The core part of this document is the safety communication layer (SCL) in clause 11. It comprises the SCL services, protocol, state machines, and management. In addition it deals with integrity measures, with protocol (FSP) and technology (FST) parameters, with the integration of "Dedicated Tools" via Tool Calling Interface technology, with port selective passivation, and with SCL diagnosis. Clause 12 complements the core part by functional safety processing either through mapping to the upper level system or local.

Extensions to parameters and commands are specified in Annex A, those to EventCodes in Annex B, and those to data types in Annex C. CRC polynomial issues are presented in Annex D, the IODD aspects in Annex E, the Device Tool Interface technology in Annex F, main scenarios in Annex G, and the system requirements in Annex H. Assessment issues are described in Annex I. Annex J specifies in more detail the "classic" port B and Annex K test issues.

5 Extensions to the Physical Layer (PL)

5.1 Overview

Figure 14 shows the adapted physical layer of an FS-Master (class A).

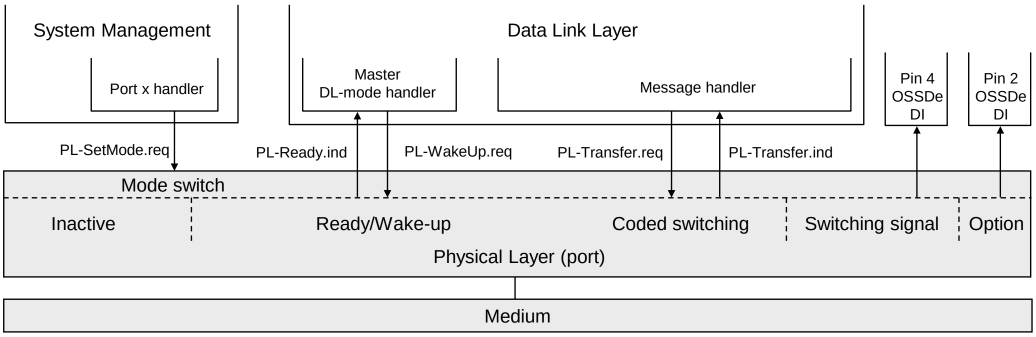


Figure 14 – The IO-Link physical layer of an FS-Master (class A)

Pin 2 and 4 shall be scanned simultaneously to achieve OSSDe functionality. The FS-Master shall scan the C/Q line for the Ready signal of the FS-Device.

Figure 15 shows the adapted physical layer of an FS-Device (class A).

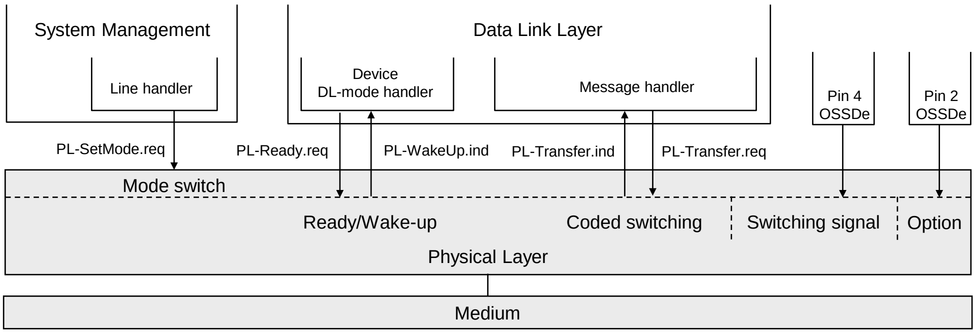


Figure 15 – The IO-Link physical layer of an FS-Device (class A)

Pin 2 and 4 carry the OSSDe signals. The FS-Device shall set the Ready signal after internal safety testing.

5.2 Extensions to PL services

5.2.1 PL_SetMode

The PL-SetMode service is extended by the additional TargetMode "OSSDe" (C/Q line and I/Q line in digital input mode).

5.2.2 PL_Ready

The PL-Ready service initiates or indicates a Ready signal on the C/Q line. Whenever the FS-Device finished its internal safety-related hardware and software tests, it sets this signal. The FS-Master polls this signal and upon reception initiates the wake-up sequence. This unconfirmed service has no parameters. The service primitives are listed in Table 3.

Table 3 – PL_Ready

Parameter name	.req	.ind
<none>		

5.3 Transmitter/receiver

5.3.1 Assumptions for the expansion to OSSDe

Figure 16 shows the cross compatibility between OSSD based safety sensors and OSSDe based FS-Devices.

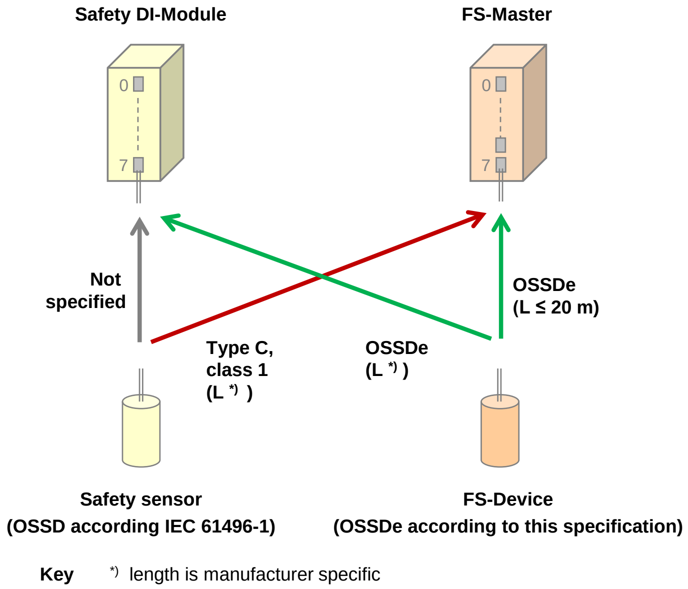


Figure 16 – Cross compatibility OSSD and OSSDe

The following assumptions are the basis for the design of the OSSDe expansion:

- The SIO paradigm of IO-Link shall apply for IO-Link Safety in order to allow manufacturers the combined function of OSSDe and IO-Link Safety communication within one FS-Device.
- A Port on the FS-Master (with "FS-DI" according to Figure 9) shall have fixed configurations as either IO-Link Safety or OSSDe interface with no or minor adjustments in respect to addressing, watchdog times, discrepancy times, or filter times.
- In order to allow OSSD based sensors on the market to be connected to the FS-Master, the FS-DI interface shall support the necessary adjustments for Type "C", class "1" devices according to [12].
- The OSSDe interface shall only be designed as input for the FS-Master port (safety sensors, Class A connectors). Most actuators are supplied by three-phase alternating current such as power drives, low voltage switch gears, motor starters, etc.
- Actuators such as valves with diversity and relays shall be supported by FS-Master with Ports "level d" (see clause 6).

5.3.2 OSSDe specifics

5.3.2.1 General

Similar to the SIO approach, FS-Master according to level "c" support connectivity to existing functional safety devices with OSSDe. OSSDe in this document is defined as two outputs with signals that are both switching in equivalent manner as opposed to antivalent manner, where one signal is normally off and the other normally on (OSSDm).

The FS-Master port is designed to achieve a maximum of possible compatibility to existing OSSD devices using interface type C, class 1 defined in [12].

Figure 17 shows a corresponding reference model from [12], adapted to IO-Link Safety. The information-"source" on the left corresponds for example to a sensor device, whereas the information-"sink" on the right side represents an input of the FS-Master Port class A. Power is supplied by the sink.

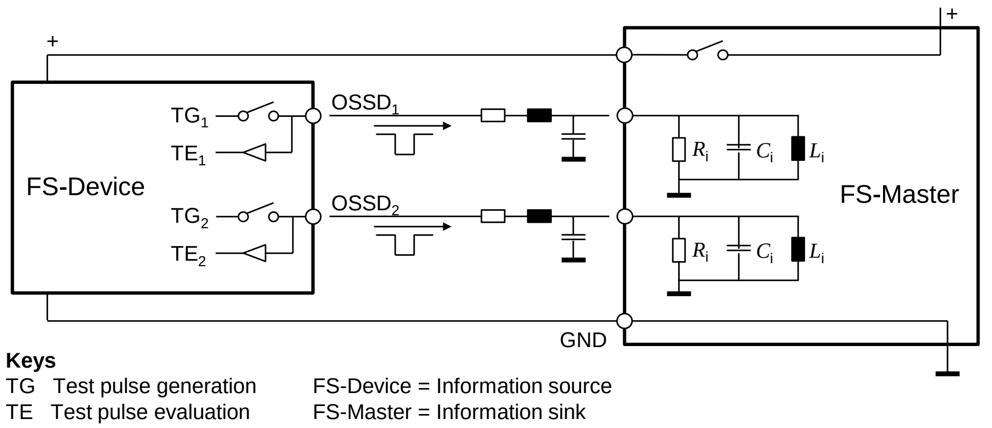


Figure 17 – Principle OSSDe function

The worst case values for the line resistance and capacitance are defined in Table 9. In case of IO-Link Safety, line inductance is negligible at a length of 20 m. The design of the FS-Master Port shall ensure values for R_i, C_i, and L_i guaranteeing proper signal behavior according to Table 8.

Table 4 shows the OSSD states and conditions defined in IEC 61496-1:2012.

Table 4 – OSSD states and conditions

State	Cause	Voltage range	Current
OFF	Demand	- 3 V to + 2 V r.m.s (+ 5 V peak)	< 2 mA (leakage) NOTE
ON	No demand	+ 11 V to + 30 V	> 6 mA
NOTE IEC 61131-9 permits 5 mA for the voltage range of 5 V to 15 V			

OFF state:

For this interface, the OFF state is defined as the "powerless" state, where voltage and current of at least one OSSDe shall be within (voltage) and below (current) defined limits (see Table 4). If the safety function is demanded, or the source (the device) detects a fault, the OSSDe signals shall go to the OFF state. Antivalent voltage levels, so-called discrepancy, on both OSSDe outputs of the device shall be treated as OFF state. The duration of this state shall be within a specified discrepancy tolerance time. If the tolerance time is exceeded the port is considered to be faulty.

ON state:

For this interface, the ON state is defined as the "powered" state, where voltage and current on both OSSDe outputs shall be within the voltage range and above defined current limits, when sinked by IEC 61131-2 inputs (see Table 4). Test pulses within specified ranges in voltage levels, durations and intervals are permitted. Antivalent voltage levels, so-called discrepancy, on both OSSDe outputs of the device shall be treated as OFF state.

5.3.2.2 Detection of cross connection faults

Tests are required for the detection of the cross connection faults specified in IEC 61496-1 and shown in Table 5.

Table 5 – Cross connection faults

Fault	Diagnostics
Short circuit between OSSD 1 and OSSD 2	Test pulses (runtime diagnosis)

Fault	Diagnostics
Short circuit between OSSD 1 or OSSD 2 and V+	Test pulses (runtime diagnosis)
Short circuit between OSSD 1 or OSSD 2 and V-	Test pulses (runtime diagnosis)
Open circuit of the power supply return cable (V-)	Type test, maximum leakage current
Open circuit of the functional earth (bonding) conductor	Type test, no functional earth
Open circuit of the screen of screened cable	Not required due to no shielding
Incorrect wiring	Discrete wiring only, organizational issue (test during commissioning)

The means for detecting short circuits are test pulses at runtime. The means for testing the behavior in case of open circuits is the type test during the assessment. Figure 18 shows the test pulses approach for the detection of cross connection faults.

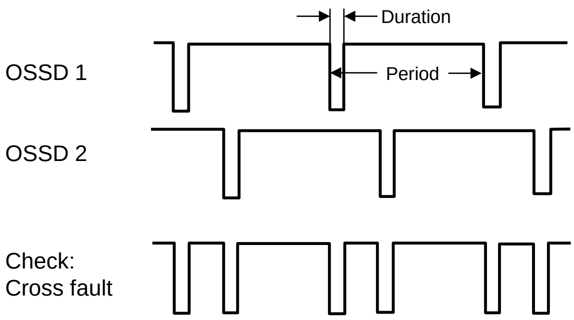


Figure 18 – Test pulses to detect cross connection faults

Three methods of testing (intervals) are commonly used:

- Test pulses at each program cycle of the safety device (dependency on configuration)
- Test pulses at fixed times
- Test pulses after any commutation OFF → ON

5.3.2.3 FS-Device OSSDe output testing

The test pulses of this interface type for testing the transmission line are created and also evaluated on the safety device side. This way the source is able to diagnose the correct functioning of the output stage. In case of any detected error both OSSDe outputs shall be switched to the safe state (Lock-out condition = OFF).

The test pulses are created in a periodic manner on both OSSD lines. In order to detect short circuits between the lines or between the lines and power-supply, the test pulses of both lines can be time-shifted to each other (see Figure 19).

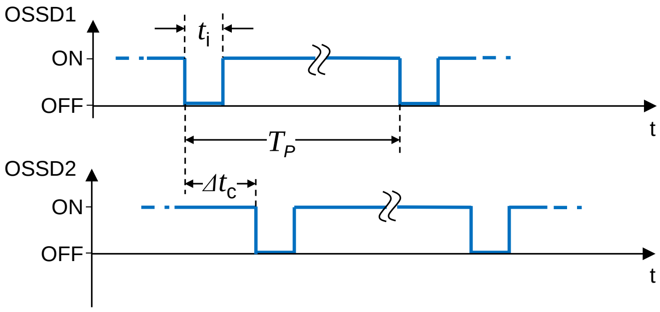


Figure 19 – OSSD timings

The following parameters specify the characteristics of the test pulses on the OSSD interface:

- Period of test pulses (T_P)

- Duration of test pulses (t_i)
- Time-shift between test pulses of both channels (Δt_c)

The characteristics of test pulses are classified in [12]. FS-Devices shall meet type C and class 1 requirements with a test pulse length $t_i \leq 1000$ µs (see Table 7).

5.3.3 Start-up of an FS-Device (Ready pulse)

Figure 20 shows the typical start-up sequence of an OSSD sensor without IO-Link Safety capability. During self-test for functional safety, both OSSD signals shall be OFF. When finished, the sensor switches to ON and starts test pulses. A demand causes the sensor to switch OFF. A fault causes the sensor to switch to lock-out condition (OFF) and to remain in this state until repair.

NOTE For simplicity, the figure shows only one OSSD channel.

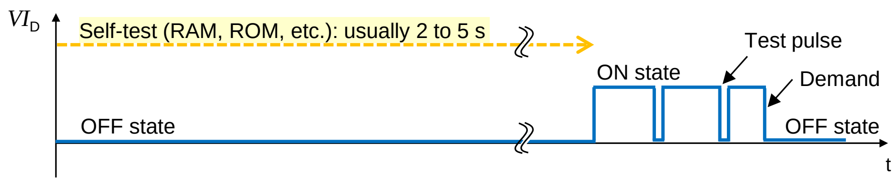


Figure 20 – Typical start-up of an OSSD sensor

Figure 21 shows the start-up of an FS-Device with OSSDe capability connected to a classic FS-DI module.

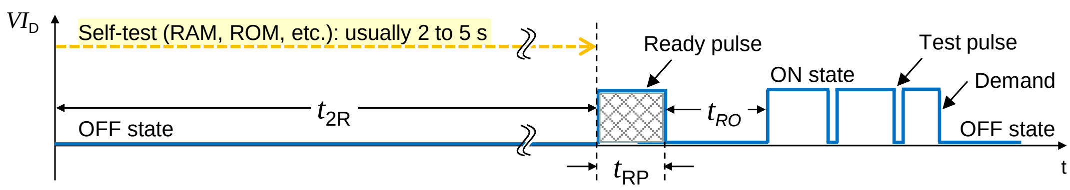


Figure 21 – Start-up of an FS-Device

In contrast to a classic sensor, the FS-Device provides only on pin 4 (see Figure 9) a so-called Ready-pulse of a certain length to indicate the FS-Master its readiness after self-testing. After a certain recovery time, the FS-Device switches to ON and starts test pulses like a classic safety sensor.

Timings and Wake-up behavior of the FS-Device are specified in 5.7.

5.3.4 Electric characteristics of a receiver in FS-Device and FS-Master

The voltage range and switching threshold definitions are the same for FS-Master and FS-Device since FS-Master ports shall be able to operate with non-safety IO-Link Devices. The definitions in Table 6 apply.

Table 6 – Electric characteristics of a receiver

Property	Designation	Minimum	Typical	Maximum	Unit	Remark
$VTHH_{D,M}$	Input threshold 'ON'	10,5	n/a	13	V	See NOTE 1
$VTHL_{D,M}$	Input threshold 'OFF'	8	n/a	11,5	V	See NOTE 1
$VHYS_{D,M}$	Hysteresis between input thresholds 'ON' and 'OFF'	0	n/a	n/a	V	Shall not be negative See NOTE 2
$VIL_{D,M}$	Permissible voltage range 'OFF'	$V0_{D,M}$ -1,0	n/a	n/a	V	With reference to relevant negative

Property	Designation	Minimum	Typical	Maximum	Unit	Remark
						supply voltage
$VIH_{D,M}$	Permissible voltage range 'ON'	n/a	n/a	$V+_{D,M}$ + 1,0	V	With reference to relevant positive supply voltage.
NOTE 1 Thresholds are compatible with the definitions of type 1 digital inputs in IEC 61131-2.						
NOTE 2 Hysteresis voltage $VHYS = VTHH - VTHL$						

Figure 22 demonstrates the switching thresholds for the detection of OFF and ON signals.

NOTE 'OFF' and 'ON' correspond to 'L' (Low) and 'H' (High) in [1] and [2].

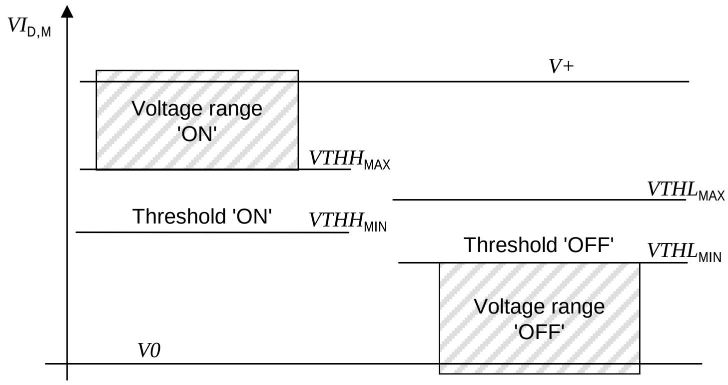


Figure 22 – Switching thresholds for FS-Device and FS-Master receivers

The FS-Master ignores pulses below 11 V (max. 15 mA or max. 30 mA) that are shorter than 1 ms.

5.4 Electric and dynamic characteristics of an FS-Device

In general, the specified values and ranges of [1] or [2] apply (see Figure 23).

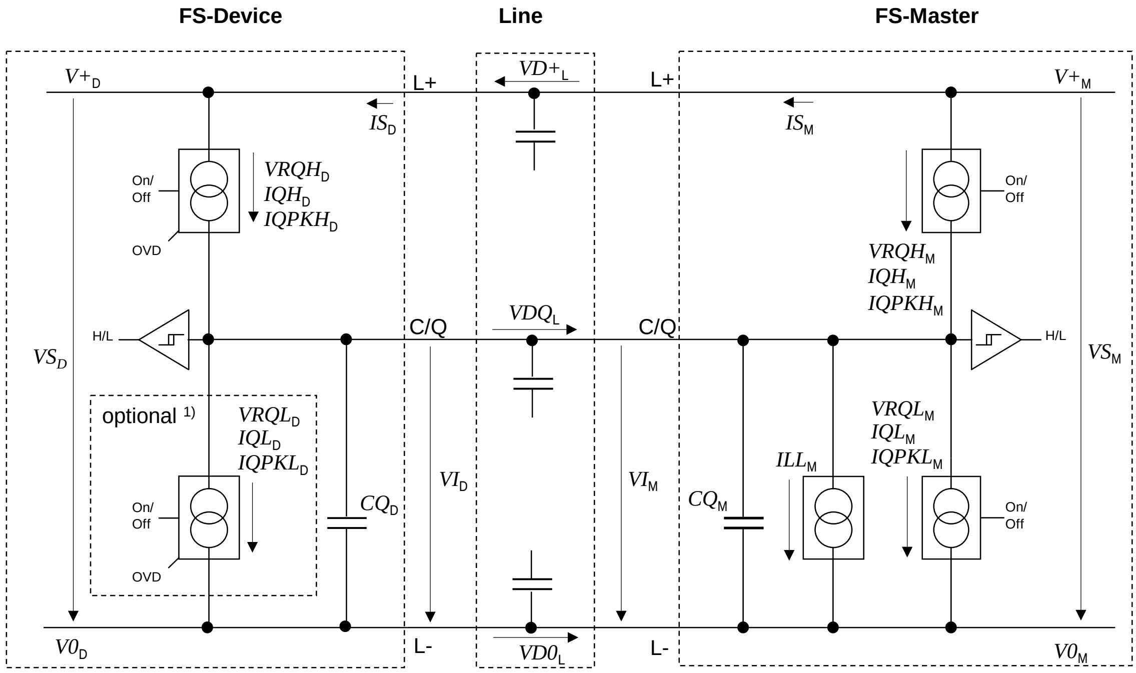


Figure 23 – Reference schematics (one OSSDe channel)

The subsequent illustrations and parameter tables refer to the voltage level definitions in Figure 24.

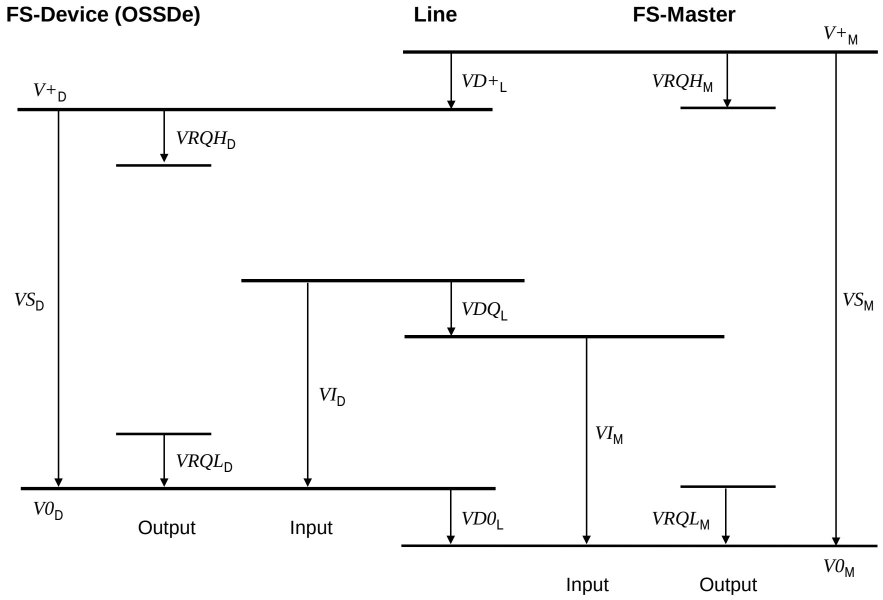


Figure 24 – Voltage level definitions

The electric and dynamic parameters for the OSSDe interface of an FS-Device are specified in Table 7.

Table 7 – Electric and dynamic characteristics of the FS-Device (OSSDe)

Property	Designation	Minimum	Typical	Maximum	Unit	Remark
VS_D	Supply voltage	18	24	30	V	See Figure 24
ΔVS_D	Ripple	n/a	n/a	1,3	V_{pp}	Peak-to-peak absolute value limits shall not be exceeded. f_{ripple} = DC to 100 kHz
IS_D	Supply current	n/a	n/a	1000	mA	See 5.9
QIS_D	Power-up consumption	n/a	n/a	70	mAs	See equation (1) and associated text
$VRQH_D$	Residual voltage 'ON'	n/a	n/a	3	V	Voltage drop compared with $V+_D$ (IEC 60947-5-2)
$VRQL_D$	Residual voltage 'OFF'	n/a	n/a	3	V	Voltage drop compared with $V0_D$ NOTE 1
IQH_D	DC driver current P-switching output ('ON' state)	50	n/a	minimum ($IQPKL_M$)	mA	Minimum value due to fallback to digital input in accordance with IEC 61131-2, type 2
IQL_D	DC driver current N-switching output ('ON' state)	0	n/a	minimum ($IQPKH_M$)	mA	Only for push-pull output stages
IQQ_D	Quiescent current to $V0_D$ ('OFF' state)	0	n/a	15	mA	Pull-down or residual current with deactivated output driver stages
CQ_D	Input capacitance	0	n/a	1,0	nF	Effective capacitance between C/Q and L+ or L- of Device in receive state. See [1] for constraints on transmission rates.

Property	Designation	Minimum	Typical	Maximum	Unit	Remark
t_{2R}	Time to Ready-pulse	n/a	n/a	5	s	See Figure 21; Parameter in IODD
t_{RP}	Duration of Ready pulse	500	n/a	1000	µs	See Figure 21
t_{RW}	End of Ready pulse to ready for Wake-up	n/a	n/a	50	µs	See Figure 27 – Start-up of an FS-Device
t_{RO}	End of Ready pulse to OSSD mode	700	n/a	Data sheet	µs	See Figure 21
T_P	Period of test pulses	10	n/a	Data sheet	ms	See [12] and Figure 19
t_i	Test pulse duration	n/a	n/a	1000	µs	See Figure 19.
t_{dis}	Discrepancy time	n/a	n/a	3	ms	Demands may occur during tests

NOTE 1 Pull-down of residual voltage with deactivated high-side output driver stage and activated low-side driver stages (if available e.g. push-pull drivers) with externally limited DC driver current of 50 mA maximum

NOTE 2 Characteristics in this table assume OSSD type "C", class "1" according to [12] and interface type 1 according to IEC 61131-2

It is the responsibility of the FS-Device designer to select appropriate ASICs according to [1] and/or to provide mitigating circuitry to meet the requirements of IEC 61496-1.

The FS-Device shall be able to reach a stable operational state (ready for Wake-up: T_{RDL}) while consuming the maximum charge (see equation (1)).

$$QIS_D = ISIR_M \times 50ms + \left(T_{RDL} - 50\ ms\right) \times IS_M \qquad (1)$$

5.5 Electric and dynamic characteristics of an FS-Master port (OSSDe)

In general, the specified values and ranges of [1] or [2] apply (see Figure 23 and Figure 24). The definitions in Table 8 are valid for the electrical characteristics of an FS-Master port.

Table 8 – Electric and dynamic characteristics of the Port interface

Property	Designation	Minimum	Typical	Maximum	Unit	Remark
VS_M	Supply voltage for FS-Devices	20	24	30	V	See Figure 24
IS_M	Supply current for FS-Devices	200	n/a	1000	mA	Rules in 5.9. shall be considered
$ISIR_M$	Current pulse capability for FS-Devices	400	n/a	n/a	mA	See Figure 25
ILL_M	Load or discharge current for 0 V < VI_M < 5 V 5 V < VI_M < 15 V 15 V< VI_M < 30 V	 0 5 5	 n/a n/a n/a	 15 15 15	 mA mA mA	See NOTE 1
$VRQH_M$	Residual voltage 'H'	n/a	n/a	3	V	Voltage drop relating to $V+_M$ at maximum driver current IQH_M

Property	Designation	Minimum	Typical	Maximum	Unit	Remark
$VRQL_M$	Residual voltage 'L'	n/a	n/a	3	V	Voltage drop relating to $V0_M$ at maximum driver current IQL_M
IQH_M	DC driver current 'H'	100	n/a	n/a	mA	
$IQPKH_M$	Output peak current 'H'	500	n/a	n/a	mA	Absolute value See NOTE 2
IQL_M	DC driver current 'L'	100	n/a	n/a	mA	
$IQPKL_M$	Output peak current 'L'	500	n/a	n/a	mA	Absolute value See NOTE 2
CQ_M	Input capacitance	n/a	n/a	1,0	nF	f=0 MHz to 4 MHz
NOTE 1 Currents are compatible with the definition of type 1 digital inputs in IEC 61131-2. However, for the range 5 V < VI_M < 15 V, the minimum current is 5 mA instead of 2 mA in order to achieve short enough slew rates for pure p-switching Devices. NOTE 2 Wake-up request current (See 5.3.3.3 in [1] or [2]).						

The Master shall provide a charge of at least 20 mAs within the first 50 ms after power-on without any overload-shutdown (see Figure 25). After 50 ms the current limitations for IS_M in Table 8 apply.

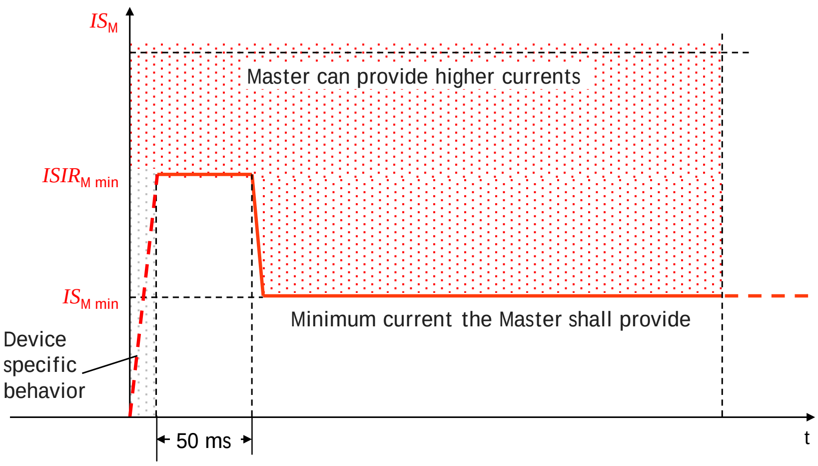


Figure 25 – Charge capability at power-up

5.6 FS-Master port FS-DI interface

Since OSSD safety sensors can provide different test pulse patterns, the FS-Master Port shall have a suitable input filter, or evaluation algorithm. For the sake of EMC considerations, by a combination of both can be used. This means, that the time, in which the signal is below U_{Hmin} must be less than the maximum allowed test pulse duration.

Any state different to both signals "high", except test pulses, shall be interpreted as safe state.

NOTE Achievable reaction times: IO-Link non safe: min. 600 µs, PROFINET: 1 ms, non-synchronized system: 2 ms

The EMC levels shall be taken into account for the layout of an input filter. The communication transmission rate 230 kbit/s conflicts with the input filter. Possible conflict resolution is shown in Figure 26.

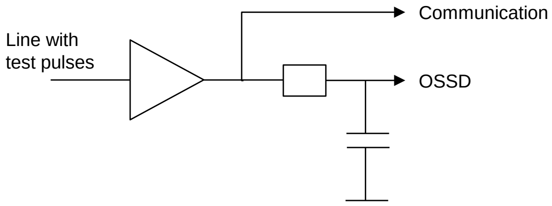


Figure 26 – OSSDe input filter conflict resolution

In general, the specified values and ranges of [1] or [2] apply. Basis is interface type 1 of IEC 61131-2. Deviating and supplementary electric and dynamic parameters for the FS-DI interfaces are specified in Table 8.

5.7 Wake-up coordination

Figure 27 shows the start-up of an FS-Device (see [1] for standard timing definitions). After accomplished self-tests, it indicates its readiness for Wake-up through an ON/Ready pulse on the C/Q line. If no Wake-up occurs within a defined time frame, it starts with test pulses (see Figure 20).

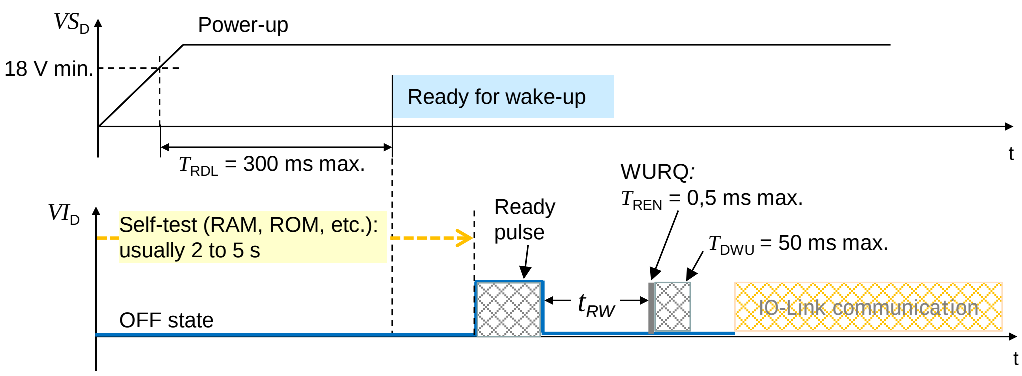


Figure 27 – Start-up of an FS-Device

NOTE Actually some safety light curtain vendors offer activation of functionality if some connection conditions are activated during start-up phase (e.g. override)

5.8 Fast start-up

Figure 28 illustrates required fast start-up non-safety and safety timings.

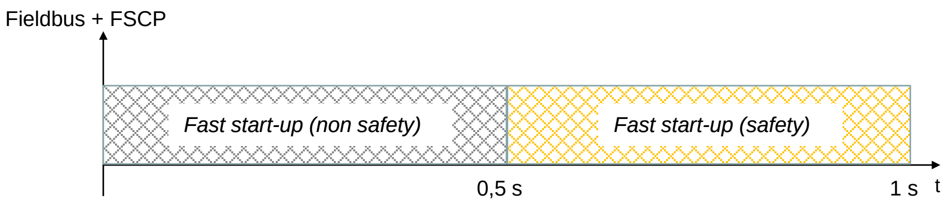


Figure 28 – Required fast start-up timings

Current safety devices usually require 2 to 5 seconds for self-testing prior to functional safe operation. The Ready-pulse concept allows for easier achievable realizations of these requirements.

5.9 Power supply

An FS-Master port shall be able to switch its power supply on and off. This enables the FS-Master to restart an FS-Device once it failed to establish communication and started OSSDe operation instead.

The FS-Master port is the only power supply for IO-Link related parts of the FS-Device. Any external power source of the FS-Device shall be totally nonreactive to these parts.

FS-Master shall provide all ports with a minimum supply of 200 mA and at least one port with a minimum supply of 1000 mA. The FS-Master shall specify the total maximum current consumption of all its ports and the derating rules.

Higher currents can conflict with the power switching components and cause interference with the signal lines. The "ripple" requirement in Table 7 shall be considered. The overall cable loop resistance shall be not more than 1,2 Ω (see Table 8 and Table 9).

5.10 Medium

5.10.1 Constraints

For the sake of simplicity in technology and commissioning, IO-Link Safety expects a wired point-to-point connection or equivalent consistent transmission and powering between FS-Master and an FS-Device. No storing elements in between are permitted.

5.10.2 Connectors

Connectors as specified in [1] for Class A are permitted.

5.10.3 Cable characteristics

Table 9 shows the cable characteristics for IO-Link Safety and non-safety Devices, if higher power supply currents than 200 mA are applied.

Table 9 – Cable characteristics

Property	Designation	Minimum	Typical	Maximum	Unit
L	Cable length	0	n/a	20	m
RL_{eff}	Overall loop resistance	n/a	n/a	1,2	Ω
CL_{eff}	Effective line capacitance	n/a	n/a	3,0	nF (<1 MHz)
NOTE These characteristics can deviate from the original characteristics defined in [1] or [2].					

6 Extensions to SIO

SIO is only defined for Pin 4 of the Master/Device port in [1]. OSSDe requires inclusion of Pin 2 as specified in clause 5. Configuration can be performed within the Master/Device applications layer (see Figure 31 and Figure 35).

7 Extensions to data link layer (DL)

7.1 Overview

Figure 31 and Figure 35 show the DL building blocks of FS-Device and FS-Master. No new or changed services are required. However, both DL-mode handlers are extended by the Ready-pulse feature as shown in 7.2 and 7.3.

7.2 State machine of the FS-Master DL-mode handler

Figure 29 shows the modifications of the FS-Master DL-mode handler versus the Master DL-mode handler in [1].

A new state "WaitOnReadyPulse_10" considers the requirement for the FS-Master to wait on the Ready-pulse of an FS-Device (see 5.7) prior to establish communication via DL_SetMode_STARTUP.

The maximum waiting time is t_{2R} as defined in Table 7. Whenever the time expired, the FS-Master shall run a power-OFF/ON cycle for the connected FS-Device in order to initiate a retry for another Ready-pulse.

The criterion to use the extra path is the guard [safety], which is derived from the new port configuration "FS_PortModes" (see 10.2.2).

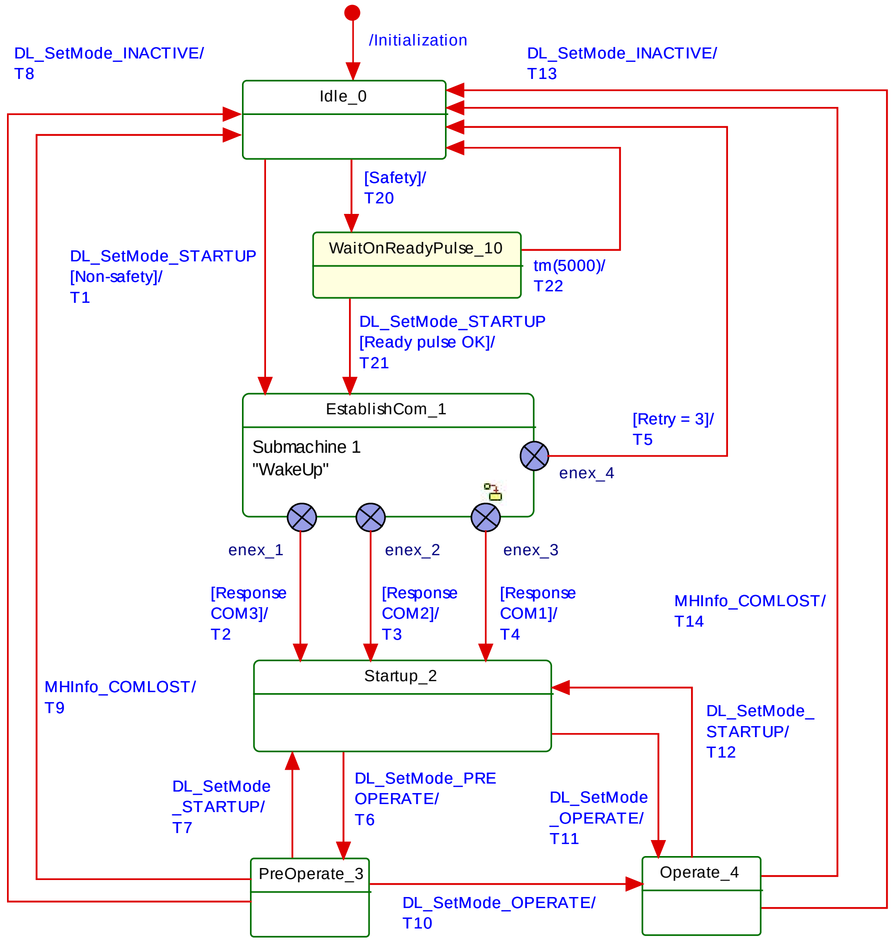


Figure 29 – State machine of the FS-Master DL-mode handler

Table 10 shows the additional state and transitions as well as internal items considering the Ready-pulse feature.

Table 10 – State transition tables of the FS-Master DL-mode handler

STATE NAME	STATE DESCRIPTION
Idle_0 to SM: Retry_9	See Table 42 in [1]
WaitOnReadyPulse_10	Waiting on the Ready-pulse from the FS-Device. A timer of 5 s is started.

TRANSITION	SOURCE STATE	TARGET STATE	ACTION
T1 to T19	*	*	See Table 42 in [1]
T20	0	10	This path is taken only if the new configuration parameter "Safety" has been assigned to "SafetyCom" or "MixedSafetyCom" respectively
T21	10	1	Set Retry = 0.
T22	10	0	FS-Master was not able to detect a Ready-pulse within 5 s. It will initiate a power OFF/ON cycle for the FS-Device to retry the Ready-pulse.

INTERNAL ITEMS	TYPE	DEFINITION
MH_xxx to xx_Conf…	Call	See Table 42 in [1]
Safety	Guard	New configuration parameter "Safety": either value "SafetyCom" or "MixedSafetyCom"
Ready pulse OK	Guard	Ready-pulse detected

7.3 State machine of the FS-Device DL-mode handler

Figure 30 shows the modifications of the FS-Device DL-mode handler versus the Device DL-mode handler in [1].

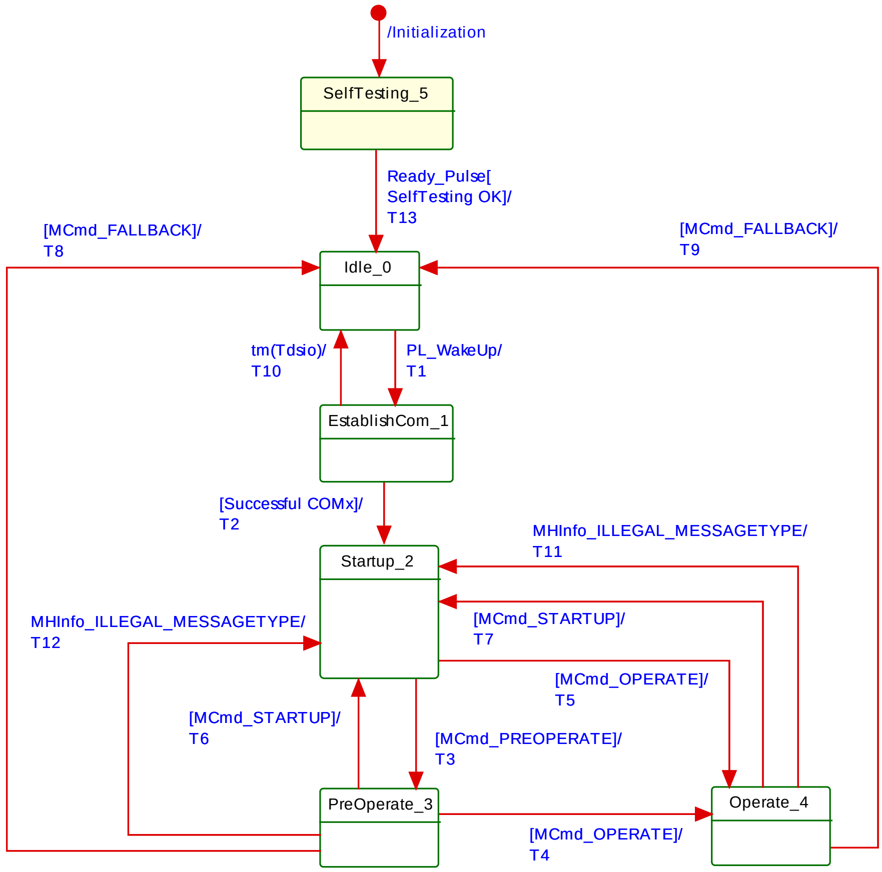


Figure 30 – State machine of the FS-Device DL-mode handler

A new state "SelfTesting_5" considers the requirement for the FS-Device to indicate its readiness for a wake-up procedure after its internal safety self-testing via a test pulse in pin 4. Self-testing may actually take more than the maximum permitted start-up time T_{RDL} of a non-safety Device (see 5.7).

Table 11 – State transition tables of the FS-Device DL-mode handler

STATE NAME	STATE DESCRIPTION
Idle_0 to Operate_4	See Table 43 in [1]
SelfTesting_5	Safety check through self-testing of µC, RAM, etc. This may take more than the permitted start-up time T_{RDL} of a non-safety Device.

TRANSITION	SOURCE STATE	TARGET STATE	ACTION
T1 to T12	*	*	See Table 43 in [1]
T13	5	0	Create a signal (Ready_Pulse) on pin 4 for duration of t_{RP}, when self-testing is completed.

INTERNAL ITEMS	TYPE	DEFINITION
T_{RDL}	Time	See Table 7 10 in [1]
t_{RP}	Time	See Table 7
Self-testing OK	Guard	Self-testing completed

8 Extensions to system management (SM)

There are no extensions to system management.

9 Extensions of the FS-Device

9.1 Principle architecture and models

9.1.1 FS-Device architecture

Figure 31 shows the principle architecture of the FS-Device. It does not include safety measures for implementation such as redundancy for the safety-related parts.

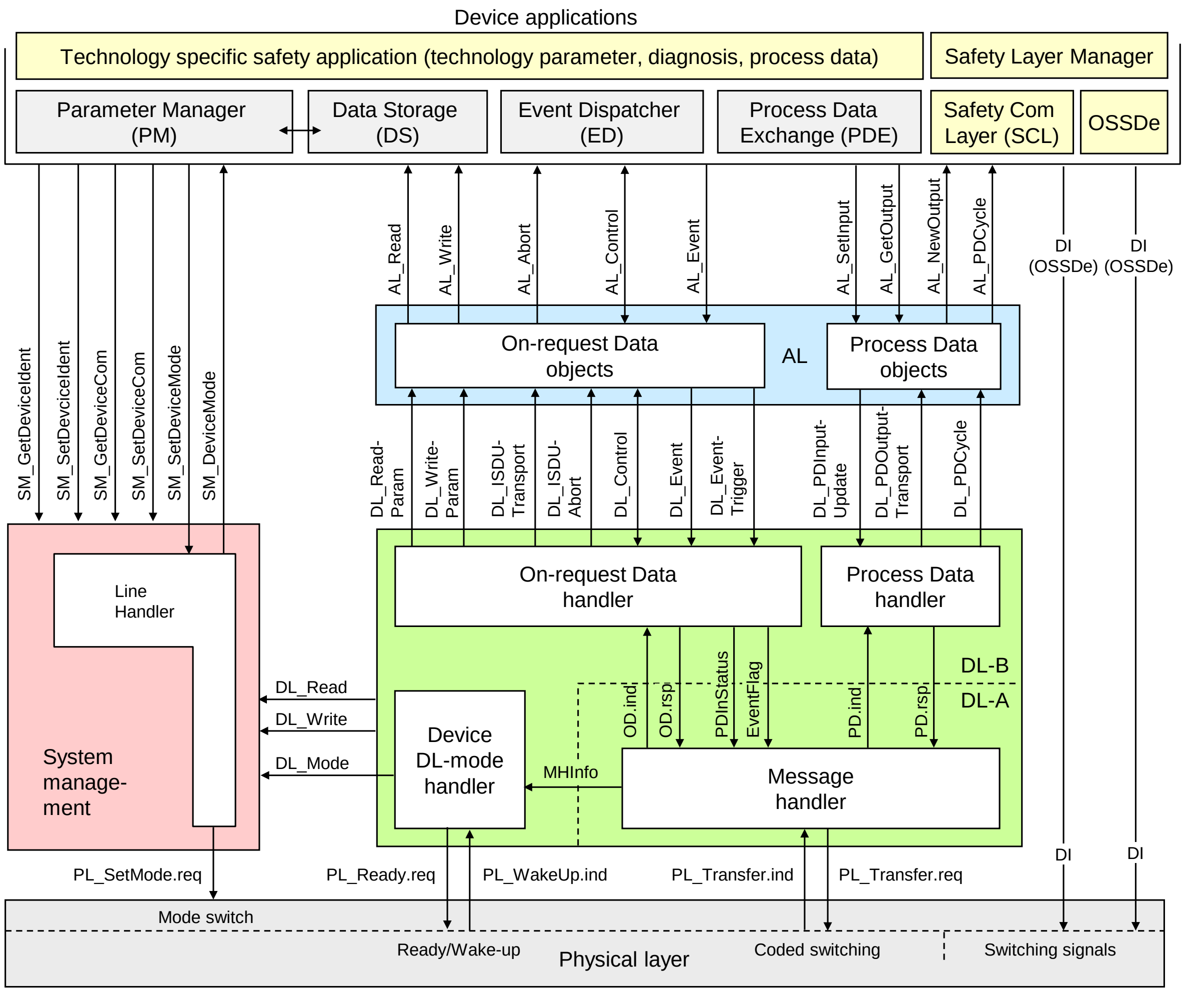


Figure 31 – Principle architecture of the FS-Device

An FS-Device comprises first of all the technology specific functional safety application. "Emergency switching off" safety devices for example can be designed such that "classic" OSSDe operation or safety communication can be configured. A Safety Layer Manager is responsible for the handling of a safety bit via the OSSDe building block or a safety PDU using the Safety Communication Layer (see clause 11).

9.1.2 FS-Device model

According to the requirement of mixed NSR and SR parameter and process data, the FS-Device model has been modified and adapted.

That means the FS-Device Index model is split into an NSR and an SR part. Figure 32 shows the areas of concern. The allocation of the SR part ("FSP" parameter) is defined within the IODD of the FS-Device.

During commissioning, the assignment of FSP parameter values take place. These instance values are secured by CRC signatures and transferred as a block to the FS-Master and to the FS-Device (see 11.7.5). At each start-up of an FS-Device, the stored FSP block in the FS-Master is transferred again and the FS-Device can check the locally stored instance parameter values for integrity via CRC signatures. This check includes technology specific "FST" parameters, which are not transferred at each start-up. The FS-Device displays its FSP parameters at predefined Indices read-only.

Technology specific parameters (FST) could be handled either in an open manner to a certain extend as standard non-safety parameters (see 11.7.8) or in a protected manner in hidden internal memory (see 11.7.9).

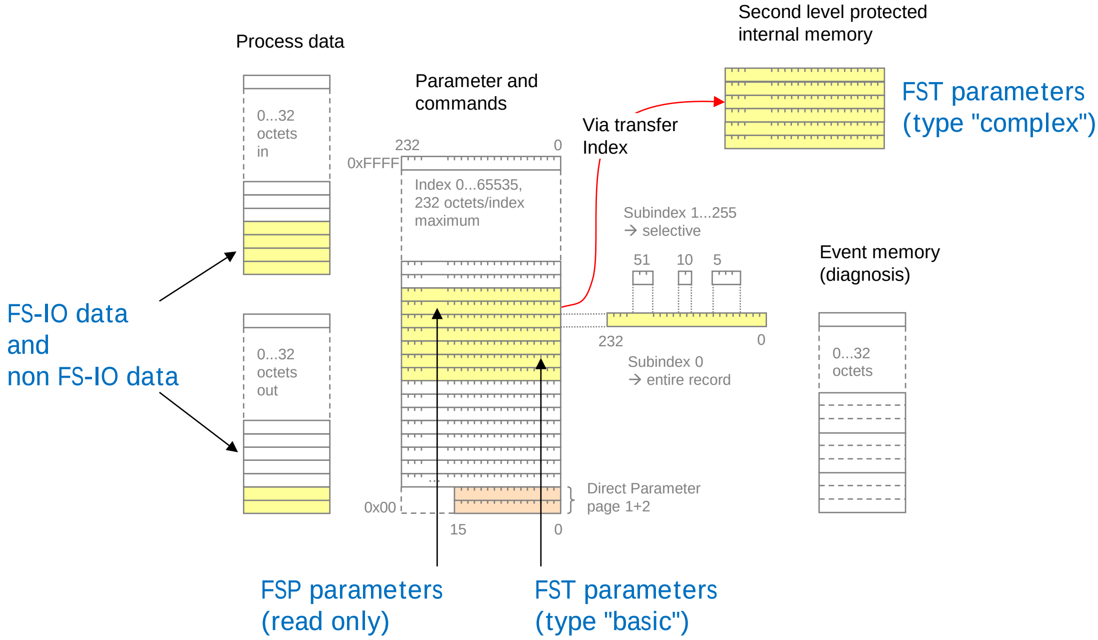


Figure 32 – The FS-Device model

The maximum space for FS-I/O data and non FS-I/O data to share is 32 octets. The space shall be filled with FS-I/O data first followed by the non FS-I/O data. The border is variable. Assuming a maximum safety protocol trailer of 5 octets, the maximum possible space for FS-I/O data is 27 octets.

9.2 Parameter Manager (PM)

There are no extensions or modifications of the Parameter Manager required.

9.3 Process Data Exchange (PDE)

Depending on "Safety" configuration, Process Data Exchange takes over or passes FS-Process Data (see 11.4.3 Safety PDU) from/to the Safety Layer Manager.

9.4 Data Storage (DS)

9.4.1 General considerations including safety

The technology specific (FST) parameters are secured by a particular CRC signature (FSP_TechParCRC) included in the FSP parameter set. Additional Authenticity parameters are used in case of FS-Device replacement. Thus, the standard Data Storage mechanism can be used for FS-Device replacement. This document specifies a straighter forward version of standard Data Storage compliant with [1].

This version of Data Storage requires that Device Access Lock (Index 0x000C) bit "0" and "1" shall always be unlocked (= "0").

9.4.2 User point of view

The Data Storage mechanism for FS-Devices is based on the general mechanism for non-safety-related Devices. It is described here from a holistic user's point of view as best practice pattern (system description). This is in contrast to current [1] or [2], where Device and Master are described separately and with more features then used within this concept.

9.4.3 Operations and preconditions for Device replacement

9.4.3.1 Purpose and objectives

Main purpose of the IO-Link Data Storage mechanism is the replacement of obviously defect Devices or Masters by spare parts (new or used) without using configuration, parameterization, or other tools. The scenarios and associated preconditions are described in the following clauses.

9.4.3.2 Preconditions for the activation of the Data Storage mechanism

The following preconditions shall be observed prior to the usage of Data Storage:

(1) Data Storage is only available for *Devices* and *Masters* implemented according to [1] or [2] or later releases (> V1.1).

(2) The *Inspection Level* of that Master port the Device is connected to shall be adjusted to "type compatible" (corresponds to "TYPE_COMP" within Table 78 in [1]).

(3) The *Backup Level* of that Master port the Device is connected to shall be either "Backup/Restore" or "Restore", which corresponds to DS_Enabled in 11.2.2.6 in [1]. See 9.4.5 within this document for details on *Backup Level*.

9.4.3.3 Preconditions for the types of Devices to be replaced

After activation of a Backup Level (Data Storage mechanism) a "faulty" Device can be replaced by a type equivalent or compatible other Device. In some exceptional cases, for example non-calibrated Devices, a user manipulation is required such as teach-in, to guarantee the same functionality and performance.

Thus, two types of Devices exist in respect to exchangeability, which shall be described in the user manual of the particular Device:

Data Storage class 1: automatic DS

The configured Device supports Data Storage in such a manner that the replacement Device plays the role of its predecessor fully automatically and with the same performance.

Data Storage class 2: semi-automatic DS

The configured Device supports Data Storage in such a manner that the replacement Device requires user manipulation such as teach-in prior to operation with the same performance.

9.4.3.4 Preconditions for the parameter sets

Each Device operates with the configured set of active parameters. The associated set of backup parameters stored within the system (Master and upper level system, for example PLC) can be different from the set of active parameters (see Figure 33).

A replacement of the Device in operation will result in an overwriting of the existing parameters within the newly connected Device by the backup parameters.

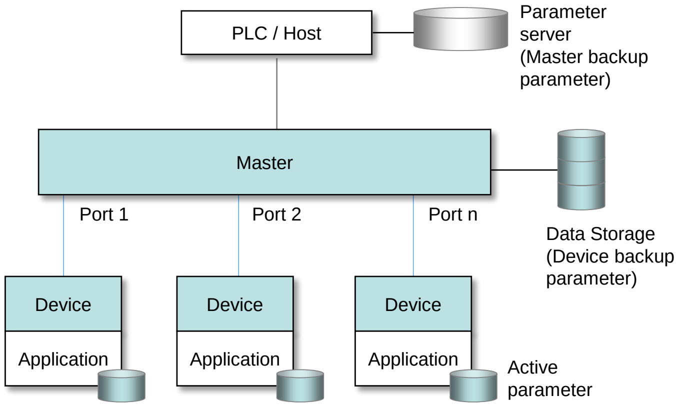


Figure 33 – Active and backup parameter

9.4.4 Commissioning

9.4.4.1 On-line commissioning

Usually, the Devices are configured and parameterized along with the configuration and parameterization of the fieldbus and PLC system with the help of engineering tools. After the user assigned values to the parameters, they are downloaded into the Device and become active parameters. Upon a system command, these parameters are uploaded (copied) into the Data Storage within the Master, which in turn will initiate a backup of all its parameters depending on the features of the upper level system.

In case of functional safety, commissioning cannot be completed without verification and validation of FSP and FST parameters as well as of entire safety functions according to the relevant safety manuals.

9.4.4.2 Off-site commissioning

Another possibility is the configuration and parameterization of Devices with the help of extra tools such as "USB-Masters" and the IODD of the Device away (off-site) from the machine/ facility (see Figure 34).

The "USB-Master" tool will arm the parameter set after configuration, parameterization, and validation (to become "active") and mark it via a non-volatile flag (see Table 13). After installation in the machine/facility these parameters are uploaded (copied) automatically into the Data Storage within the Master (backup).

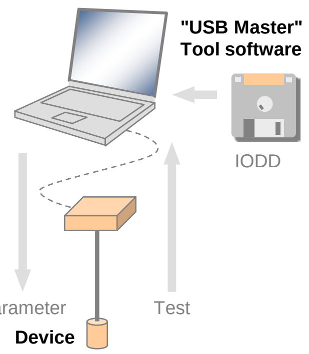


Figure 34 – Off-site commissioning

9.4.5 Backup Levels

9.4.5.1 Purpose

Within an automation project with IO-Link usually three situations with different user requirements for backup of parameters via Data Storage can be identified:

- commissioning ("Disable");

- production ("Backup/Restore");
- production ("Restore").

Accordingly, three different "Backup Levels" are defined allowing the user to adjust the system to the particular functionality such as for Device replacement, off-site commissioning, parameter changes at runtime, etc.

These adjustment possibilities lead for example to drop-down menu entries for "Backup Level".

9.4.5.2 Overview

Table 12 shows the recommended practice for Data Storage within an IO-Link system. It simplifies the activities and their comprehension since activation of the Data Storage implies transfer of the parameters.

Table 12 – Recommended Data Storage Backup Levels

Backup Level	Data Storage adjustments	Behavior
Commissioning ("Disable")	Master port: Activation state: "DS_Cleared"	Any change of active parameters within the Device will *not* be copied/saved. Device replacement *without* automatic/semi-automatic Data Storage.
Production ("Backup/Restore")	Master port: Activation state: "DS_Enabled" Master port: UploadEnable Master port: DownloadEnable	Changes of active parameters within the Device will be copied/saved. Device replacement *with* automatic/semi-automatic Data Storage supported.
Production ("Restore")	Master port: Activation state: "DS_Enabled" Master port: UploadDisable Master port: DownloadEnable	Any change of active parameters within the Device will *not* be copied/saved. If the parameter set is marked to be saved, the "frozen" parameters will be restored by the Master. However, Device replacement *with* automatic/semi-automatic Data Storage *of frozen parameters* is supported.

Legacy rules and presetting:

- For Devices according to [1] with preset *Inspection Level* "NO_CHECK" only the *Backup Level* "Commissioning" shall be supported. This should also be the default presetting in this case.
- For Devices according to [1] with preset *Inspection Level* "TYPE_COMP", all three *Backup Levels* shall be supported. Default presetting in this case should be "Backup/Restore".
- For Devices according to [1] with preset *Inspection Level* "IDENTICAL", only the *Backup Level* "Commissioning" shall be supported.

The following clauses describe the phases in detail.

9.4.5.3 Commissioning ("Disable")

The Data Storage is disabled while in commissioning phase, where configurations, parameterizations, and PLC programs are fine-tuned, tested, and verified. This includes the involved IO-Link Masters and Devices. Usually, saving (upload) the active Device parameters makes no sense in this phase. As a consequence, the replacement of Master and Devices with automatic/semi-automatic Data Storage is not supported.

9.4.5.4 Production ("Backup/Restore")

The Data Storage will be enabled after successful commissioning. Current active parameters within the Device will be copied (saved) into backup parameters. Device replacement with automatic/semi-automatic Data Storage is now supported via download/copy of the backup parameters to the Device and thus turning them into active parameters.

Criteria for the particular copy activities are listed in Table 13. These criteria are the conditions to trigger a copy process of the active parameters to the backup parameters, thus ensuring the consistency of these two sets.

Table 13 – Criteria for backing up parameters ("Backup/Restore")

User action	Operations	Data Storage
Commissioning session (see 9.4.4.1)	Parameterization of the Device via Master tool (on-line). Transfer of active parameter(s) to the Device will cause backup activity.	Master tool sends ParamDownloadStore; Device sets "DS_Upload" flag and then triggers upload via "DS_UPLOAD_REQ" Event. "DS_Upload" flag is deleted as soon as the upload is completed.
Switching from commissioning to production	Restart of Port and Device because Port configuration has been changed	During system startup, the "DS_Upload" flag triggers upload (copy). "DS_Upload" flag is deleted as soon as the upload is completed
Local modifications	Changes of the active parameters through teach-in or local parameterization at the Device (on-line)	Device technology application sets "DS_Upload" flag and then triggers upload via "DS_UPLOAD_REQ" Event. "DS_Upload" flag is deleted as soon as the upload is completed.
Off-site commissioning (see 9.4.4.2)	Phase 1: Device is parameterized off-site via "USB-Master" tool (see Figure 34). Phase 2: Connection of that Device to a Master port.	Phase 1: "USB-Master" tool sends ParamDownloadStore; Device sets "DS_Upload" flag (in non-volatile memory) and then triggers upload via "DS_UPLOAD_REQ" Event, which is ignored by the "USB-Master". Phase 2: During system start-up, the "DS_Upload" flag triggers upload (copy). "DS_Upload" flag is deleted as soon as the upload is completed.
Changed port configuration (in case of "Backup/Restore" or "Restore")	Whenever port configuration has been changed via Master tool (on-line): e.g. Configured VendorID (CVID), Configured DeviceID (CDID), see 11.2.2 in [1].	Change of port configuration to different VendorID and/or DeviceID as stored within the Master triggers "DS_Delete" followed by an upload (copy) to Data Storage (see 11.8.2, 11.2.1 and 11.3.3 in [1]).
PLC program demand	Parameter change via user program followed by a SystemCommand	User program sends SystemCommand ParamDownloadStore; Device sets "DS_Upload" flag and then triggers upload via "DS_UPLOAD_REQ" Event. "DS_Upload" flag is deleted as soon as the upload is completed.

9.4.5.5 Production ("Restore")

Any changes of the active parameters through teach-in, tool based parameterization, or local parameterization shall not lead automatically to a download ("restore") of the entire parameter set; the upload can be disabled.

Criteria for the particular copy activities are listed in Table 14. These criteria are the conditions to trigger a copy process of the active parameters to the backup parameters, thus ensuring the consistency of these two sets.

Table 14 – Criteria for backing up parameters ("Restore")

User action	Operations	Data Storage
Change port configuration	Change of port configuration via Master tool (on-line): e.g. Configured VendorID (CVID), Configured DeviceID (CDID); see 11.2.2.5 in [1].	Change of port configuration triggers "DS_Delete" followed by an upload (copy) to Data Storage; see 11.8.2, 11.2.1 and 11.3.3 in [1].

9.4.6 Use cases

9.4.6.1 Device replacement (@ "Backup/Restore")

The stored (saved) set of back-up parameters overwrites the active parameters (e.g. factory settings) within the replaced compatible Device of same type. This one operates after a re-start with the identical parameters as its predecessor.

The preconditions for this use case are

(1) Devices and Master port adjustments according to 9.4.3.2;

Backup Level: "Backup/Restore"

The replacement Device shall be re-initiated to "factory settings" in case it is not a new Device out of the box (for "factory reset" see 10.6.4 in [1])

9.4.6.2 Device replacement (@ "Restore")

The stored (saved) set of back-up parameters overwrites the active parameters (e.g. factory settings) within the replaced compatible Device of same type. This one operates after a re-start with the identical parameters as its predecessor.

The preconditions for this use case are

(1) Devices and Master port adjustments according to 9.4.3.2;

Backup Level: "Restore"

9.4.6.3 Master replacement

9.4.6.3.1 General

This feature depends heavily on the implementation and integration concept of the Master designer and manufacturer as well as on the features of the upper level system (fieldbus).

9.4.6.3.2 Without fieldbus support (base level)

Principal approach for a replaced (new) Master using a Master tool:

(1) Set port configurations: amongst others the *Backup Level* to "Backup/Restore" or "Restore"

Master "reset to factory settings": clear backup parameters of all ports within the Data Storage in case it is not a new Master out of the box

Active parameters of all Devices are automatically uploaded (copied) to Data Storage (backup)

9.4.6.3.3 Fieldbus support (comfort level)

Any kind of fieldbus specific mechanism to back up the Master parameter set including the Data Storage of all Devices is used. Even though these fieldbus mechanisms are similar to the IO-Link approach, they are following their certain paradigm which may conflict with the described paradigm of the IO-Link back up mechanism (see Figure 33).

9.4.6.3.4 PLC system

The Device and Master parameters are stored within the system specific database of the PLC and downloaded to the Master at system startup after replacement.

This top down concept may conflict with the active parameter setting within the Devices.

9.4.6.4 Project replication

Following the concept of 9.4.6.3.3, the storage of complete Master parameter sets within the parameter server of an upper level system can automatically initiate the configuration of Masters and Devices besides any other upper level components and thus support the automatic replication of machines.

Following the concept of 9.4.6.3.4, after supply of the Master by the PLC, the Master can supply the Devices.

10 Extensions of the FS-Master

10.1 Principle architecture

Figure 35 shows the principle architecture of the FS-Master.

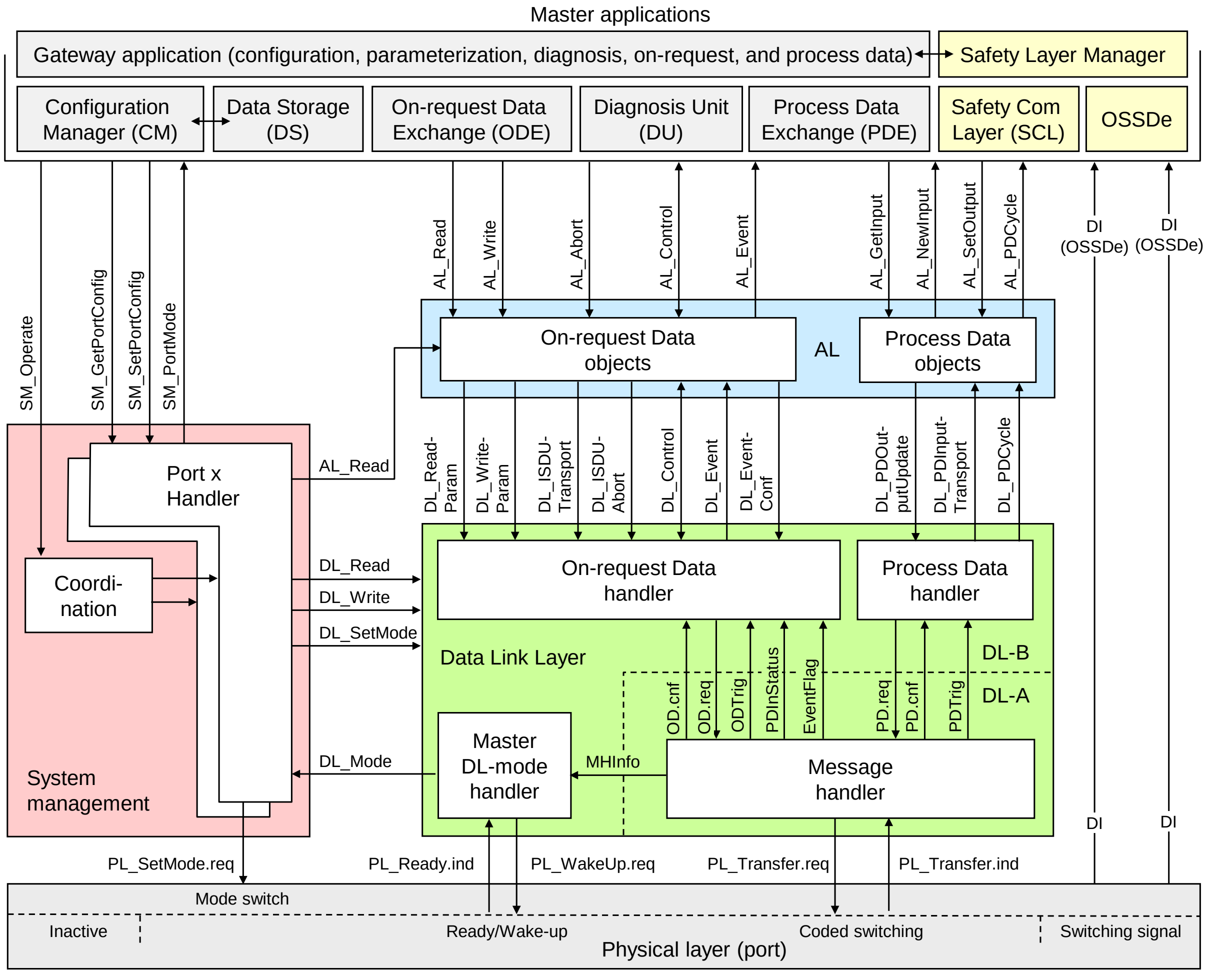


Figure 35 – Principle architecture of the FS-Master

Core part of an FS-Master is the original standard Master except for the Ready-pulse and its handling (see 5.3.3 and 7.2). The Master applications have been extended by a Safety Layer Manager dealing with safety communication (see clause 11) and OSSDe.

10.2 Safety Layer Manager (SLM)

10.2.1 Purpose

The Safety Layer Manager takes care of the safety PDU, whenever safety communication has been configured or of one safety bit, whenever OSSDe has been configured for a particular port.

It holds the FSP parameter block consisting of the authenticity record and the protocol record (see 11.7.5) as well as the FS I/O structure description (see Table A.1 and E.5.5).

10.2.2 FS_PortModes

The FS-Master shall support five FS_PortModes adjustable via the FS-Master Tool.

NonSafetyCom
This setting enables pure IO-Link communication with only NSR Process Data of a port.

SafetyCom
This setting enables pure safety communication without NSR Process Data of a port.

MixedSafetyCom
This setting enables safety communication of SR and NSR Process Data of a port.

OSSDe
This setting enables OSSDe operation of a port.

SIO
This setting enables SIO operation of a port.

10.2.3 FSP parameter blocks

10.2.3.1 FSP parameter use cases

Figure 36 illustrates some use cases related to the FSP parameters (see A.1).

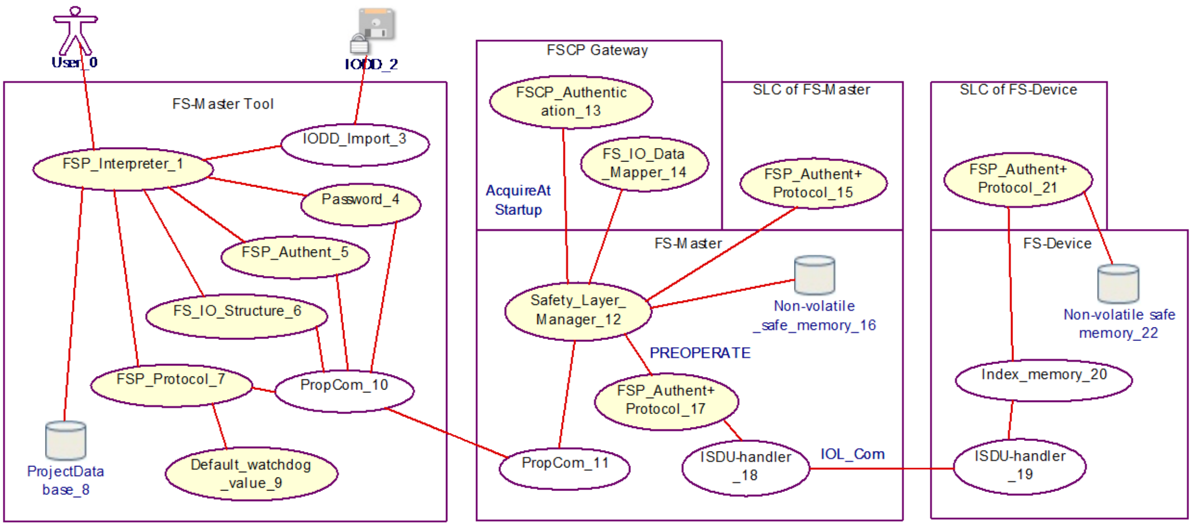


Figure 36 – FSP parameter use cases

Table 15 shows a listing of the items in Figure 36 and references to clauses within this document or to other IO-Link specifications (bibliography).

Table 15 – Use case reference table

No.	Item	Type	Reference	Remarks
0	User	Roles: - Observer - Maintenance - Specialist	–	Responsibility of the software tool manufacturer
1	FSP_Interpreter	GUI-functions	E.g. Figure 56	
2	IODD (secured)	Device description	Annex E	
3	IODD_Import	Activity	Annex E	
4	Password	Activity	Clause 10.2.3.2	Role dependent
5	FSP_Authent	Activity	Clause 11.7.5	
6	FS_IO_Structure	FS I/O description	Annex A.1	
7	FSP_Protocol	Activity	Clause 11.7.6	
8	ProjectDatabase	FS-Master Tool	–	Proprietary
9	Default_watchdog_value	Activity	Annex A.2.6	
10	PropCom (not standardized)	Communication	Clause 10.2.3.1	Proprietary
11	PropCom (not standardized)	Communication	Clause 10.2.3.1	Proprietary
12	Safety_Layer_Manager	Activity	Clause 10.2	
13	FSCP_Authentication	Activity	Clause 11.7.5	

No.	Item	Type	Reference	Remarks
14	FS_IO_Data_Mapper	Gateway application	Clause 12.1	FSCP Integration
15	FSP_Authent+Protocol	FS-Master SCL	Clause 11.5.2	
16	Non-volatile memory	FS-Master	–	Implementation
17	FSP_Authent+Protocol	Transfer	Clause 11.5.3	
18	ISDU-Handler	FS-Master DL	[1]	IO-Link standard
19	ISDU-Handler	FS-Device DL	[1]	IO-Link standard
20	Index_memory	Activity	[1]	IO-Link standard
21	FSP_Authent+Protocol	Activity	Clause 11.5.3	
22	Non-volatile memory	FS-Device	–	Implementation

In the following, a typical parameterization session of a project in the ProjectDatabase is described, where a new FS-Device is planned, configured, and parameterized for a particular port. After installation of IODD and associated Dedicated Tool, the user of an FS-Master Tool opens the parameter tab page (see illustration in Figure 56). After entry of the password for safety projects (see 10.2.3.2), FSP parameters are enabled to be displayed and Dedicated Tools are enabled to be launched.

The *authenticity parameter* values carry "0" as default within the IODD of an FS-Device. For details see 10.2.3.3.

The IODD contains the *I/O data structure description* of the safety Process Data as a record secured by CRC signature (see A.2.9 and E.5.6).

Most of the *protocol parameter* values are preset by default values provided by the FS-Device manufacturer within the IODD, except for the value of FSP_TechParCRC, which has a particular responsibility. A value of "0" means commissioning/test (see Annex G). The consequences are

- No validity check of technology parameters at start-up
- No blocking of FSP authenticity parameter acceptance within the FS-Device
- No Data Storage

Any value not equal to "0" will arm all three activities. For details see 10.2.3.4.

After parameter assignment, the FSP parameter instance values can be stored in the ProjectDatabase.

When online, the FS-Master Tool uses a proprietary communication ("PropCom") to the FS-Master (not standardized in [1]). Any transmission error (see Table 16) can falsify the message bits and thus, each FSP parameter record is secured by CRC signature.

NOTE Standardization of "PropCom" is responsibility of IO-Link integration into a fieldbus.

Upon power-on, the Safety Layer Manager of the FS-Master acquires the FSCP authenticity code and stores the values. The FS-Master Tool reads these values and replaces the default "0" by the actual FSCP authenticity code and assigns a port number ≠0. A CRC signature calculation secures the entire FSP authenticity parameter record. All three records, FSP authenticity, FSP protocol, and I/O structure description can be transferred to the Safety Layer Manager of the FS-Master.

NOTE The activities described above assume an FSP_TechParCRC value of "0" (commissioning).

The Safety Layer Manager propagates the I/O structure description record to the FS_IO_DataMapper. The FSP authenticity and FSP protocol records are propagated to the local FS-Master safety communication layer (SLC) and in PREOPERATE state to the FS-Device safety communication layer (SLC). The FS-Device accepts the authenticity code and stores it locally.

From now on the IO-Link Safety system is able to run in "monitored operational mode". That means personnel are required to watch the machine.

The user is now able to enter and test the technology specific parameters (see illustration in Figure 56). After verification and validation, the user launches the Dedicated Tool, confirms the value assignments and transfers the CRC signature to the FSP_TechParCRC field. With a value of ≠ "0", the system can be armed:

- Data Storage
- Blocking of FSP authenticity parameter acceptance within the FS-Device (comparison only)
- Validity check of authenticity and technology parameters at start-up

10.2.3.2 Password

The password mechanism is only required for the FS-Master. It shall consider the roles of the upper level FSCP system and inherit permissions from there if possible. Due to increased security requirements (IEC 62443), the mechanism shall be based on encryption methods. For details see Annex A.2.10.

Dedicated Tools can have additional password mechanisms independent from the FS-Master.

10.2.3.3 FSP parameter block – authenticity

FSP authenticity parameters are specified in Annex A.2.1. The authenticity activities for an FSCP-System are described in 10.2.3.1 including the CRC signature calculation.

For stand-alone FS-Masters the entry of unique and unambiguous values per FS-Master is required per machine or production center, if there is a possibility to misconnect FS-Device amongst different FS-Masters. FS-Devices will accept and store FSP authenticity values only when FSP_TechParCRC = "0".

10.2.3.4 FSP parameter block – protocol

FSP protocol parameters are specified in Annex A.1. Manufacturer/vendor pre-sets values and defines ranges within the IODD for protocol version and mode, port mode, watchdog, and TechParCRC.

Manufacturer/vendor shall determine the pre-set value for the watchdog timer considering the FS-Device response time at the indicated transmission rate. The FS-Master Tool can calculate and suggest a value based on the performance data of the used FS-Master and on the pre-set value from the IODD.

The FS-Master Tool calculates the CRC signature across the FSP protocol parameter record.

10.2.3.5 FS I/O structure description

With the help of this information, the mapping process within the FSCP gateway can be controlled or monitored (see 11.7.7 and A.2.9).

The FS-Device can check the validity of the safety PDin/PDout structure via the FSP_IO_StructCRC signature within the FSP parameter record.

10.3 Process Data Exchange (PDE)

Safety Layer Manager is responsible to set-up the safety-related Process Data depending on FS_PortModes (see 10.2.2). It can be either a Safety PDU or single bits (one from OSSDe and another one for the qualifier). Process Data Exchange takes over or passes SR Process Data (see 11.4.3 Safety PDU) from/to the Safety Layer Manager.

10.4 Data Storage (DS)

In [1], Data Storage has been specified separately for Master and Device. In practice it turned out to be straighter forward to specify the mechanism as a whole in one place. It can be found in 9.4 in this document.

11 Safety communication layer (SCL)

11.1 Functional requirements

The functional requirements for safety communication are laid down in [11]. Main application area is "safety for machinery". Usually this means operational stop of a machine until clearance or repair and restart only after an operator acknowledgement. Primarily relevant are IEC 62061 and ISO 13849.

Other major requirements are suitability for up to SIL3/PLe safety functions, port specific passivation, and parameterization using dedicated tools. Safety measures and residual error rates for authenticity, timeliness, and data integrity of safety messages (safety PDUs) shall be compliant with IEC 61784-3, Edition 3.

11.2 Communication faults and safety measures

The point-to-point communication basis of IO-Link allows for a very lean protocol type and a hardware independent safety communication layer stack with a small memory footprint. Table 16 shows the communication errors to be considered and the chosen safety measures

- (Sequence) counter / inverted counter;
- Watchdog timer and receipt messages;
- Connection validation at commissioning, start-up, and repair; and
- Cyclic redundancy check for data integrity.

Table 16 – Communication errors and safety measures

Communication error	Protocol safety measures			
	Counter/Inverted counter	**Timeout with receipt**	**Connection validation [a]**	**Cyclic redundancy check (CRC)**
Corruption	–	–	–	X
Unintended repetition	X	X	–	–
Incorrect sequence	X	–	–	–
Loss	X	X	–	–
Unacceptable delay	–	X	–	–
Insertion	X	–	–	–
Masquerade	–	–	–	X
Addressing	–	–	X	–
Loop-back of messages	X	–	–	–

a Similar procedure as with functional safety digital input modules possible due to point-to-point communication

It is assumed, that there are no storing elements within the IO-Link communication path between FS-Master and FS-Device. Thus, a two bit counter is sufficient as a safety measure. A value 0b000 of this counter indicates a start or reset position of this counter. In cyclic mode it counts up to 0b111 and returns to 0b001.

The message send and receive concept of IO-Link allows for a simple watchdog timer and message receipt safety measure concept corresponding to the "de-energize to trip" principle.

It is assumed that an FS-Master is the owner of a functional safety connection ID of the upper level FSCP communication system similar to an FS-DI-Module within a remote I/O. A customer is required to perform a validation procedure, whenever a change occurred with the connected safety devices. IO-Link Safety relies on such a concept. Additionally, due to the standard "data storage" mechanism of IO-Link and the functional safety nature of the FS-Master, it is possible to provide a more convenient mechanism.

A CRC signature is used for the data integrity check of transmitted safety PDUs. Two options can be configured. A 16 bit CRC signature for safety I/O data up to 4 octets or a 32 bit CRC signature for safety IO data up to 26 octets can be chosen.

11.3 SCL services

11.3.1 Positioning of safety communication layers (SCL)

Figure 37 shows the positioning of the IO-Link Safety Communication Layer (SCL).

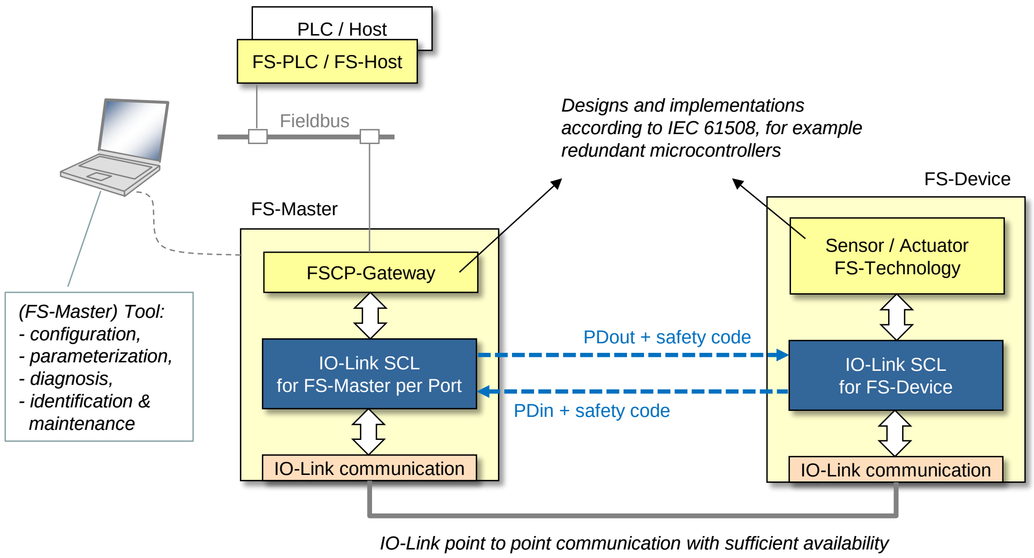


Figure 37 – Positioning of the IO-Link Safety Communication Layer (SCL)

For each port with a connected FS-Device an instance of the IO-Link SCL is required. The SCLs are exchanging safety PDUs consisting of output Process Data (PDout) together with safety code to the FS-Device and input Process Data (PDin) together with safety code from the FS-Device. The SCLs are using standard IO-Link communication as a "black channel".

Sufficient availability through for example correct installations, low-noise power supplies, and low interferences are preconditions for this "black channel" to avoid so-called nuisance trips. Nuisance trips cause production stops and subsequently may cause management to remove safety equipment.

This document does not specify implementation related safety measures such as redundant microcontrollers, RAM testing, etc. It is the responsibility of the manufacturer/vendor to take appropriate measures against component failures or errors according to IEC 61508.

11.3.2 FS-Master SCL services

IO-Link safety applications include (but are not limited to) connections to upper level FSCP fieldbus systems. FSCPs usually provide also safety codes and control/monitoring services (signals).

Figure 38 shows the FS-Master Safety Communication Layer signals (services) depicted by arrows in the upper part of the figure. For each FSCP to be connected to, a mapping or emulation of corresponding SCL services is required.

A service name carries either an extension "_C" (Control), if it controls the safety communication activities or an extension "_S" (Status), if it is reporting on the activities.

Some of the service names correspond to the signal names of the Control Byte or Status Byte (see lower part of the figure and 11.4.5). That means they are correlated, but there is some control logic of the SCL in between. This control logic is time discrete and not continuous even if it is depicted as logic OR ("≥") box. Definitive are the state charts and the state transition tables of the SCL (see 11.5.2).

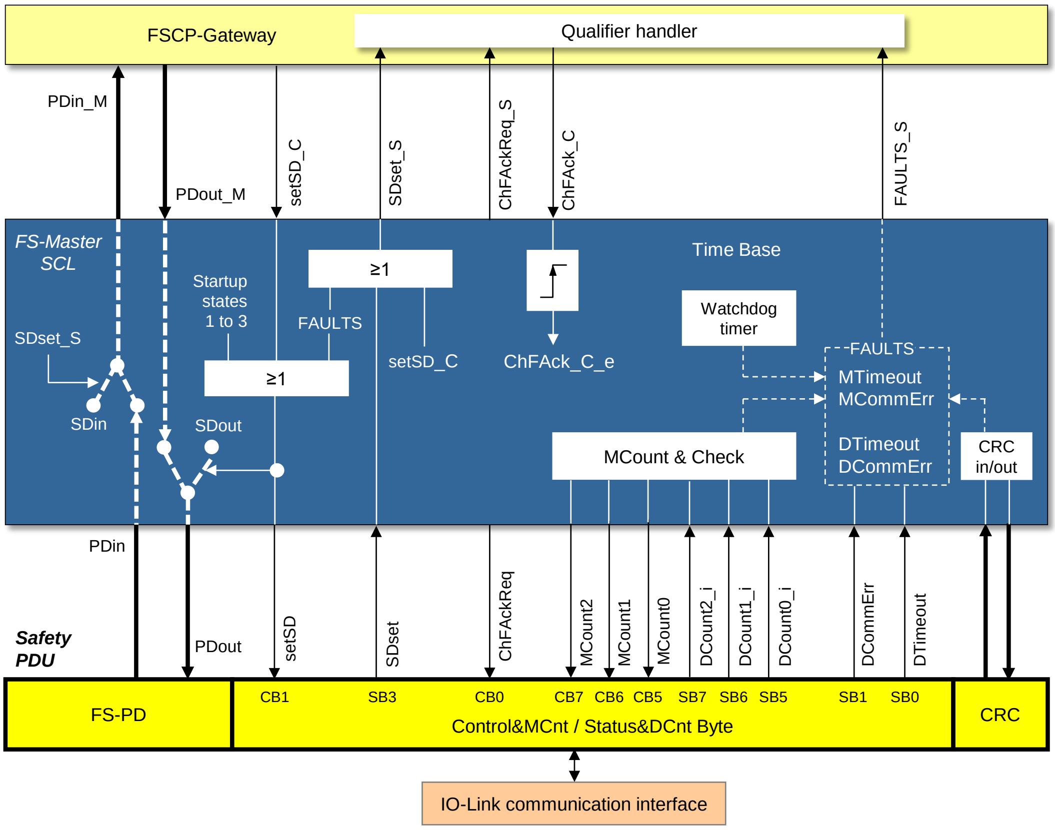


Figure 38 – FS-Master Safety Communication Layer services

The following services in Table 17 shall be available to the FSCP gateway or to a programmer of an FS-Master system.

Table 17 – SCL services of FS-Master

Service/signal	Definition
PDin_M, PDout_M	These services carry the actual Process Data values, both SDin (all bits "0") and SDout (all bits "0") in case of safe state or the real process values from or to the FS-Device.
SDin, SDout	These services carry Process Data values all zero.
setSD_C	In case of emergency, safety control programs usually set output Process Data (PDout_M) for an actuator to "0". However, in some cases, for example burner ventilators, shut down may not be a safe state. This service, if set to "1", is additional information allowing an FS-Device to establish a safe state no matter what the values of Process Data are. Independent from PDout_M, this service causes the SCL to send SDout values to the FS-Device and to send SDin to the FSCP gateway (PDin_M) via SDset_S.
SDset_S	This service, if set to "1", causes the qualifier handler to set the qualifier bit for the Process Data of the connected FS-Device (see 11.10.4). In addition, it causes the SCL to send SDin to the FSCP gateway (PDin_M).
ChFAckReq_S	The FS-Master SCL sets this service to "1" in case of FAULTS or FS-Master timeouts. It shall be propagated via FSCP and indicated to the operator.
ChFAck_C	After check-up and/or repair, the operator is requested to acknowledge a "ChFAckReq_S" service via a "1". This is a precondition for the SCL to resume regular operation after 3 transmission cycles with SDin and SDout values. The SCL-internal signal ChFAck_C_e is used for actual evaluation instead of the ChFAck_C service. It is only set, whenever the ChFAck_C service changed value (edge-sensitive) to avoid any continuously pressed acknowledgment button.
Fault_S	Any communication error (counter mismatch or CRC signature error) and/or timeouts cause the qualifier handler to set the qualifier bit for the Process Data of the connected FS-Device (see 11.10.4).

The lower part of the figure shows a combined input and output safety PDU specified in 11.4.3 and 11.4.5.

11.3.3 FS-Device SCL services

Figure 39 shows the FS-Device Safety Communication Layer services depicted by arrows in the upper part of the figure.

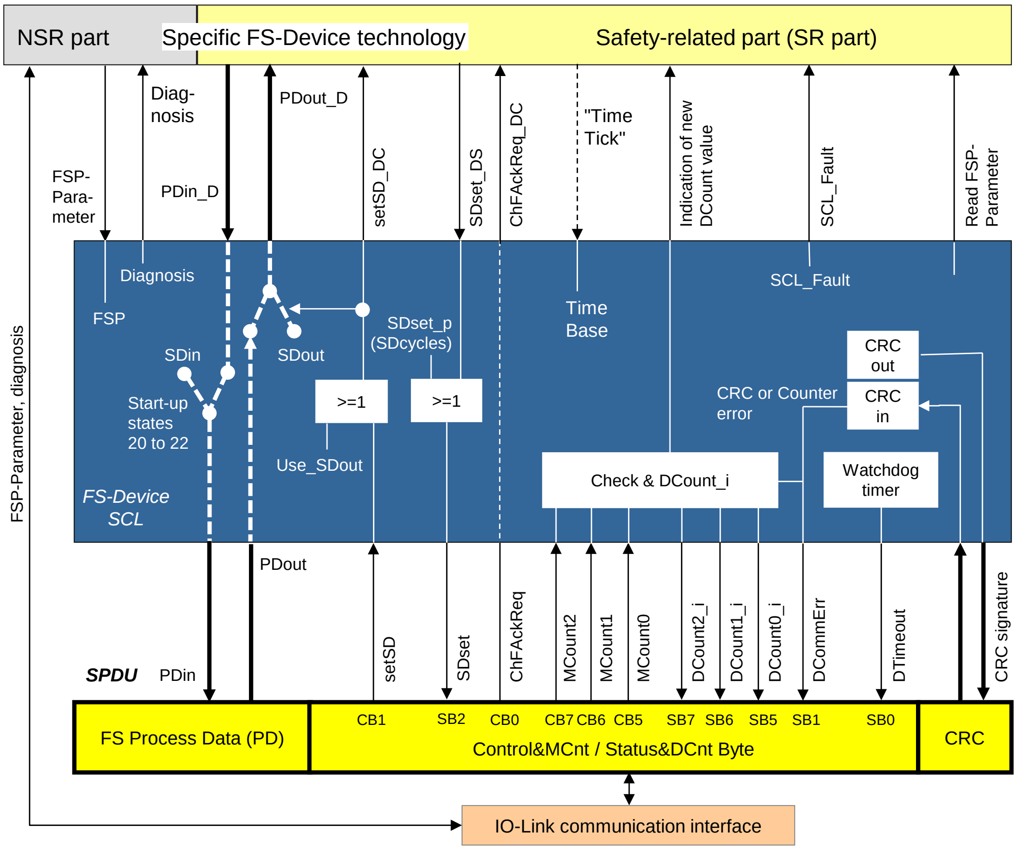


Figure 39 – FS-Device Safety Communication Layer services

A service name carries either an extension "_DC" (Device Control) if it controls the FS-Device technology or an extension "_DS" (Device Status) if it is reporting its status.

Some of the service names correspond to the signal names of the Control Byte or Status Byte (see lower part of the figure and 11.4.5). That means they are correlated, but there is some control logic of the SCL in between. This control logic is time discrete and not continuous even if it is depicted as logic OR ("≥") box. Definitive are the state charts and the state transition tables of the SCL (see 11.5.3).

The following services in Table 18 shall be available to the safety-related part of the FS-Device technology. Some services are non-safety-related and shall be available to the non-safety-related part of the FS-Device.

Table 18 – SCL services of FS-Device

Service/signal	Definition
PDin_D, PDout_D	These services carry the actual Process Data values. Real process values from the FS-Device and SDout (all bits "0") in case of safe state or the real process values to the FS-Device.
SDin, SDout	These services carry Process Data values all zero. Signal Use_SD indicates the usage of Process Data all zero.
setSD_DC	In case of emergency, safety control programs usually set output Process Data (PDout) for an actuator to "0". However, in some cases, for example burner ventilators, shut down may not be a safe state. This service, if set to "1", is additional information allowing an FS-Device to establish a safe state no matter what the values of Process

Service/signal	Definition
	Data are. Independent from PDout, this service causes the SCL to send SDout values to the FS-Device.
SDset_DS	This service, if set to "1", indicates that the FS-Device either reacts on a setSD_DC = "1" when the safe state is established or has been forced to establish safe state due to error or failure and delivers input Process Data values "0" (PDin_D).
ChFAckReq_DC	This service, if set to "1", indicates a pending operator acknowledgment. This signal is not safety-related and can be used to control an indicator, for example LED (light emitting diode).
Time tick	The SCL can be designed totally hardware independent, if a periodic service call controls a time base inside the SCL.
Indication of new DCount value	Short demands of FS-Devices may not trip a safety function due to its chain of independent communication cycles across the network. Therefore, a demand shall last for at least two SCL cycles. This service provides the necessary information to implement the demand extension if required.
SCL_Fault	This service provides faults (errors) of the SCL software.
Read_FSP_Parameter	This service allows the FS-Device technology for reading the current FSP (protocol) parameter
Non-safety-related services:	
FSP_Parameter	The FS-Master transmits the FSP parameter record (block) at each start-up during PREOPERATE to the FS-Device. These parameters are propagated to the SCL using this service.
Diagnosis	SCL diagnosis information can be propagated to the IO-Link Event system using this service.

The lower part of Figure 39 shows a combined input and output safety PDU specified in 11.4.3 and 11.4.5.

11.4 SCL protocol

11.4.1 Protocol phases to consider

Figure 40 shows the principle protocol phases to consider for the design according IEC 61784-3.

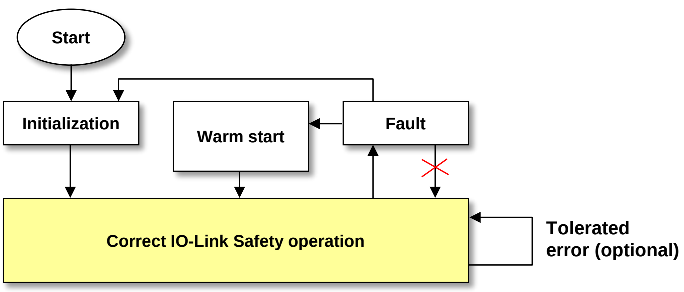


Figure 40 – Protocol phases to consider

The principle protocol phases and the corresponding requirements are listed in Table 19.

Table 19 – Protocol phases to consider

Phase	Activities	Requirements
Initialization	Establish communication, transfer FSP parameter to FS-Device, SD cycles	- Actuator shall be de-energized - SDout values shall be used during the first 3 SCL communication cycles
Setup or change	Commissioning, FST parameter backup	- As long as the FSP_TechParCRC is set to "0", cyclic data exchange of PD values is enabled.

Phase	Activities	Requirements
Operation	Process Data exchange, power-down of FS-Device	- It is the responsibility of the FS-Device technology to detect undervoltages and to set SD values.
Restart after transition from fault	Timeout, operator acknowledgment	- Operator acknowledgment is required prior to a restart - MCounter reset (resynchronization) - SDout values shall be used during the first 3 SCL communication cycles
Warm start after transition from fault	CRC or counter error, operator acknowledgment	- Operator acknowledgment is required prior to a restart - SCL communication is not reset - SDout values shall be used during the first 3 SCL communication cycles
Shutdown	Contact bouncing, EMC voltage dips/changes	- It is the responsibility of the FS-Device technology to detect undervoltages and to set SD values.

11.4.2 FS-Device faults

The SCL protocol copes with faults occurring during transmission of safety PDUs such as CRC errors or timeouts. It is the responsibility of the designer of the FS-Device to cope with FS-Device faults and to make sure that the necessary functional safety actions will take place, for example setting of safety Process Data and the SDset_DS service.

11.4.3 Safety PDU (SPDU)

Figure 41 shows the structure of SPDUs of the FS-Master and FS-Device together with standard input and output data. The design follows the concept of explicit transmission of the safety measures for timeliness and authenticity according to IEC 61784-3 in contrast to the implicit transmission via inclusion in the overall CRC signature calculation.

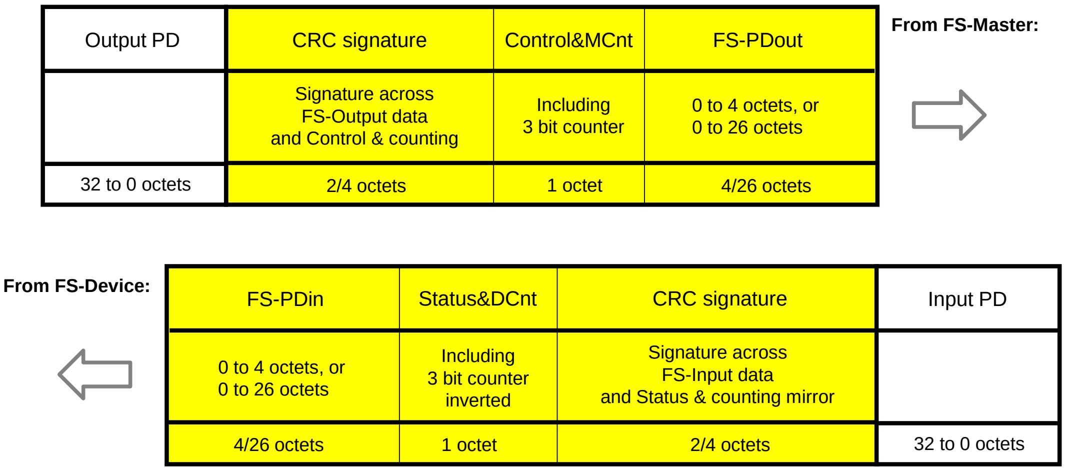


Figure 41 – Safety PDUs of FS-Master and FS-Device

The timeliness measure is represented by a 3 bit counter within the protocol management octets (see 11.4.5).

Inclusion of authenticity code in the cyclic checking is not necessary due to the point to point communication of IO-Link. This check is performed during commissioning and at start-up.

The design follows also the "de-energize to trip principle". In case of a timeout, or a CRC error, or a counter error, the associated qualifier bit will be set. It will be only released after an explicit operator acknowledgment on the FS-Master side.

After a CRC error a warm start is possible.

11.4.4 FS-Input and FS-Output data

The maximum possible size of the FS-Input and FS-Output data reaches from 0 to 26 octets depending on the amount of required standard IO-Link data. See 11.4.6 for optimization issues and trade-offs.

NOTE Currently the safety trailer consists of only 3 or 5 octets and theoretically 28 octets could be available. However, since not all design verification steps are passed, a reserve of 1 octet is planned.

The possible data types are listed in Table 23.

11.4.5 Status and control

One octet is used in both transmission directions for the protocol flow of IO-Link Safety.

Table 20 shows the signals to control the protocol layer of an FS-Device and a counter value for the timeliness check together with a local watchdog timer adjusted through the "FSP_Watchdog" parameter (see A.2.6).

Table 20 – Control and counting (Control&MCnt)

CB7	CB6	CB5	CB4	CB3	CB2	CB1	CB0
Sequence counter, bit 2	Sequence counter, bit 1	Sequence counter, bit 0	Reserved ("0")	Reserved ("0")	Reserved ("0")	Activate safe state	Channel fault acknowledge request (indication)
MCount2	MCount1	MCount0	–	–	–	SetSD	ChFAckReq

Table 21 shows the feedback of the protocol layer of an FS-Device and the inverted counter value for the timeliness check. The counter values are inverted to prevent from loop-back errors.

Table 21 – Status and counting mirror (Status&DCnt)

SB7	SB6	SB5	SB4	SB3	SB2	SB1	SB0
Sequence counter, bit 2; inverted	Sequence counter, bit 1; inverted	Sequence counter, bit 0; inverted	Reserved ("0")	Reserved ("0")	Safe state activated	Communication error: CRC or counter incorrect	Communi-cation fault: Timeout
DCount2_i	DCount1_i	DCount0_i	–	–	SDset	DCommErr	DTimeout

Table 22 shows the values of MCount and DCount_i during protocol operation.

Table 22 – MCount and DCount_i values

Phase	MCount		DCount_i	
	Dec	Bin	Dec	Bin
Initial or after timeout	0	000	7	111
Cyclic	1	001	6	110
	2	010	5	101
	3	011	4	100
	4	100	3	011
	5	101	2	010
	6	110	1	001
	7	111	0	000

11.4.6 CRC signature

For the design of the CRC mechanism and the calculation of the residual error probability/rate several parameters and assumptions are required:

- Explicit transmission of safety measures as opposed to implicit transmission. In this case, formulas are available within IEC 61784-3, Edition 3.

- The sampling rate of safety PDUs is assumed to be a maximum of 1000 sampled safety PDUs per second.
- The monitoring times for errors in safety PDUs are listed in Table 30. Any detected CRC error within the safety communication layer shall trip the corresponding safety function (safe state). During the monitoring time only one nuisance trip is permitted. Maintenance is required.
- The generator polynomials in use shall be proven to be proper within the SPDU range.
- The seed value to be used for the CRC signature calculation is "1" (see D.3.6).
- In case the result of the CRC signature calculation leads to a "0", a "1" shall be sent and evaluated at the receiver side correspondingly.
- The assumed bit error probability for calculations is 10^{-2}.

Figure 42 shows the so-called 1 % share rule of the IEC 61784-3. For IO-Link Safety it means, the residual error rate of an IO-Link Safety logical connection shall not exceed 1 % of the average probability of a dangerous failure (PFH) of that safety function with the highest SIL the safety communication is designed for, which is SIL3. This value is 10^{-9}/h.

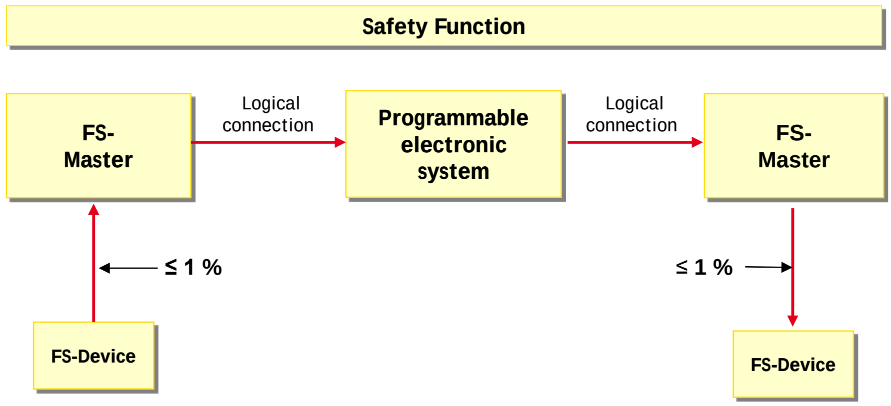


Figure 42 – The 1 % share rule of IEC 61784-3

Calculations under the above conditions have shown the following possibilities (see Annex D):

– For a CRC16 proper polynomial (*0x4EAB*) 4 octets of process data (safety PDU length = 7 octets);

– For a CRC32 proper polynomial (*0xF4ACFB13*) 26 octets of process data (safety PDU length = 32 octets).

Thus, support of two variants is provided: CRC-16 with up to 4 octets of safety I/O data and CRC-32 with up to 26 octets.

11.4.7 Data types for IO-Link Safety

11.4.7.1 General

The cyclically exchanged functional safety data structures between FS-Device and FS-Master comprise FS process I/O data and the IO-Link Safety protocol trailer. They are transmitted in Safety PDUs.

Acyclically exchanged functional safety data structures are transmitted in IO-Link On-request Data (OD) containers either from a dedicated tool or from a user program within an FS-PLC. In this case additional securing mechanisms (e.g. CRC signature) are required at each and every transfer or after a parameter block.

11.4.7.2 FS process I/O data (PDin and PDout)

For the FS process I/O data a well-defined set of data types and a corresponding description is defined for both FS-Device and FS-Master for correct processing and mapping to the upper-level FSCPs. Table 23 lists the three permitted data types (see Annex C).

Table 23 – FS process I/O data types

Data type	Coding	Length	See [1]	Device example
BooleanT/bit	BooleanT ("packed form" for efficiency, no WORD structures); assignment of signal names to bits is possible.	1 bit	Clause E.2.2; Table E.22 and Figure E.8	Proximity switch
IntegerT(16)	IntegerT (enumerated or signed)	2 octets	Clause E.2.4; Table E.4, E.7 and Figure E.2	Protection fields of laser scanner
IntegerT(32)	IntegerT (enumerated or signed)	4 octets	Clause E.2.4; Table E.4, E.6, and Figure E.2	Encoder or length measurement (≈ ± 2 km, resolution 1 µm)

11.4.7.3 Qualifier

FS-Devices normally do not require qualifiers (see 11.10.2). The qualifier bits are configured together with the Process Data (or Safe Data = SD) during the mapping to the upper level FSCP system. The data structures depend on the rules of these FSCP systems.

In case of FS-Terminals (see 11.10.3) the rules in Table 24 for the layout of binary and digital data and their qualifier bits apply.

Table 24 – Rules for the layout of values and qualifiers

No.	Rule	Remark
1	Only Boolean (DI, DO) and IntegerT(16) or IntegerT(32) (AI, AO) data types shall be used. Any value shall be assigned to one of these categories.	
2	Boolean values precede Integer values.	
3	IntegerT(16) precedes IntegerT(32) values	
4	Values precede qualifier in an octet-wise manner	
5	Qualifiers follow directly input values. In case of no input values only the qualifiers for output values are placed.	
6	Qualifier for input values precede qualifier for output values	
7	Qualifiers for each category (DI, DO, AI, AO) are packed separately in an octet-wise manner.	
8	If data types are missing the remaining data types catch up.	

Table 25 shows the ranking of values and qualifiers.

Table 25 – Order of values and qualifier

Order	To FS-Master	To FS-Device
1	Value DI	Value DO
2	Value AI	Value AO
3	Qualifier DI	–
4	Qualifier AI	–
5	Qualifier DO	–
6	Qualifier AO	–

11.4.7.4 IO-Link Safety protocol trailer

The data types for the protocol trailer ("safety code") are specified in Annex C.5.

11.4.7.5 FSP and FST parameter

No particular data type definitions are required.

11.5 SCL behavior

11.5.1 General

The state machines for the FS-Master and the FS-Device safety communication layer are designed using the chosen safety measures in Table 16 and the protocol signals in 11.4.5.

11.5.2 SCL state machine of the FS-Master

Figure 43 shows the FS-Master state machine for wired IO-Link point-to point communication.

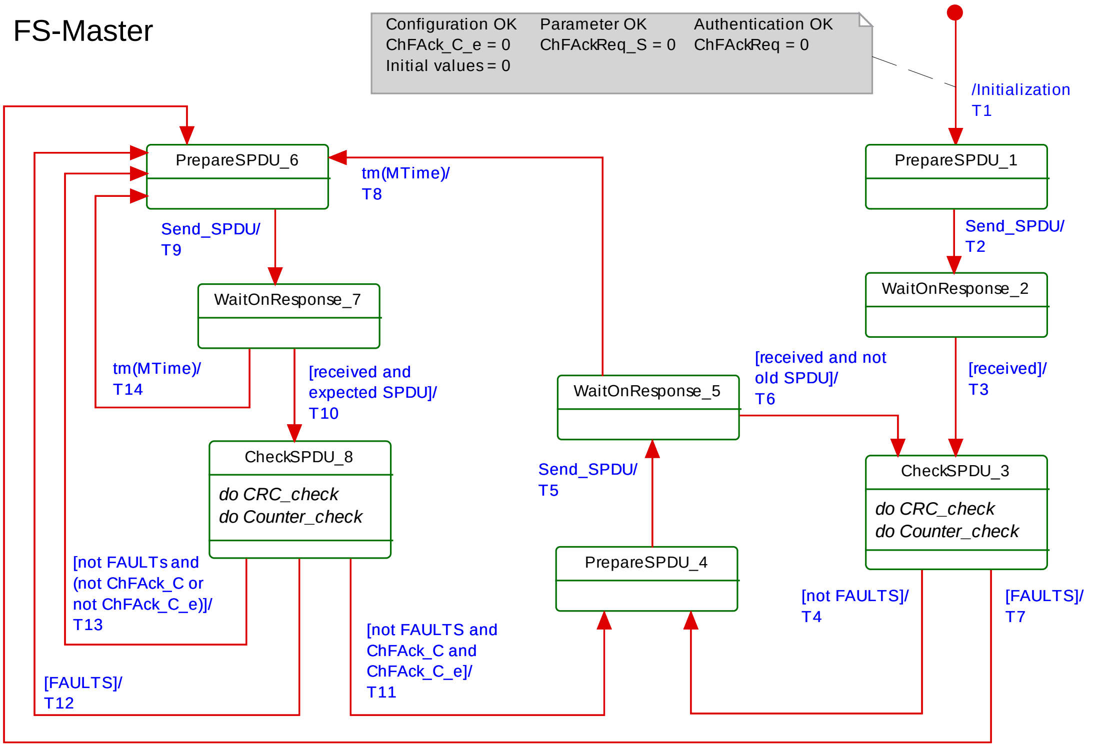


Figure 43 – SCL state machine of the FS-Master

The terms used in Figure 43 are defined in Table 26.

Table 26 – Definition of terms used in SCL state machine of the FS-Master

Term	Definition
ChFAck_C	Operator acknowledgment for the safety function via the FS-Gateway
FAULTS	MTimeout: FS-Master timeout when waiting on an FS-Device SPDU response, or MCommErr: FS-Master detects a corrupted FS-Device SPDU response (incl. counter error), or DTimeout: FS-Device reported a timeout of its SCL via Status&DCnt Byte, or DCommErr: FS-Device reported a CRC (incl. counter error) by its SCL via Status&DCnt Byte

Table 27 – FS-Master SCL states and transitions

STATE NAME	STATE DESCRIPTION
Initialization	Initial state of the FS-Master SCL instance upon power-on (one per port).
1 PrepareSPDU	Preparation of a *(regular)* SPDU for the FS-Device. Send SPDU when prepared.
2 WaitOnResponse	SCL is waiting on SPDU from FS-Device.
3 CheckSPDU	Check received SPDU for not FAULTS (→ T4). In case of FAULTS: errors within the Status&DCnt Byte (DCommErr, DTimeout, SDset) → T7
4 PrepareSPDU	Preparation of a *(regular)* SPDU for the FS-Device. Send SPDU when prepared.

STATE NAME	STATE DESCRIPTION
5 WaitOnResponse	SCL is waiting on next SPDU from FS-Device not carrying the previous DCount_i.
6 PrepareSPDU	Preparation of an (*exceptional*) SPDU for the FS-Device (due to MTimeout, missing OpAck, or FAULTS).
7 WaitOnResponse	SCL is waiting on next SPDU from FS-Device not carrying the previous DCount_i. When received → T10, after MTimeout → T14.
8 CheckSPDU	Check received SPDU for a CRC error (MCommErr) and for potential FS-Device faults within the Status&DCnt Byte (DTimeout, DCommErr). Once a fault occurred, no automatic restart of a safety function is permitted unless an operator acknowledgement signal (ChFAck_C) arrived (see Figure 38). Hint: A delay time may be required avoiding the impact of an occasional system shutdown.

TRAN-SITION	SOURCE STATE	TARGET STATE	ACTION
T1	0	1	use SD, setSD =1, SDset_S =1 MCount = 0
T2	1	2	–
T3	2	3	–
T4	3	4	MCount = MCount + 1 **if** MCount = 8 **then** MCount = 1 **if** SDset =1 or setSD_C =1 **then** use SDin, SDset_S =1 **else** use PDin, SDset_S =0 **if** setSD_C =1 **then** use SDout, setSD =1 **else** use PDout, setSD =0
T5	4	5	restart MTimer
T6	5	3	–
T7	3	6	use SD, setSD =1, SDset_S =1 MCount = MCount + 1 **if** MCount = 8 **then** MCount = 1
T8	5	6	use SD, setSD =1, SDset_S =1 MCount = 0
T9	6	7	restart MTimer
T10	7	8	–
T11	8	4	ChFAckReq =0, ChFAckReq_S =0, ChFAck_C_e =0, MCount = MCount + 1 **if** MCount = 8 **then** MCount = 1 **if** SDset =1 or setSD_C =1 **then** use SDin, SDset_S =1 **else** use PDin, SDset_S =0 **if** setSD_C =1 **then** use SDout, setSD =1 **else** use PDout, setSD =0
T12	8	6	ChFAckReq =0, ChFAckReq_S =0, ChFAck_C_e =0, use SD, setSD =1, SDset_S =1 MCount = MCount + 1 **if** MCount = 8 **then** MCount = 1
T13	8	6	ChFAckReq =1, ChFAckReq_S =1, /*set qualifier/acknowledgment request*/ **if** ChFAck_C = 0 **then** ChFAck_C_e =1 use SD, setSD =1, SDset_S =1 MCount = MCount + 1 **if** MCount = 8 **then** MCount = 1
T14	7	6	ChFAckReq =0, ChFAckReq_S =0, ChFAck_C_e =0, use SD, setSD =1, SDset_S =1 MCount = 0

INTERNAL ITEMS	TYPE	DEFINITION
MTimer	Timer	This timer checks the arrival of the next valid SPDU from the FS-Device in time. The FS-Master Tool is responsible to define this watchdog time. Value range is 0 … 65 535 ms.
ChFAck_C_e	Flag	By means of this auxiliary variable (bit) it is ensured that the safe state will be left only after a signal change of ChFAck_C from 0 → 1 (edge). Without this mechanism an operator could overrule safe states by permanently actuating the ChFAck_C signal.
FAULTS	Flags	Permanent storage of the following errors or failures can be omitted within the FS-Master, if it can be assumed that the upper level FSCP system prevents from automatic restart of safety functions (no FS-Device persistence): - MCommErr - MTimeout - DCommErr, including counter error (Status&DCnt Bit 1) - DTimeout (Status&DCnt Bit 0)
Expected SPDU	Guard	Mirrored inverted counter (DCount_i = inverted MCount)
Not old SPDU	Guard	Counter value ≠ value of previous SPDU
do CRC_check	Activity	SCL calculates CRC signature across received SPDU while signature value = "0" and compares with received CRC signature
do Counter_check	Activity	SCL checks whether DCount carries an expected value (mirror)
NOTE Variables within ACTIONs are defined in 11.3		

11.5.3 SCL state machine of the FS-Device

Figure 44 shows the corresponding FS-Device state machine.

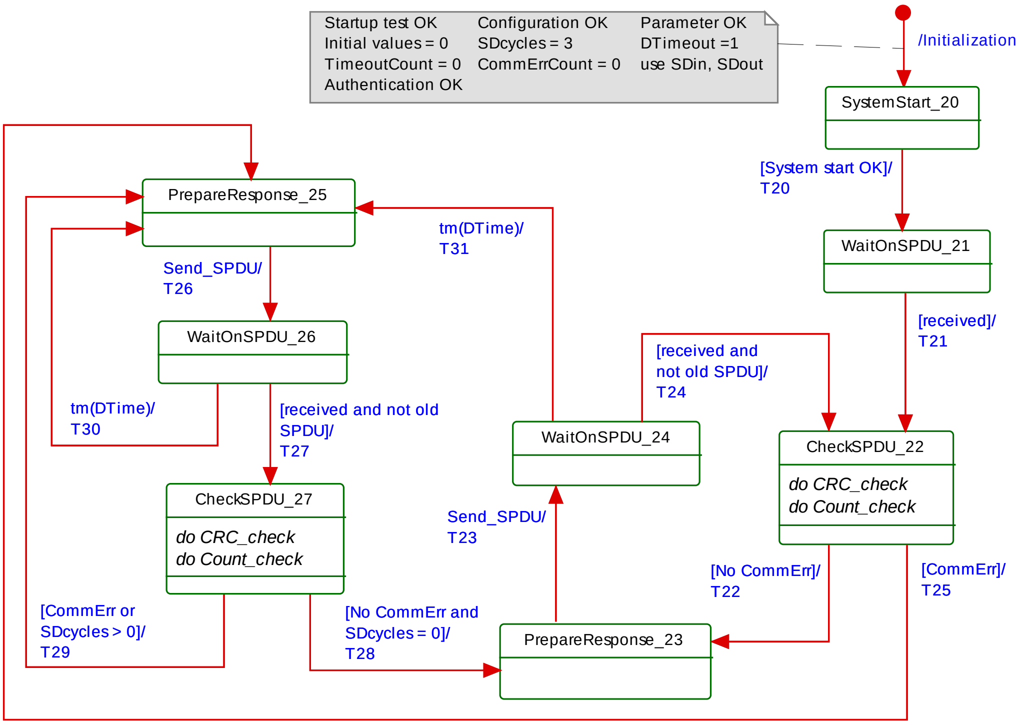


Figure 44 – SCL state machine of the FS-Device

The terms used in Figure 44 are defined in Table 28.

Table 28 – Definition of terms used in SCL state machine of the FS-Device

Term	Definition
CommErr	The SCL within the FS-Device detected a CRC or counter error in the received SPDU

Term	Definition
CommErrCount	See INTERNAL ITEM in Table 29
SDcycles	See INTERNAL ITEM in Table 29
DTimeout	FSP_WatchdogTime expired
TimeoutCount	See INTERNAL ITEM in Table 29

1437

1438 **Table 29 – FS-Device SCL states and transitions**

STATE NAME	STATE DESCRIPTION
Initialization	Initialization of the FS-Device upon power-on. Upon power-on, the FS-Device (actuator) sets the PDout to "0". Upon power-on the FS-Device (sensor) is sending "0".
20 SystemStart	Immediately after FSP parameterization the FS-Device sets PDout to SDout values. Immediately after FSP parameterization it is sending Process Data (PD).
21 WaitOnSPDU	SCL is waiting on next SPDU from FS-Master.
22 CheckSPDU	Check received SPDU from FS-Master for CRC errors; set ChFAckReq_DC = ChFAckReq. When guard "No CommErr" = true → T22. When guard "CommErr " = true → T25
23 PrepareResponse	Preparation of (*regular)* SPDU response for the FS-Master (response message)
24 WaitOnSPDU	SCL is waiting on next (*regular*) SPDU from FS-Master not carrying the previous MCount. After FSP_WatchdogTime expired → T27. When SPDU received and guard "MCounter_incremented" = true → T24 (*regular* cycle)
25 PrepareResponse	Preparation of (*exceptional*) SPDU response for the FS-Master (due to DTimeout or DCommErr = error report bits in Status&DCnt Byte)
26 WaitOnSPDU	SCL is waiting on next SPDU from FS-Master not carrying the previous MCount. After FSP_WatchdogTime expired → T30. When SPDU received and guard "MCounter_incremented" = true → T27
27 CheckSPDU	Check received SPDU from FS-Master for CRC errors; set ChFAckReq_DC = ChFAckReq. When guard "No CommErr and SDcycles >=1" = true → T28. When guard "CommErr or SDcycles <1" = true → T29

1439

TRAN-SITION	SOURCE STATE	TARGET STATE	ACTION
T20	20	21	–
T21	21	22	–
T22	22	23	use PDin_D, DCommErr = 0, /*Status&DCnt, Bit 1*/ DTimeout = 0, /*Status&DCnt, Bit 0*/ DCount_i = MCount_inv, restart DTimer **if** SDcycles <> 0 **then** use SDout, setSD_DC=1, SDset =1, /*during SDcycles: SDset_p =1*/ SDcycles = SDcycles - 1 **else** use PDout, setSD_DC=0, SDset = 0 **if** setSD =1 /*use_SD =1*/ **then** use SDout, setSD_DC=1,
T23	23	24	**if** SDset_DS = 1 /* FS-Device fault*/ **then** SDset = 1
T24	24	22	–
T25	22	25	use PDin_D, use SDout, SDset = 1, DCommErr =1, /*Status&DCnt, Bit 1*/ CommErrCount = 1, DCount_i = MCount_inv, SDcycles = 3, restart DTimer
T26	25	26	–

TRAN-SITION	SOURCE STATE	TARGET STATE	ACTION
T27	26	27	–
T28	27	23	use PDin_D, use SDout, setSD_DC=0, SDset = 0, DCount_i = MCount_inv, DCommErr =0, /*Status&DCnt, Bit 1*/ DTimeout =0, /*Status&DCnt, Bit 0*/ restart DTimer,
T29	27	25	use PDin_D, use SDout, setSD_DC=1, SDset = 1, DCount_i = MCount_inv, restart DTimer **if** CommErr **then** DCommErr = 1, /*Status&DCnt, Bit 1*/ CommErrCount = 1, SDcycles = 3, **else** SDcycles = SDcycles -1 **if** CommErrCount = 1 **then** DCommErr = 1, /*Status&DCnt, Bit 1*/ CommErrCount = 0 **else** DCommErr = 0 /*Status&DCnt, Bit 1*/ **if** TimeoutCount = 1 **then** DTimeout = 1, /*Status&DCnt, Bit 0*/ TimeoutCount = 0 **else** DTimeout = 0 /*Status&DCnt, Bit 0*/
T30	26	25	use PDin_D, use SDout, setSD_DC=1, SDset =1, DTimeout =1, /*Status&DCnt, Bit 0*/ TimeoutCount =1, SDcycles = 3, restart DTimer, DCount_i = MCount_inv
T31	24	25	use PDin_D, use SDout, setSD_DC=1, SDset =1, DTimeout =1, /*Status&DCnt, Bit 0*/ TimeoutCount =1, SDcycles = 3, restart DTimer, DCount_i = MCount_inv

INTERNAL ITEM	TYPE	DEFINITION
MCount_inv	Variable	Inverse value of current MCount value
SDcycles	Counter	This decremental counter is used to cause the FS-Device setting SDout and SDset for at least 3 cycles during start-up and after a fault. Value range is 3 to 0.
CommErrCount	Counter	This decremental counter is used to guarantee the bit "DCommErr" within the Status&DCnt Byte is being set at least for 1 cycle or for a maximum of 2 cycles. Value range is 1 to 0.
TimeoutCount	Counter	This decremental counter is used to guarantee the bit "DTimeout" within the Status&DCnt Byte is being set at least for 1 cycle or for a maximum of 2 cycles. Value range is 1 to 0.
do CRC_check	Activity	SCL calculates CRC signature across received SPDU while signature value = "0" and compares with received CRC signature
do Counter_check	Activity	SCL checks whether MCount carries either "0" or an expected subsequent value
NOTE Variables within ACTIONs are defined in 11.3		

It is very unlikely for an FS-Device to receive SPDUs with all octets "0". The SCL within the FS-Device shall ignore such an SPDU. Normally, at least the CRC signature will be "1" if Process data and Control Byte are "0" according to the rules in 11.4.6.

11.5.4 Sequence charts for several use cases

11.5.4.1 FS-Master and FS-Device both with power ON

Figure 45 shows the sequence chart of a regular start-up of both FS-Master and FS-Device.

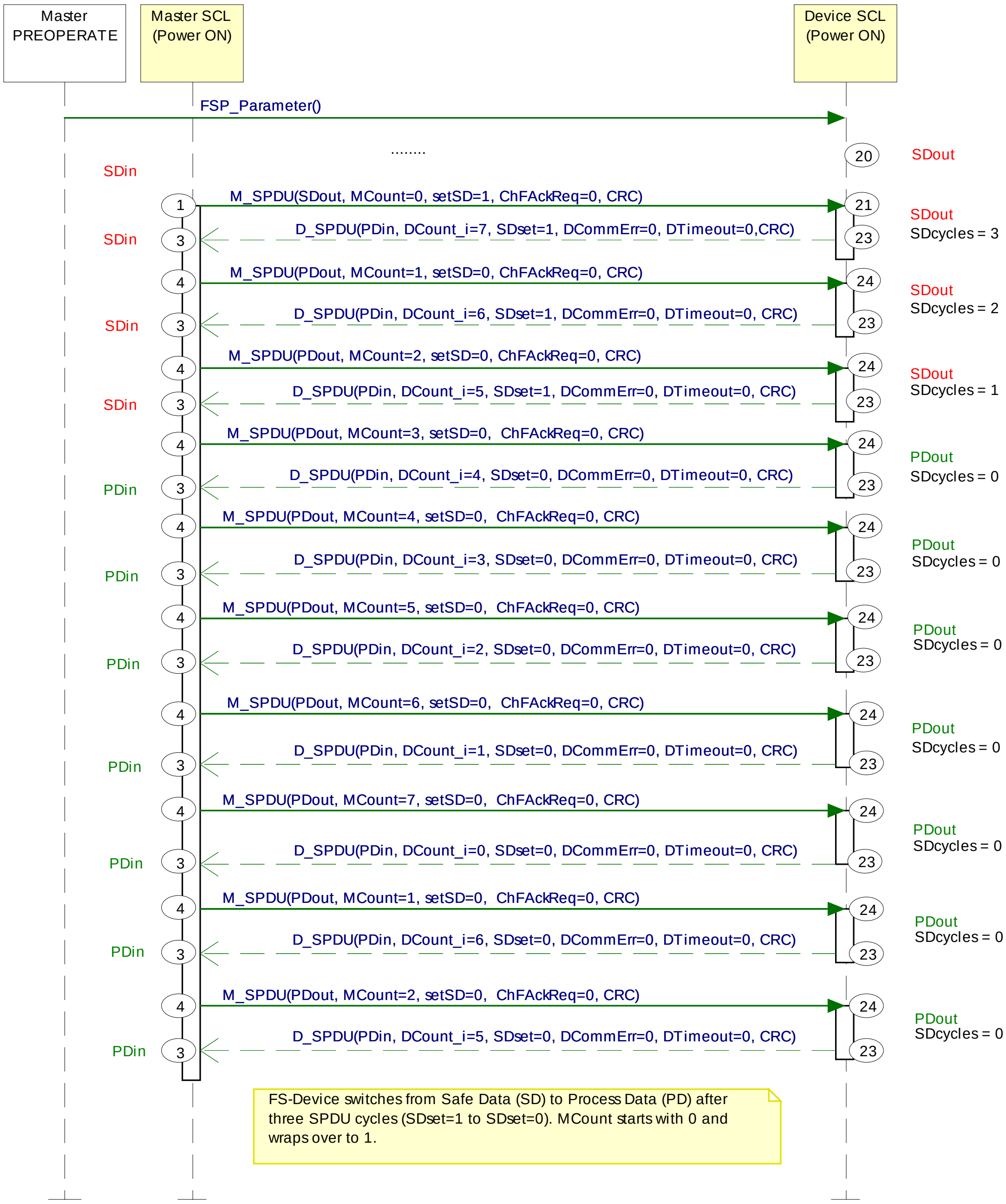


Figure 45 – FS-Master and FS-Device both with power ON

Upon power-on both FS-Master and FS-Device are providing SDin (PDin = "0") and SDout (PDout = "0") respectively. Both keep these values for 3 communication cycles (SDcycles) before switching to the regular mode, where only the MCounter and DCounter values are changing.

11.5.4.2 FS-Master with power OFF → ON

Figure 46 shows the sequence chart of regular operation while FS-Master has been switched OFF and ON again.

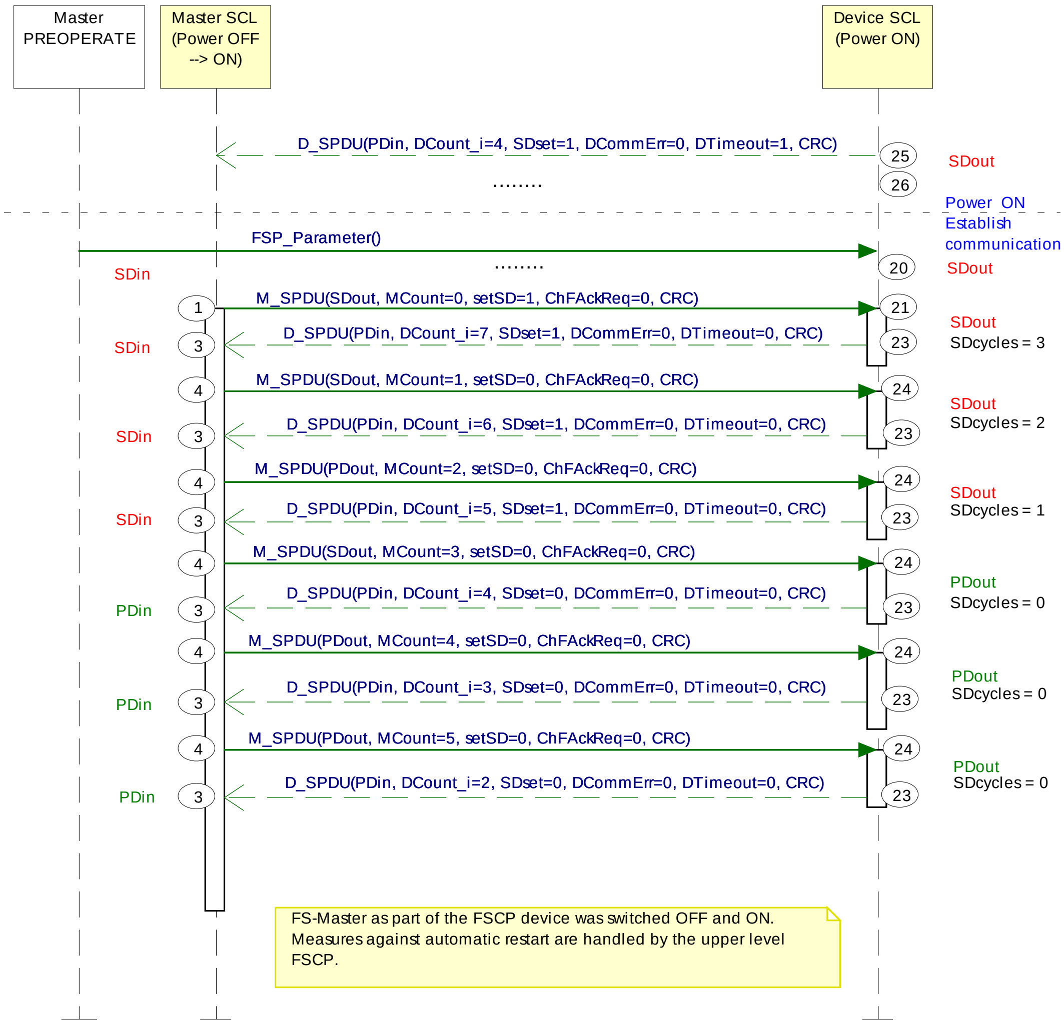


Figure 46 – FS-Master power OFF → ON

The FS-Device communication part is always powered by the FS-Master. Thus, if the FS-Master is switched OFF and ON, the FS-Device is just following and a regular start-up occurs. Since the FS-Master is part of an upper level FSCP system, this FSCP system is responsible to prevent from automatic restart of safety functions in this case.

11.5.4.3 FS-Device with delayed SCL start

Figure 47 shows the sequence chart when the SCL start within the FS-Device is delayed.

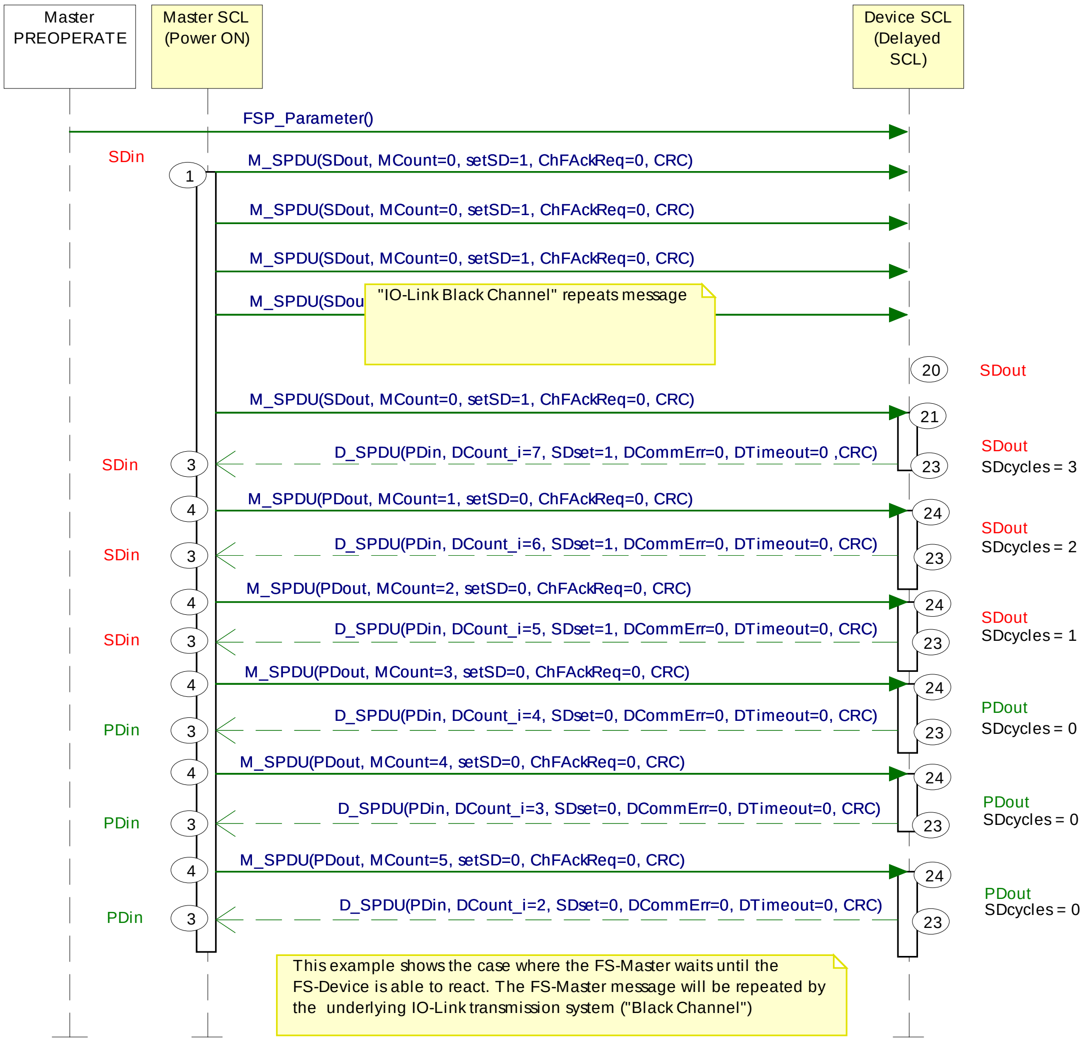


Figure 47 – FS-Device with delayed SCL start

This diagram shows how an FS-Master SCL waits on the SCL of the FS-Device in case of delays. The initial SPDU of the FS-Master is repeated by the IO-Link transmission system (black channel) until the SCL of the FS-Device is ready to process in state 21.

As long as the SCL of the FS-Device is not ready, the response SPDU contains all "0" and the FS-Master SCL will ignore such an SPDU. PDvalid/invalid of IO-Link is reserved for the non-safety part of the entire message.

11.5.4.4 FS-Device with power OFF and ON

Figure 48 shows the sequence chart when the FS-Device switches power OFF and ON again.

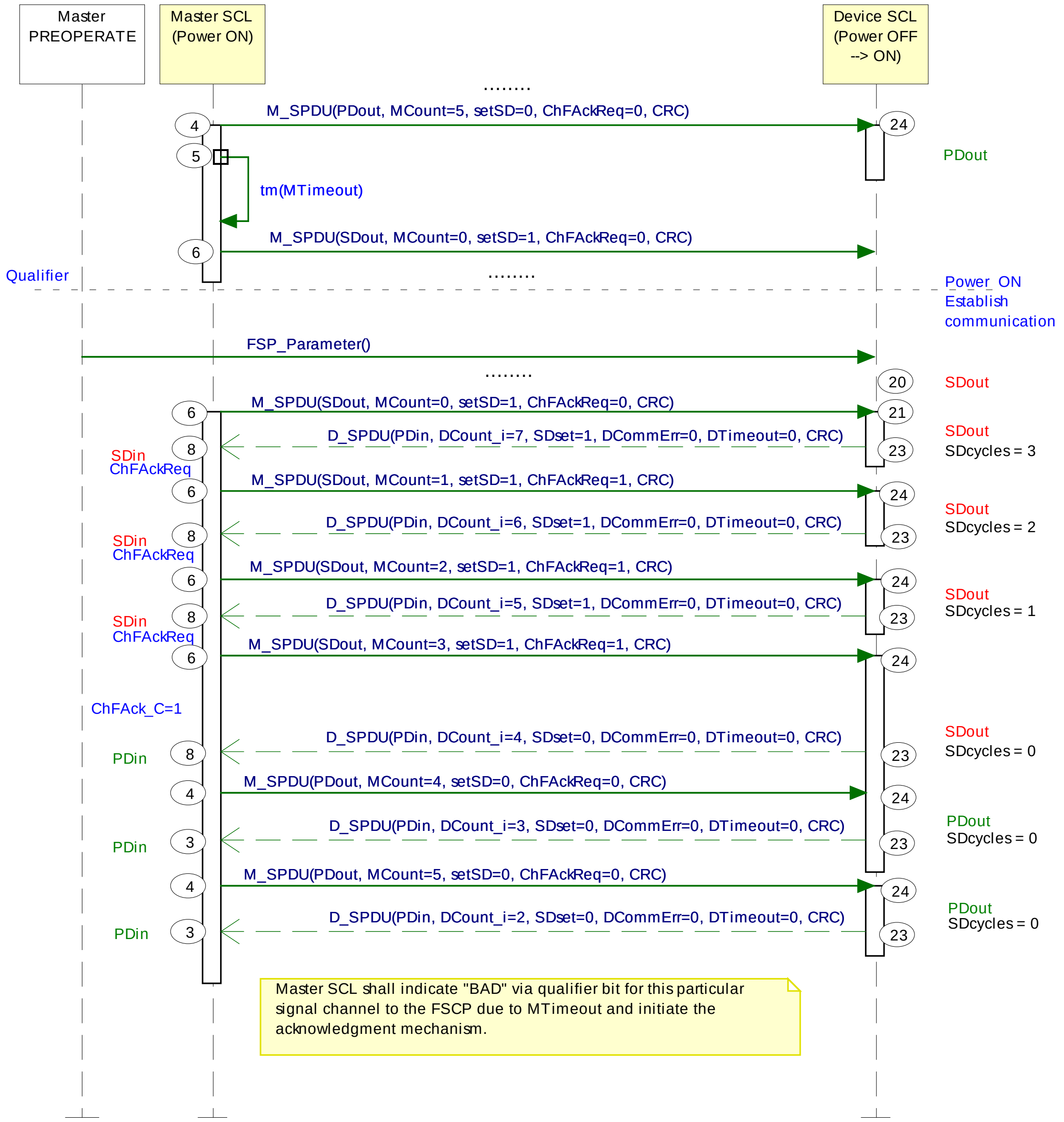


Figure 48 – FS-Device with power OFF and ON

This case assumes for example a short unplug and plug of the FS-Device causing a FAULT (MTimeout) on the FS-Master side. This FAULT causes a Qualifier bit to be set that requires via ChFAckReq (=1) an acknowledgment via ChFAck_C (=1). FS-Master and FS-Device keep SDin and SDout until this acknowledgment arrived.

11.5.4.6 FS-Master detects CRC signature error

Figure 49 shows the sequence chart when the FS-Master detects a CRC signature error.

Figure 49 – FS-Master detects CRC signature error

FS-Master received an SPDU with falsified data or falsified CRC signature which results in a "CRC error" (MCommErr). Both FS-Master and FS-Device switch to SDin and SDout respectively and the FS-Master/Gateway creates a qualifier bit and indicates a ChFAckReq signal. This signal is indicated also to the FS-Device via ChFAckReq (=1) for indication via LED (light emitting diode) to the user.

FS-Master and FS-Device keep SDin and SDout until the acknowledgment ChFAck_C (=1) arrived.

11.5.4.7 FS-Device detects CRC signature error

Figure 50 shows the sequence chart when the FS-Device detects a CRC signature error.

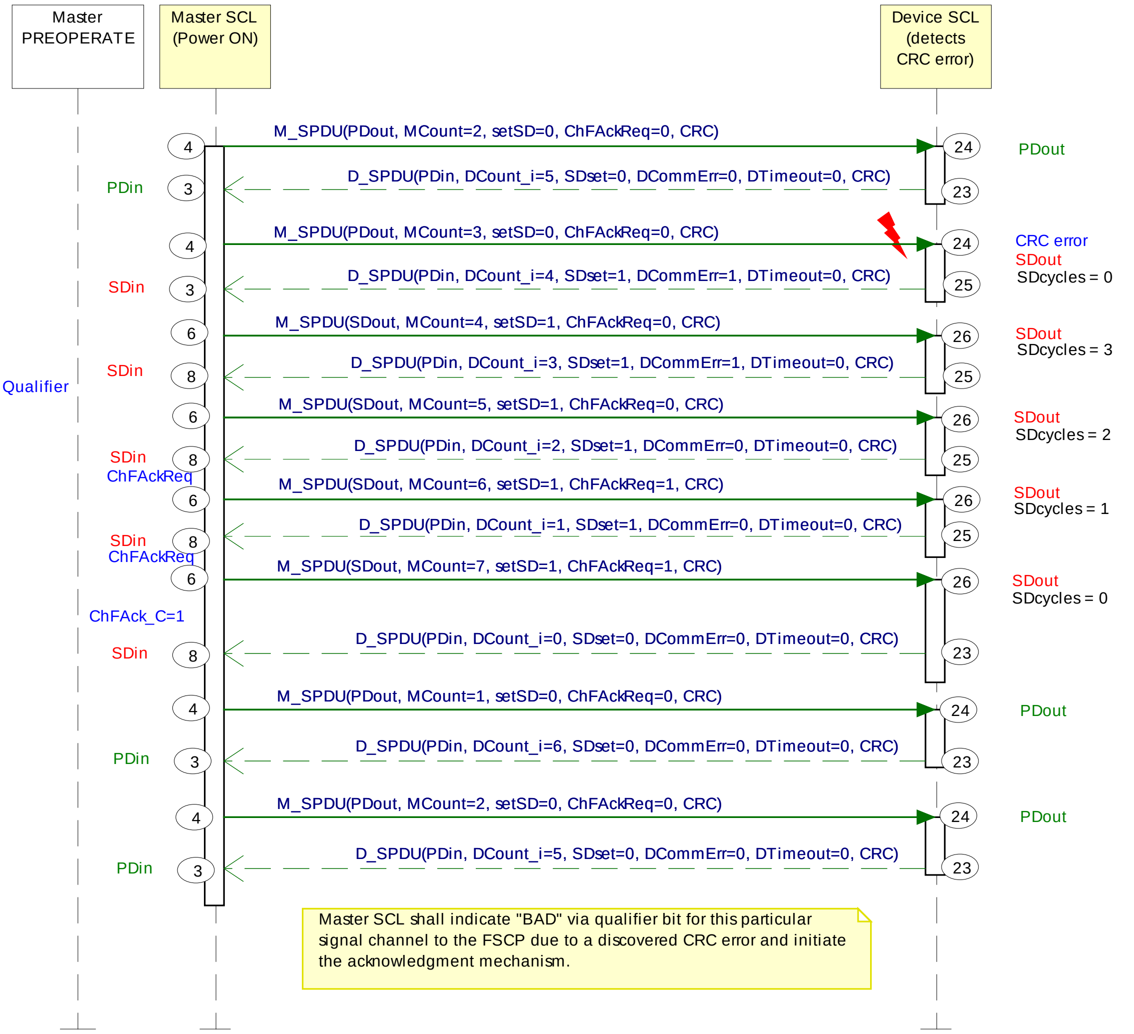


Figure 50 – FS-Device detects CRC signature error

FS-Device received an SPDU with falsified data or falsified CRC signature which results in a "CRC error" (DCommErr). Both FS-Master and FS-Device switch to SDin and SDout respectively caused by FS-Device Status Byte information (SDset=1 and DCommErr=1). The FS-Master/Gateway creates a qualifier bit and indicates a ChFAckReq signal. This signal is indicated also to the FS-Device via ChFAckReq (=1) for indication via LED (light emitting diode) to the user.

The FS-Device runs through 3 SDcycles and afterwards FS-Master and FS-Device keep SDin and SDout until the acknowledgment ChFAck_C (=1) arrived.

11.5.5 Monitoring of safety times

Figure 51 illustrates IO-Link times and safety times.

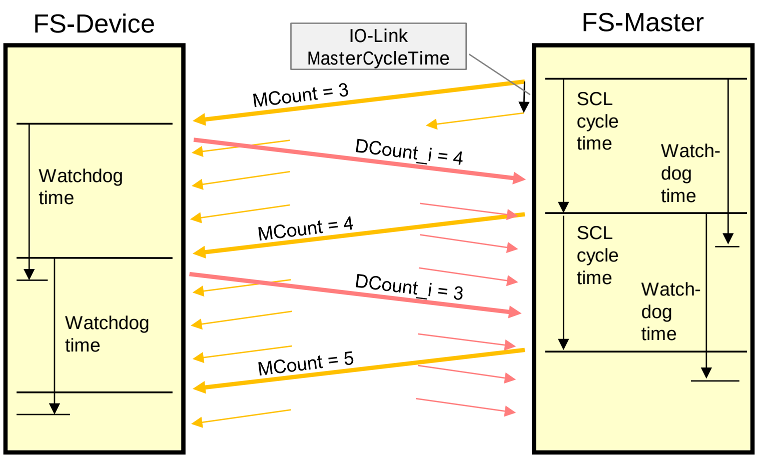


Figure 51 – Monitoring of the SCL cycle time

The base IO-Link system ("black channel") transmits SPDUs within the IO-Link MasterCycleTime (see [1], Table B.1) from the FS-Master to the FS-Device and back. The same SPDU, for example with MCount = 3, may be sent several times before the Safety Communication Layer (SCL) starts the next SCL cycle with MCount = 4. In the meantime, the FS-Master received the response SPDU from the FS-Device with DCount_i = 4.

Table 30 shows timing constraints.

Table 30 – Timing constraints

Item	Constraints	
Synchronization	At each start-up and after an FS-Master timeout, the FS-Master SCL uses MCount = 0	
SCL cycle time	The SCL cycle time comprises the transmission time of the FS-Master SPDU, the FS-Device processing time, the transmission time of the FS-Device response SPDU, and the FS-Master processing time until the next FS-Master SPDU (see Figure 51)	
Watchdog time	The entire SCL cycle time is monitored by the watchdog timer, whose time value is defined by the parameter FSP_Watchdog (see A.2.6).	
Counter check	The counter values are included in the cyclic CRC signature calculation. An incorrect CRC signature value will already lead immediately to a safe state. The immediate counter check in some states is used for discarding "outdated SPDUs".	
Repetition	Repetition in case of detected incorrect CRC signatures is not provided	
PFH-Monitor	The FS-Master holds the information about the reliability of both SPDU transmissions from the FS-Device and to the FS-Device (see Table 21, bit 1). Thus, the FS-Master monitors the average probability of a dangerous failure within a given time frame (PFH-Monitor time). The FS-Master state machine is designed such that any corrupted SPDU leads always to a safe state. Whenever the unlikely event of a detected corrupted SPDU occurs during the shift of production or operation, the responsible operator is assigned to play the role of the PFH-Monitor and can tolerate the indication and acknowledge it. In case of frequent indications more often than once per PFH-Monitor time, a check of the installation or the transmission quality should be performed (see Annex H.6).	
PFH-Monitor time (h)	10	FSP_ProtMode = 0x01; 16 bit CRC, see A.2.5
	10	FSP_ProtMode = 0x02; 32 bit CRC, see A.2.5

11.5.6 Reaction in the event of a malfunction

11.5.6.1 General

Subclauses 11.5.6.2 to 11.5.6.10 specify possible communication errors. They are derived from clause 5.3 in IEC 61784-3, Ed.3, and refer to Table 16 in this document. Additional notes are provided to indicate the typical behavior of the IO-Link black channel.

11.5.6.2 Corruption

Messages may be corrupted due to errors within a communication participant, due to errors on the transmission medium, or due to message interference.

NOTE 1 Bit falsifications within messages during transfer is a normal phenomenon for any standard communication system, such errors are detected at receivers with high probability by use of a hash function, in case of IO-Link a checksum (CKT or CKS), and the message is ignored (Appendix A.1, and clause 7.2.2.1 in [1] or [2]). After two retries the Master initiates a complete restart with wake-up.

NOTE 2 If the recovery or repetition procedures take longer than a specified deadline, a message is classed as "Unacceptable delay" (see 11.5.6.6).

Countermeasures:

The CRC signature as specified in 11.4.6 detects the bit errors in messages between FS-Master and FS-Device to the extent required for SIL3 applications. The CRC signature is generated across the SPDU including the PD or SD data, and the Control&MCnt or Status&DCnt octet for cyclic communication.

At start-up, the FSP parameters are sent once to the FS-Device via ISDU services. They are secured by the 16 bit FSP_ProtParCRC signature. The frequency of its occurrence is assumed to be 1/day as parameter for the calculation of the residual error rate.

The CRC signature of the first SPDU sent by the SCL of the FS-Master after start-up includes the FSP_ProtParCRC signature. All following cyclic SPDUs exclude this signature.

11.5.6.3 Unintended repetition

Due to an error, fault or interference, messages are repeated.

NOTE 1 Repetition by the sender is a normal procedure when an expected acknowledgment/response is not received from a target station, or when a receiver station detects a missing message and asks for it to be resent.

Countermeasures:

The data within the black channel are transferred cyclically. Thus, an incorrect message with an SPDU that is inserted once will immediately be overwritten by a correct message. The thereby possible delay of a demand can be one DTime/MTime.

11.5.6.4 Incorrect sequence

Due to an error, fault or interference, the predefined sequence (for example natural numbers, time references) associated with messages from a particular source is incorrect.

NOTE 1 In IO-Link only one sequence is active from one source, the message handler.

Countermeasures:

The receiver will detect any incorrect sequence due to the stringently sequential expectation of the MCount and DCount values.

11.5.6.5 Loss

Due to an error, fault or interference, a message or acknowledgment is not received.

Countermeasures:

Lost information will be detected by stringently changing and examining the MCount/DCount and/or MTime/DTime within the safety communication layer of the respective receiver.

11.5.6.6 Unacceptable delay

Messages may be delayed beyond their permitted arrival time window, for example due to bit falsifications in the transmission medium, congested transmission lines, interference, or due to communication participants sending messages in such a manner that services are delayed or denied (for example FIFOs in switches, bridges, routers).

NOTE 1 IO-Link provides a point-to-point communication interface with defined message sequences and thus the probability for congestion and storage of messages is very low.

Countermeasures:

A consecutive counter in each message (MCount/DCount) together with a watchdog timer (MTime/DTime) will detect unacceptable delays.

11.5.6.7 Insertion

Due to a fault or interference, a message is received that relates to an unexpected or unknown source entity.

NOTE 1 These messages are additional to the expected message stream, and because they do not have expected sources, they cannot be classified as Correct, Unintended repetition, or Incorrect sequence.

NOTE 2 IO-Link provides a point-to-point communication interface (Port) and thus the probability for insertion of messages is very low.

Countermeasures:

The receiver will detect any incorrect sequence due to the stringently sequential expectation of the MCount and DCount values.

11.5.6.8 Masquerade

Due to a fault or interference, a message is inserted that relates to an apparently valid source entity, so a misdirected non-safety related message may be received by a safety related participant, which then treats it as safety related correct message.

NOTE 1 Communication systems used for safety-related applications can use additional checks to detect Masquerade, such as authorised source identities and pass-phrases or cryptography.

NOTE 2 IO-Link provides a point-to-point communication interface (Port) and thus the probability for insertion of messages is very low.

Countermeasures:

The receiver will detect any incorrect sequence due to the stringently sequential expectation of the MCount and DCount values. After changes of wiring, the FS-Devices can detect misconnections through the FSP_Authenticity1/2 and FSP_Port parameters (see A.2.1 and A.2.2) at start-up.

11.5.6.9 Addressing

Due to a fault or interference, a safety related message is delivered to the incorrect safety related participant, which then treats reception as correct. This includes the so-called loopback error case, where the sender receives back its own sent message.

NOTE 1 The probability of not detecting a misdirected non-safety related message is lower than the probability of not detecting a misdirected safety related message since the SPDU structures are similar due to the shared protocol procedures.

NOTE 2 IO-Link provides a point-to-point communication interface (Port) and thus the probability for insertion of messages is very low.

Countermeasures:

The receiver will detect any incorrect sequence due to the stringently sequential expectation of the MCount and DCount values. After changes of wiring, the FS-Devices can detect misconnections through the FSP_Authenticity1/2 and FSP_Port parameters (see A.2.1 and A.2.2) at start-up.

11.5.6.10 Loop-back

A special addressing error is the so-called loopback error case, where the sender receives back its own sent message.

Countermeasures:

IO-Link Safety provides for inverted values for MCount and DCount from the FS-Device.

11.5.7 Start-up (communication)

An FS-Device starts always after an FS-Master since the FS-Master shall be the only one to power-up at least the communication part of the FS-Device. Both devices usually require time for safety self-tests that may exceed the standard timings defined in [1].

Due to the initial behavior of an FS-Device as an OSSDe, the start-up is coordinated and specified in 5.7, 7.2, and 7.3.

The start-up of the underlying IO-Link communication system is specified in [1] and Figure 55. Any deviating FSP authenticity or protocol parameter CRC signature shall lead to a safe state of the particular FS-Master port.

11.6 SCL management

11.6.1 Parameter overview (FSP and FST)

Annex A specifies a number of functional safety related parameters for communication protocol services (FSP) as well as for the handling and integrity purposes of FS-Device technology parameters (FST).

The parameters are subdivided into 4 groups:

- Authenticity
- Safety communication
- FS-I/O structure description
- Auxiliary parameters

The authenticity parameters combine the safety connection ID ("A-Code") of the FS-Master (assigned by the upper level FSCP system) with the port number of the connected FS-Device. Due to the point-to-point nature of the FS-Device communication with its Master, a one-time check at start-up is sufficient to ensure authenticity (see 11.7.5).

The Safety Communication Layers (SCL) require parameters for protocol versions, protocol modes such as CRC-16 or CRC-32, watchdog for timeliness, CRC signature to secure technology parameters, and a CRC signature to secure the safety communication parameters.

The next group contains a description of the FS I/O data structure, the FS-Device wants to exchange with the FSCP-Host. This description facilitates the mapping to the description which some FSCP systems require for set-up. During the mapping process the FS-Master tool appends the qualifier bits, which are necessary for port-selective passivation.

Auxiliary parameters are specified for several purposes. For example, to secure the functional safety parameter description within the IODD, to support the automatic calculation of the minimum watchdog time, to protect parameters from unauthorized access via a password, and to enable invocation of an associated IOPD tool.

Figure 52 shows an overview of the components and the activities around parameterization.

An FS-Master as a gateway comes with a parameter description file for the FSCP system. With the help of an engineering tool and these parameters, the FS-Master receives during commissioning for example its FSCP connection ID ("A-Code" for authenticity) and its FSCP watchdog time ("T-Code" for timeliness). Thus, the FSCP communication cycles are independent from the IO-Link safety communication cycles between FS-Master and FS-Device.

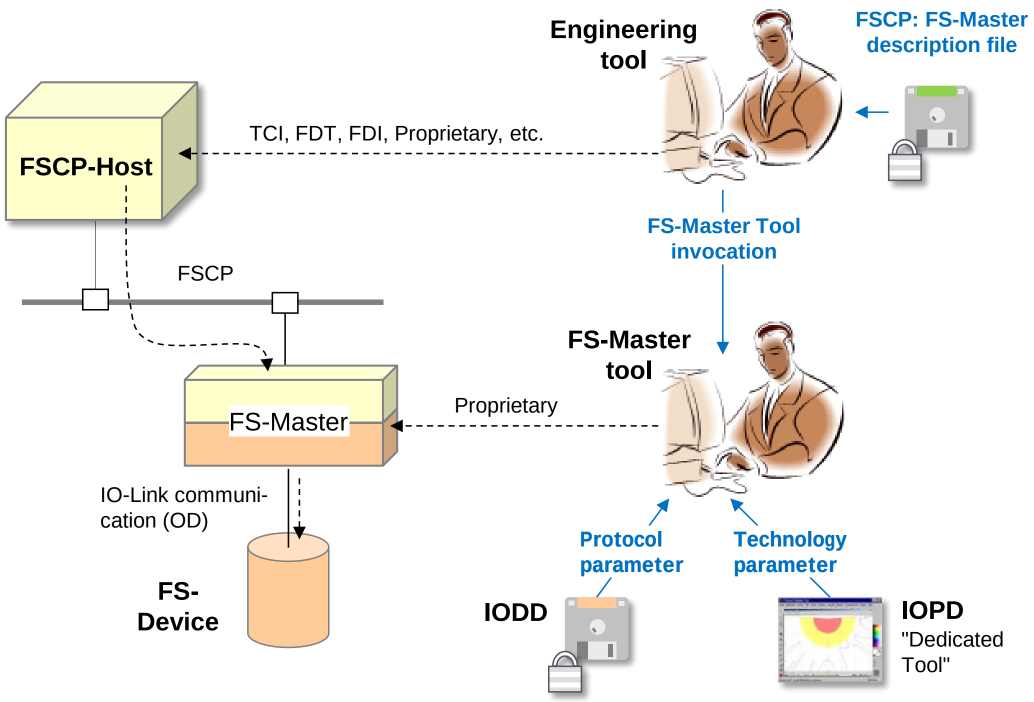


Figure 52 – Parameter types and assignments

An FS-Master with its IO-Link side can be configured and parameterized with the help of its FS-Master tool. The IODD of an FS-Device contains besides the non-safety parameters also the safety section with the parameters in Annex A. The parameters can be set-up off-line or online the same way as with a non-safety system. The FSCP authenticity parameter can be copied from the engineering tool display to the IO-Link system FS-Master tool display (see A.2.1).

It is possible to describe a small set of technology parameters (FST) in a non-safety manner, thus allowing the usage of the IO-Link standard data storage mechanism as described in 9.4.

However, a separate dedicated IOPD tool, developed according IEC 61508-3 shall be used to calculate a CRC signature across the instance of the FST parameters. This CRC signature shall be entered into the corresponding FSP parameter (see A.2.7).

The IOPD tool uses a new standardized IOPD communication interface and the same path to the FS-Device as the FS-Master tool itself.

11.6.2 Parameterization approaches

11.6.2.1 FS-Master-centric

The configuration and parameterization of a stand-alone IO-Link safety system corresponds mainly to the approach described in 11.6.1. The authenticity uses a default value in this case (see A.2.1).

Figure 52 shows a loosely coupled system, where the parameterization is performed within the IO-Link safety part. Within the FSCP system, predefined FS I/O data structures are available and can be selected during commissioning.

11.6.2.2 FSCP-Host-centric

Some automation application areas prefer an FSCP-Host-centric approach. In this case, all parameters are expected to be stored within the FSCP-Host and downloaded at start-up into the FS-Master (FSCP-subsystem) and further down into the FS-Device.

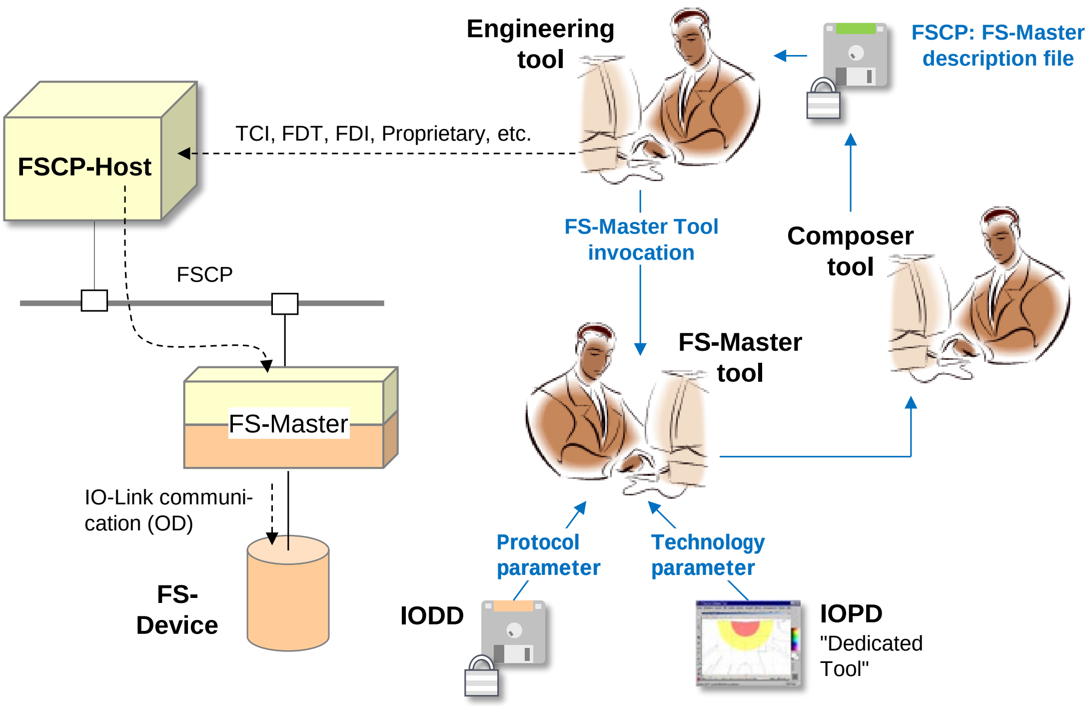


Figure 53 – FSCP-Host-centric system

Due to the fieldbus-independent design of IO-Link and IO-Link safety, all parameters are mapped into the fieldbus device description file (for example EDS, GSD, etc.) with the help of a Composer tool. It is one of the objectives of IO-Link safety to optimize the design of safety parameters such that an efficient mapping is possible.

11.7 Integrity measures

11.7.1 IODD integrity

The parameters specified in Annex A are coded in an IODD file using XML. In order to protect the safety parameter description within this file, a CRC signature ("FS_IODD_CRC") shall be calculated across its safety-related parts (see Annex D and Annex E.3). Usually, the IODD file travels many ways and the CRC signature helps detecting potentially scrambled bits.

11.7.2 Tool integrity

When opening the IODD, the FS-Master tool (interpreter of the IODD file) shall calculate the CRC signature across the safety-related parts and compare the result with the parameter "FSP_ParamDescCRC".

During the data manipulations within the FS-Master tool as well as within Device Tools/IOPDs ("Dedicated Tools") such as display, intended modification, storage/retrieval, and down/upload, parameter values could become incorrect. It is the responsibility of the designer to develop the tools fulfilling the requirements of IEC 61508-3 or ISO 13849-1 for software tools classified as T3.

In case of an FSCP-Host-centric system, these requirements apply for the Composer and the Engineering tool.

11.7.3 Transmission integrity

Since communication between the FS-Master tool and the FS-Device is proprietary, it is the responsibility of the FS-Master tool to ensure transmission integrity and authenticity, for example through CRC signatures and/or read back (see Table 16 and D.3.1).

11.7.4 Verification and validation

It is the responsibility of the FS-Device designer to specify the necessary verification and validation steps (for example tests; see H.6) within the user/safety manual and/or within the "dedicated tool" (IOPD).

11.7.5 Authenticity

In either the FS-Master-centric or in the FSCP-Host-centric approach a record of parameter data is stored in the FS-Master per port/FS-Device as shown in Figure 54**.**

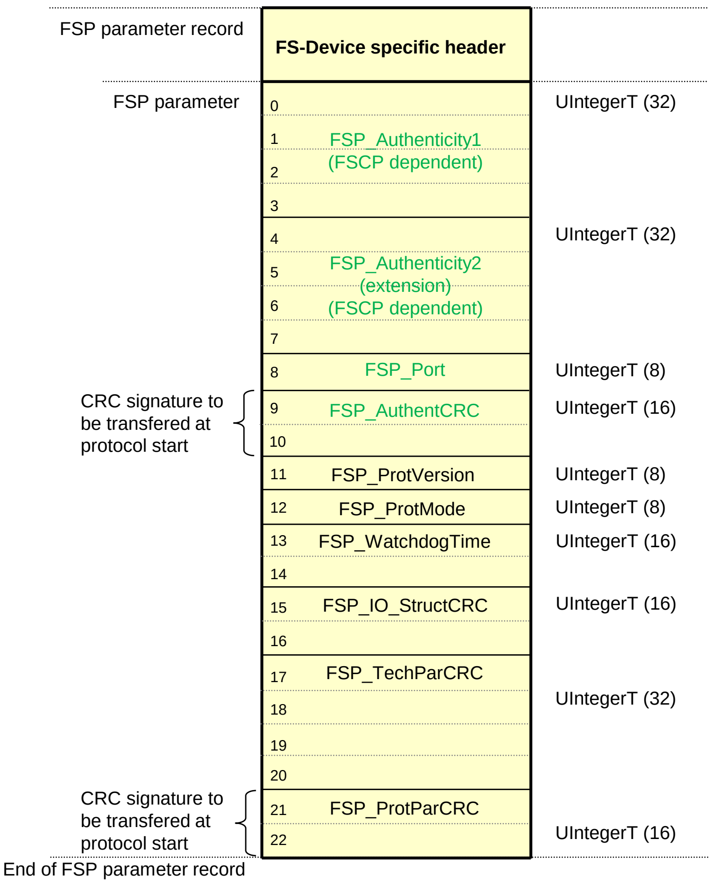


Figure 54 – Structure of the protocol parameter (FSP) record

The authenticity parameters are secured by FSP_AuthentCRC for transmission from FS-Master Tool to FS-Master and further to the FS-Device. The procedure of the FSCP authenticity acquisition from the FSCP gateway and subsequent handling of the FSP authenticity record is described in 10.2.3.3.

11.7.6 Storage integrity

Both parameter records (authenticity and protocol) of Figure 54 are stored in both FS-Master and FS-Device and may fail over time (see also Table A.1).

At each start-up, the FS-Master transfers both parameter records to the FS-Device during PREOPERATE as shown in Figure 55 and described in 10.2.3.1.

The FS-Device will detect a potential mismatch between the downloaded authenticity parameter set and the already stored values in the FS-Device, for example if FS-Devices are misconnected to a different port or even to a different FS-Master. The FS-Device stores authenticity parameters only during commissioning, i.e. when the FSP_TechParCRC signature value is "0". When the FSP_TechParCRC signature value is ≠ "0", The FS-Device will only compare the stored authenticity values with the newly transferred values.

The protocol parameters are propagated to the safety communication layer at each start-up. The protocol engine detects any mismatch between the locally stored parameters (for example due to falsified bits) and the newly transferred record during initialization and blocks safety communication.

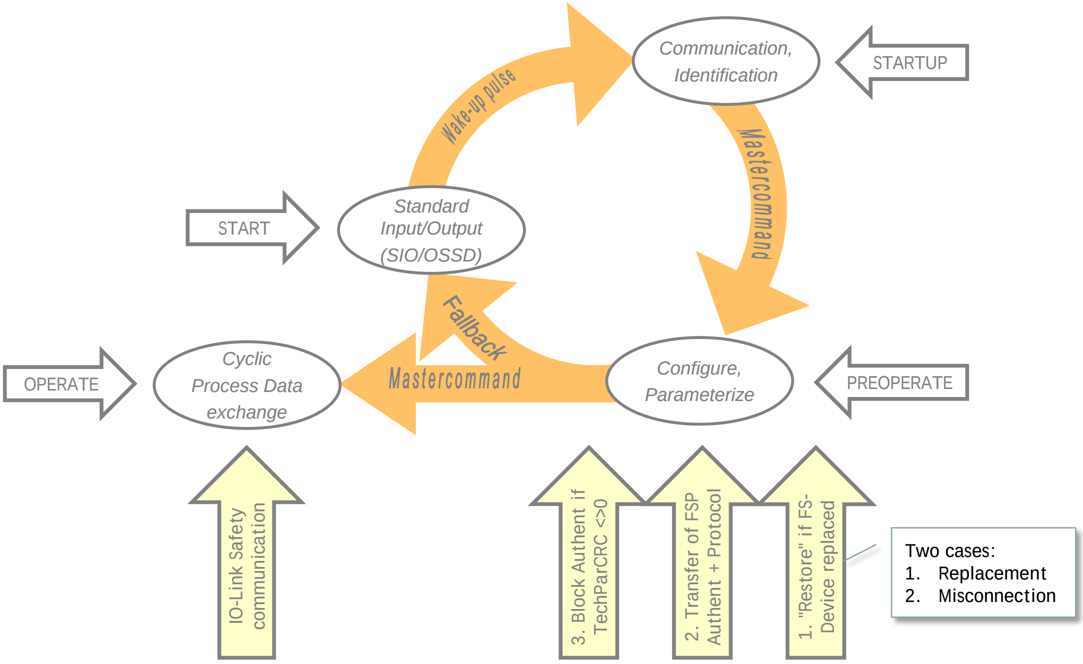


Figure 55 – Start-up of IO-Link safety

In case the FS-Device has been replaced due to a failure, the technology specific parameters (FST) and the FSP parameters are "restored" from Data Storage if the FS-Device caries all Authenticity parameters = "0". If Authenticity is not "0", the FS-Device shall ignore them and keep the existing (see 9.4, E.5.7, and step 1. in Figure 55). In this case a misconnection can be assumed or the FS-Device has already been in use and requires testing and a reset of the Authenticity parameters (see Annex G).

11.7.7 FS I/O data structure integrity

All I/O data of the connected FS-Devices should be mapped in an efficient manner into the FSCP I/O data as shown in 12.1.

Due to the additional qualifier bits required for port-selective passivation, the original FS-Device specific data structure is not directly visible within the FSCP I/O data structure exchanged with the FSCP-Host.

The safety-related interpreter of the FS-Master Tool transfers the entire instance data together with the CRC signature to the FS I/O data mapper as shown in 10.2.3.1 (see also A.2.9).

11.7.8 Technology parameter (FST) based on IODD

One of the objectives of IO-Link safety is FS-Device exchange without tools by using the original data storage mechanism of IO-Link. As a precondition, the FST-parameter description is required within the IODD (see E.5.7).

The FST parameters are displayed in this case within the FS-Master tool (see Figure 56, FST-Parameters section). Values can be assigned as with non-safety parameters.

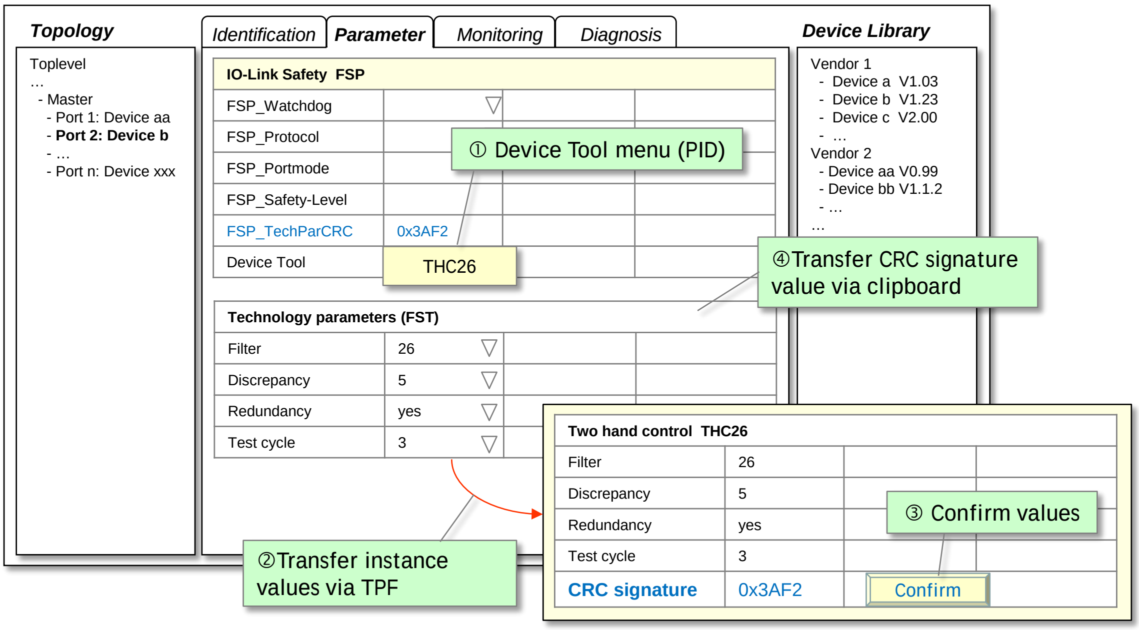


Figure 56 – Securing of FST parameters via dedicated tool

After test and validation, the Device Tool is invoked via menu (step①). Instance values are transferred via TPF (step②) and displayed again. The user compares the instance values and confirms the correctness via the "Confirm" button (step③). The Device Tool then calculates the CRC signature across the instance data of the FST parameters (see "CRC signature" in Figure 56), which can be copied and pasted into the "FSP_TechParCRC" field of the FSP parameters (step ④).

Since this parameter is part of the FSP parameter block, the FS-Device can check the integrity of these FST parameters together with the protocol parameters.

11.7.9 Technology parameter (FST) based on existing dedicated tool (IOPD)

In cases, where existing safety devices already have their PC program with password protection, wizards, teach-in functions, verification instructions, online monitoring, diagnosis, special access to device history for the manufacturer, etc., an FST parameter description may not be available. Figure 57 shows an example.

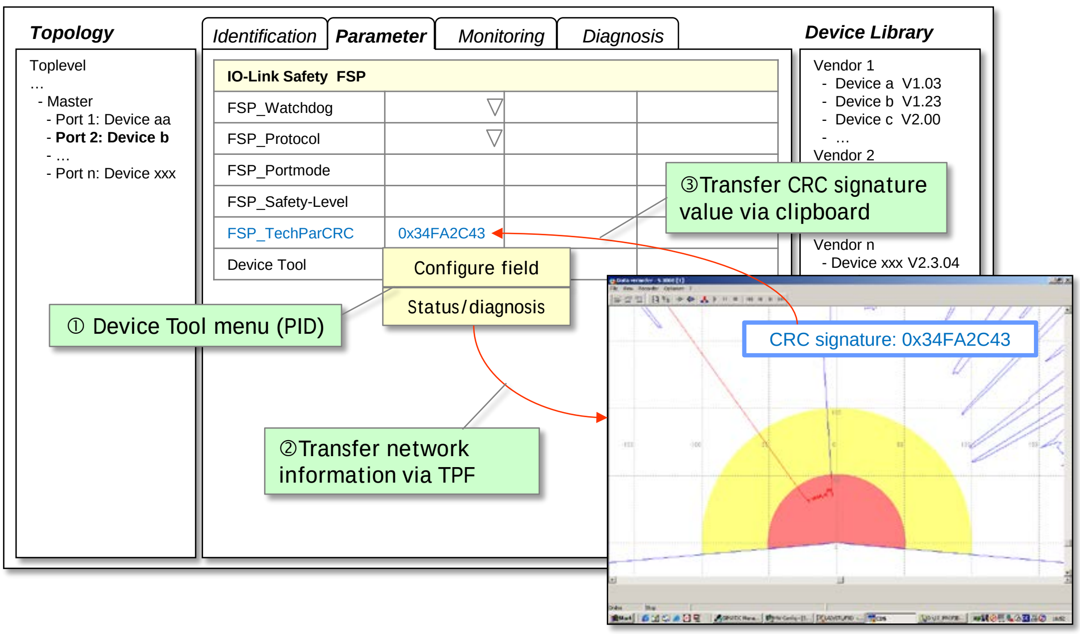


Figure 57 – Modification of FST parameters via Device Tool

Such a Device Tool requires communication with its particular FS-Device and therefore access to a Communication Server (see Annex F.5). It can be invoked via menu entries (step①) and thus jump directly into for example configuration or status/diagnosis functions. Network information is transferred via TPF (step②). After test and validation, it shall provide a display of the calculated CRC signature across the instance data, which can be copied and pasted into the "FSP_TechParCRC" field of the FSP parameters (step③).

These FS-Devices shall be supported by the data storage mechanism of IO-Link and an FS-Device replacement without tools is possible.

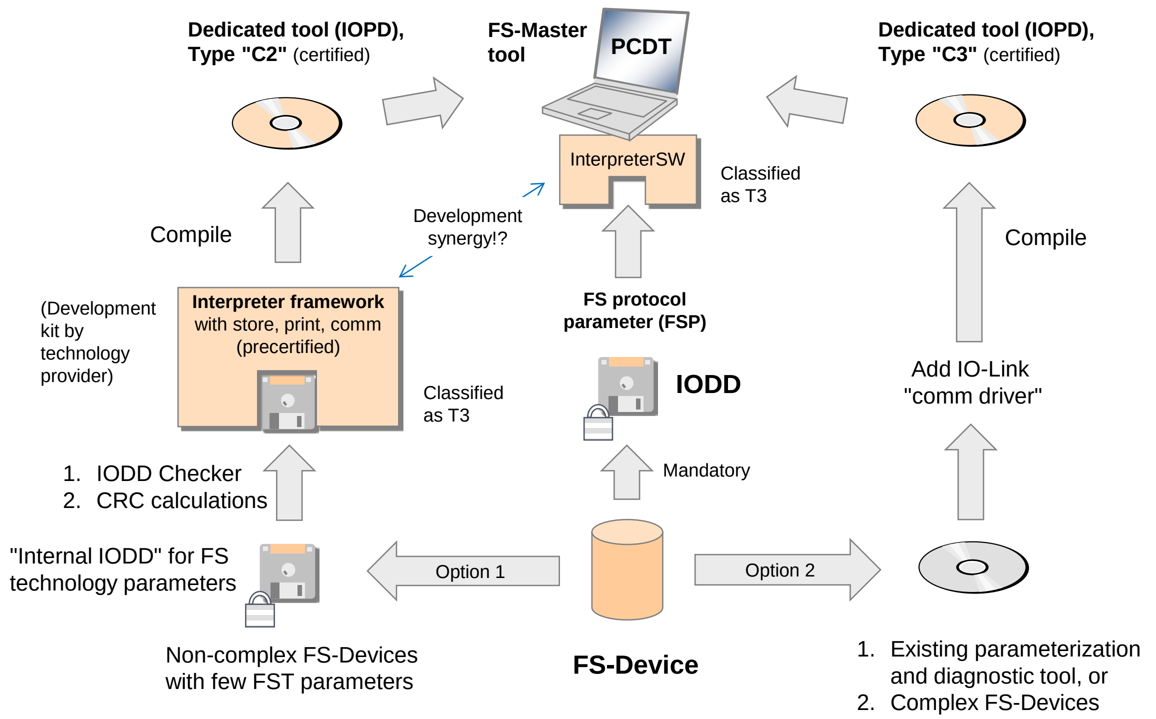
The Data Storage limit per FS-Device is 2048 octets according to [1].

11.8 Creation of FSP and FST parameters

Standards for "Safety-for-Machinery" such as ISO 13849-1 and IEC 62061 require "dedicated tools" for the parameterization of safety devices. For the ease of development and logistics of software tools it is recommended to use the process described in Figure 58.

For FS-Devices with only a few FST parameters, no business logic, and no wizard and help systems, one particular "Interpreter Framework" should be developed in a safe manner according to IEC 61508 and equipped with the necessary communication interfaces. The result will be made available for the whole IO-Link Safety community as a development kit at a certain fee. The FS-Device developer can create an individual "Internal IODD" for the FST parameters of a particular FS-Device (Option 1 in Figure 58). The "Interpreter Framework" together with the individual "Internal IODD" will then be compiled using the brand name, company and FS-Device identifiers to one dedicated tool (IOPD). This executable software can be certified by assessment bodies.

For FS-Devices with more complex FST parameters, the IOPD can be developed individually or existing tools can be used. In both cases the tools can be equipped with the necessary communication interfaces (Option 2 in Figure 58).



Key **IOPD** = **IO**-Link **P**arameterization & **D**iagnostic T3 = software tools are classified T3 according ISO 13849 and IEC 61508-3

Figure 58 – Creation of FSP and FST parameters

In any case, the dedicated tool (IOPD) shall calculate and display the CRC signature across all FST parameters. This signature can be copied into the entry field of the FSP parameter

"FSP_TechParCRC", such that an FS-Device can verify the correctness of locally stored FST parameters after start-up and download of the FSP parameter set to the FS-Device.

For each and every FS-Device the same set of FSP (protocol) parameters shall be created in an extended IODD. This IODD is mandatory and contains the usual conventional parameters and additionally the FSP parameters.

11.9 Integration of dedicated tools (IOPD)

11.9.1 IOPD interface

Usually, a so-called Master-Tool (PCDT) provides engineering support for a Master and its Devices via Device descriptions in form of XML files (IODD). In principle, this is the same for FS-Master and FS-Device. For functional safety besides an extended IODD it is necessary for an FS-Device vendor to provide an additional Dedicated Tool (IOPD) as shown in Figure 58.

In order for the IOPD to communicate with its FS-Device a new standardized communication interface is required.

11.9.2 Standard interfaces

Usually, Master Tools are integrated using existing standards such as FDT, the upcoming FDI, or proprietary solutions. Such a variety is not acceptable for FS-Devices and therefore, easy and proven-in-use technology has been selected and adopted for IO-Link. It is called "Device Tool Interface" (DTI).

Annex F provides the specification for this interface.

Figure 59 illustrates the communication hierarchy of FDT and others for the fieldbus as well as the connection via the "Device Tool Interface" and the underlying IO-Link communication.

The FS-Device Tool (IOPD) does not have to know about the fieldbus environment it is connected to. The example in Figure 59 illustrates how it sends a "Read Index 0x4231" service and how the FS-Master Tool packs this service into a fieldbus container and passes it to the fieldbus communication server.

The addressed FS-Master is connected to the fieldbus and receives this container. It unpacks the IO-Link Read service and performs it with the addressed FS-Device connected to a port.

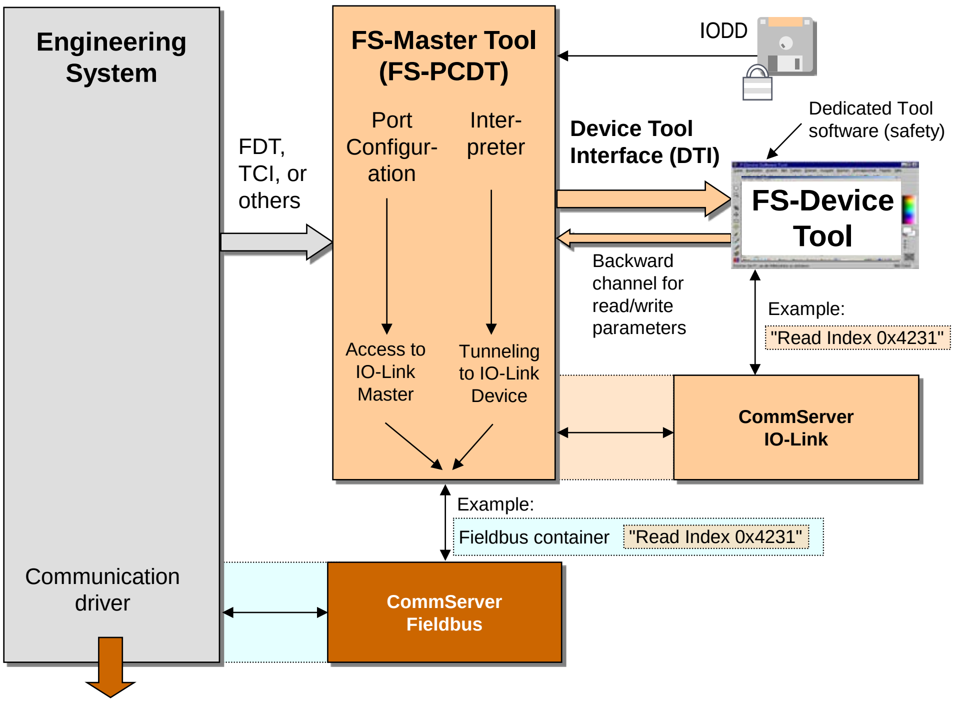


Figure 59 – Example of a communication hierarchy

11.9.3 Backward channel

An FS-Master vendor does not know in advance which FS-Devices a customer wants to connect to the FS-Master ports. As a consequence, the fieldbus device description of such an FS-Master can only provide predefined "containers" for the resulting I/O data structure of the FS-Device ensembles connected to the ports. In functional safety this is even more complicated since the description of the data structures shall be coded and secured.

As a consequence of the variety of different configurations and parameterizations of a particular FS-Device, it usually for example

- requires different I/O data structures to exchange with PLCs or hosts;
- has different reaction times due to configured high or low resolutions;
- can have different SIL, PL, category, or PFH values impacting the overall safety level of a safety function.

The classic "fieldbus device description" to inform an engineering system is not flexible in this respect. Its advantage is the testability and certification for its interoperability with engineering tools.

Nevertheless, a "backward channel" within the tool interfaces allows for nowadays flexible manufacturing and ease of engineering and commissioning. An example is specified in [15] clause 4.15.5.

Annexes F.3.5 and F.9.4 specify an extension for this "backward channel".

11.10 Passivation

11.10.1 Motivation and means

Figure 60 illustrates the motivation for Port selective passivation. Usually for efficiency reasons, the signals 0 to 7 of FS-Devices connected to Ports are not mapped individually to an FSCP-PDU, but rather packed into one FSCP-PDU. Each of these signals can be assigned to a separate safety function n to n+7. If a fault occurs in one of the signal channels, a collective passivation for the entire FSCP-PDU would be necessary causing all safety functions to trip.

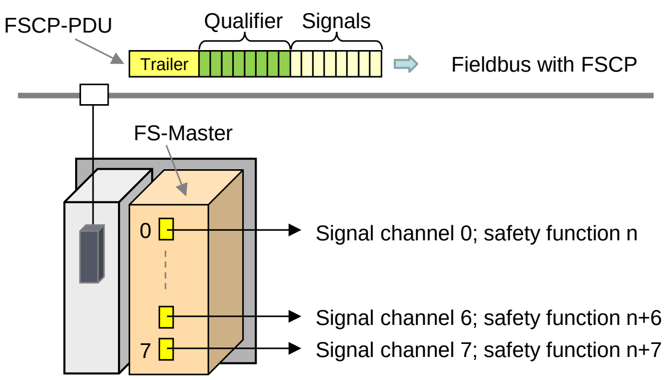


Figure 60 – Motivation for Port selective passivation

FSCPs usually provide so-called qualifier bits associated to the signal process data, which enable selectively passivating that particular signal channel and the associated safety function.

Safety of machinery usually requires an operator acknowledgment after repair of a defect signal channel to prevent from automatic restart of a machine. It is not necessary to provide the acknowledgment for each signal channel and it can be one for all the channels.

11.10.2 Port selective (FS-Master)

In 11.10.1 a use case is described where the signal channel corresponds directly with a particular FS-Device. The qualifier and acknowledgement mechanism shall be implemented within the FS-Master in accordance with the specifications of the particular FSCP.

It can be helpful for the user to provide an indication in each FS-Device that an operator acknowledgment is required prior to a restart of a safety function. CB0 (ChFAckReq) within the Control&MCnt octet (see Table 20) shall be used for that purpose. It is not safety-related.

Optionally, in case of FS_PortMode "OSSDe" (see 10.2.2), the signal ChFAckReq can be connected separately to the corresponding FS-Device indication (see H.1).

11.10.3 Signal selective (FS-Terminal)

Figure 13 shows the use case of an FS-Terminal where an FS-Device provides several signal channels to switching devices such as E-Stop buttons.

For those FS-Devices the design rules in 11.4.7.3 apply. The acknowledgment mechanisms shall be implemented within the safety Process Data.

11.10.4 Qualifier settings in case of communication

Figure 61 illustrates the embedding of the qualifier handler in case of FS_PortModes "SafetyCom" and "MixedSafetyCom" (see 10.2.2). The services/signals "FAULT_S", "SDset_S", "ChFAckReq_S", and "ChFAck_C" are specified in 11.3.2 and 11.5.2.

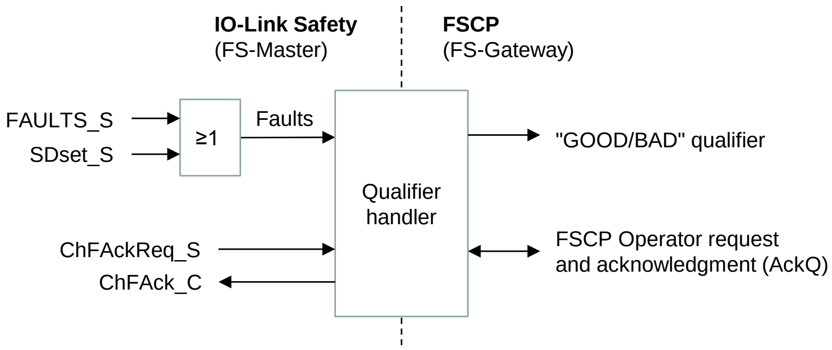


Figure 61 – Qualifier handler (communication)

The qualifier bits "GOOD/BAD" shall be set according to the definitions in Table 31 during the FSCP mapping procedure.

Table 31 – Qualifier bits "GOOD/BAD"

Faults	Qualifier	Signal state
0	GOOD	1
1	BAD	0

11.10.5 Qualifier handling in case of OSSDe

Figure 62 illustrates the embedding of the qualifier handler in case of FS_PortModes "OSSDe" (see 10.2.2). Definitions of Table 31 apply.

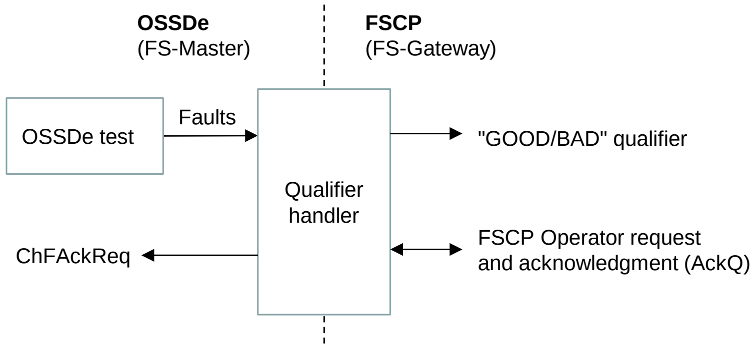


Figure 62 – Qualifier handler (OSSDe)

Figure 63 shows the state machine for the behavior of the qualifier handler (OSSDe).

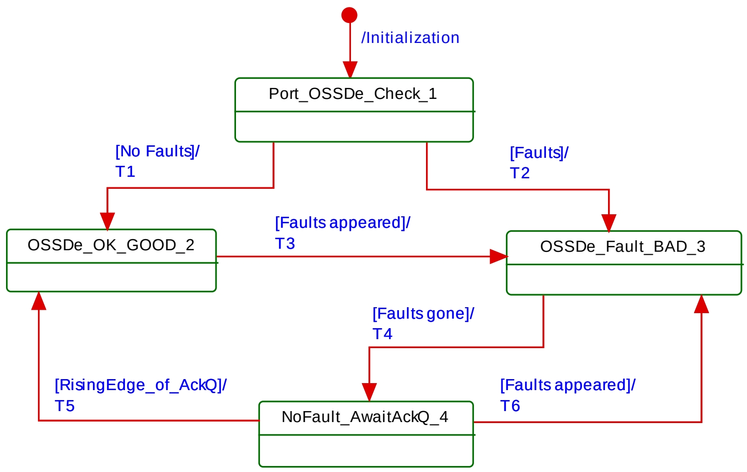


Figure 63 – Qualifier behavior per FS-Master port

Table 32 shows the state and transition table for the qualifier behavior.

Table 32 – State transition table for the qualifier behavior

STATE NAME	STATE DESCRIPTION
Initialization	Use SD, Qualifier = BAD, ChFAckReq =0
1 Port_OSSDe_Check	Perform Port diagnosis to detect Faults
2 OSSDe_OK_GOOD	Perform Port diagnosis cyclically to detect Faults
3 OSSDe_Fault_BAD	Perform Port diagnosis cyclically to detect Faults
4 NoFault_AwaitAckQ	Wait on rising edge of AckQ

TRAN-SITION	SOURCE STATE	TARGET STATE	ACTION
T1	1	2	Use PD, Qualifier = GOOD, AckQ = 0, ChFAckReq =0
T2	1	3	Use SD, Qualifier = BAD, AckQ = 0, ChFAckReq =0
T3	2	3	Use SD, Qualifier = BAD, AckQ = 0, ChFAckReq =0
T4	3	4	Use SD, Qualifier = BAD, AckQ = 0, ChFAckReq =1
T5	4	2	Use PD, Qualifier = GOOD, AckQ = 1, ChFAckReq =0
T6	4	3	Use SD, Qualifier = BAD, AckQ = 0, ChFAckReq =0

INTERNAL ITEMS	TYPE	DEFINITION
RisingEdge_of_AckQ	Flag	Means to prevent from permanently actuating the AckQ signal.
AckQ	Flag	Flag depending on the individual upper level FSCP system and its mapping.

INTERNAL ITEMS	TYPE	DEFINITION
Faults	Flag	Any detected fault such as a wire break, short circuit.
ChFAckReq	Flag	Signal set by qualifier handler (see 11.10.2 and H.1)

11.11 SCL diagnosis

The Safety Communication Layer can create its own EventCodes such as CRC error, counter error, or timeout as listed in Annex B.1.

12 Functional safe processing (FS-P)

12.1 Recommendations for efficient I/O mappings

Figure 64 shows how efficiency can be increased when packing I/O data from the connected safety devices into one FSCP SPDU instead of several individual FSCP SPDUs. On the left, the bits of safety devices (OSSD) are packed into one FSCP SPDU by the FS-DI module. On the right, the FS-Devices use each an FSCP SPDU to transmit a bit. In the middle an IO-Link Safety FS-Master/Gateway packs bits into one FSCP SPDU similar to an FS-DI.

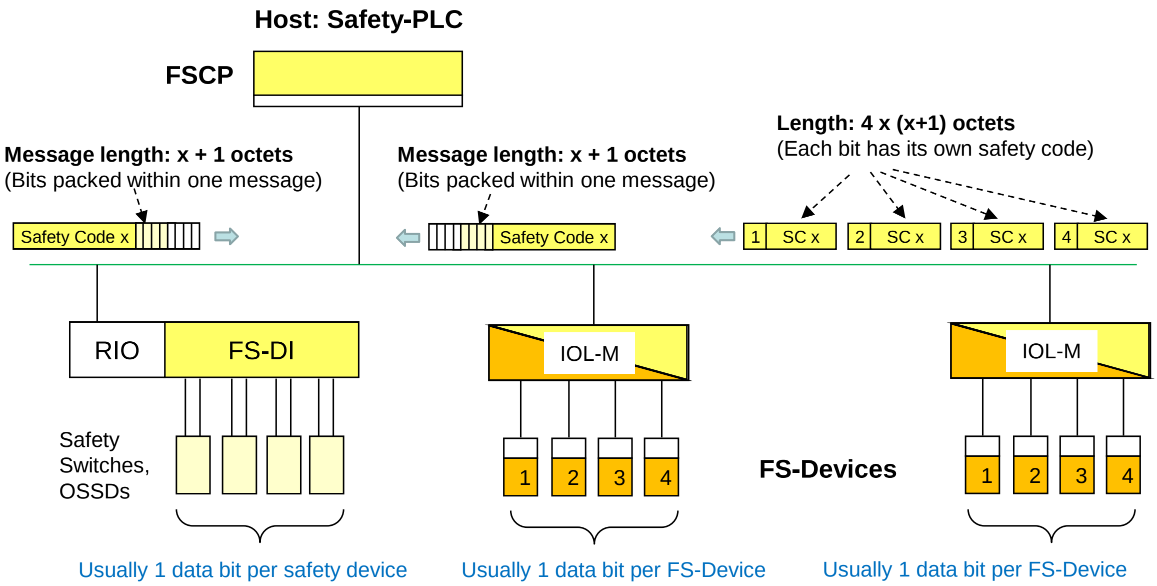


Figure 64 – Mapping efficiency issues

The FS I/O data structure shall be created as a multiple of 16 bits.

12.2 FS logic control

Specification and implementation of an FS logic control to provide local safety functions are manufacturer's responsibility and not standardized (see ③ in clause 1 and Figure 2).

Annex A
(normative, safety-related)

Extensions to parameters

A.1 Indices and parameters for IO-Link Safety

The Index range reserved for IO-Link Safety includes 255 Indices from 0x4200 to 0x42FF.

Table A.1 shows the specified Indices for IO-Link profiles, the protocol parameters (FSP) of IO-Link Safety, comprising authenticity, protocol, I/O data structures, and auxiliary blocks as well as the total reserved range for IO-Link Safety, and the second range of Indices for IO-Link profiles.

Table A.1 – Indices for IO-Link Safety

Index (dec)	Sub index	Object name	Access	Length	Data type	M/O/C	Purpose/reference
		...					
0x4000 to 0x41FF		Profile specific Indices					For example: Smart sensors
		Authenticity (11 octets)					
0x4200 (16896)	1	FSCP_Authen-ticity_1	R/W	4 octets	UIntegerT	M	"A-Code" from the upper level FSCP system; see A.2.1
	2	FSCP_Authen-ticity_2	R/W	4 octets	UIntegerT	M	Extended "A-Code" from the upper level FSCP system
	3	FSP_Port	R/W	1 octet	UIntegerT	M	PortNumber identifying the particular FS-Device; see A.2.2
	4	FSP_Authent CRC	R/W	2 octets	UIntegerT	M	CRC-16 across authenticity parameters; see A.2.3
		Protocol (12 octets)					
0x4201 (16897)	1	FSP_ ProtVersion	R/W	1 octet	UIntegerT	M	Protocol version: 0x01; see A.2.4
	2	FSP_ ProtMode	R/W	1 octet	UIntegerT	M	Protocol modes, e.g. 16/32 bit CRC; see A.2.5
	3	FSP_ Watchdog	R/W	2 octets	UIntegerT	M	Monitoring of I/O update; 1 to 65 535 ms; see A.2.6
	4	FSP_IO_ StructCRC	R/W	2 octets	UIntegerT	M	CRC-16 signature across I/O structure description block; see A.2.9
	5	FSP_TechParCRC	R/W	4 octets	UIntegerT	M	Securing code across FST (technology specific parameter); see A.2.7
	6	FSP_Prot ParCRC	R/W	2 octets	UIntegerT	M	CRC-16 across protocol parameters; see A.2.8
		Auxiliary parameters					
0x4210 (16912)		FS_ Password	W	32 octets	StringT	M	Password for access protection of FSP parameters and Dedicated Tools; see A.2.10
0x4211 (16913)		Reset_FS_ Password	W	32 octets	StringT	M	Password to reset the FST Parameters to factory settings and to reset implicitly the FS-Password; see A.2.11

Index (dec)	Sub index	Object name	Access	Length	Data type	M/O/C	Purpose/reference
0x4212 (16914)		FSP_ ParamDescCRC	R	2 octets	UIntegerT	M	CRC-16 signature securing authenticity, protocol, and FS I/O structure description within IODD; see A.2.12
...							
0x4213 (16915) to 0x42FF (17151)		Reserved for IO-Link Safety					
0x4300 to 0x4FFF		Profile specific Indices					For example: Firmware update and BLOB
...							
Key M = mandatory; O = optional; C = conditional							

A.2 Parameters in detail

A.2.1 FSCP_Authenticity

During off-line commissioning of an IO-Link Safety project, the default value of this parameter is "0". During on-line commissioning, the FS-Master acquires the FSCP authenticity ("A-Code") from the FSCP gateway as described in 10.2.3.1. The FS-Device receives this parameter at each start-up. In case of FSP_TechParCRC = "0", it either stores the authenticity parameter or rejects it with an error message if the value is "0" (see Table B.1). In case the system is armed (FSP_TechParCRC ≠ "0") the FS-Device only compares its locally stored value with the transferred value to detect any misconnection (incorrect port or FS-Master), see Annex G.

Some FSCPs provide extended authenticity. In those cases, the extended code shall be included in this parameter.

Padding bits and octets shall be filled with "0".

A.2.2 FSP_Port

The FS-Master Tool identifies the FS-Master PortNumber of the attached FS-Device and stores it in this parameter. Storage and checking of the parameter by the FS-Device corresponds to A.2.1. Numbering starts at "1". Default PortNumber in IODD is "0" and means PortNumber of a particular Device has not been assigned yet.

A.2.3 FSP_AuthentCRC

For the CRC signature calculation of the entire authenticity block, the CRC-16 in Table D.1 shall be used. This CRC polynomial has a Hamming distance of ≥ 6 for lengths ≤ 16 octets. A seed value "0" shall be used (see D.3.6).

A.2.4 FSP_ProtVersion

Table A.2 shows the coding of FSP_ProtVersion.

Table A.2 – Coding of protocol version

Value	Definition
0x00	Reserved
0x01	This protocol version
0x02 to 0xFF	Reserved

A.2.5 FSP_ProtMode

Table A.3 shows the coding of FSP_ProtMode.

Table A.3 – Coding of protocol mode

Value	Definition
0x00	Reserved
0x01	0 to 4 octets of FS I/O Process Data; 16 bit CRC
0x02	0 to 26 octets of FS I/O Process Data; 32 bit CRC
0x03 to 0xFF	Reserved

A.2.6 FSP_Watchdog

The FS-Device designer determines the I/O update time and uses it as default value of this parameter within the IODD. The I/O update time is the time period between two safety PDUs with subsequent counter values (I/O samples) including possible repetitions within the IO-Link communication layer (black channel; see 11.5.5).

With the help of the parameter default value (I/O update time), the transmission times of the safety PDUs, and FS-Master processing times, the FS-Master Tool can estimate the total time and suggest the value of the "FSP_Watchdog" parameter.

The value range is 1 to 65 535 (measured in ms). A value of "0" is not permitted. The SCL of the FS-Device is responsible to check the validity at start-up and to create an error in case (see Table B.1).

A.2.7 FSP_TechParCRC

This document specifies two basic methods for the assignment of technology specific parameters (FST). The FS-Device designer is responsible for the selection of the securing method.

The method in 11.7.8 is based on IODD and suggests using one of the CRC generator polynomials in Table D.1. If calculation of the CRC signature value results in "0", a "1" shall be used.

The method in 11.7.9 depends on the method used within an existing FS-Device Tool (Dedicated Tool). Whatever method is used, the tool shall display a securing code after verification and validation that can be copied and pasted into the FSP_TechParCRC parameter entry field.

During commissioning a value of "0" can be entered to allow for certain behavior of the IO-Link Safety system (see 10.2.3.1). During production, this value shall be ≠ "0".

For technology specific parameter block transfers > 232 octets, the BLOB mechanism (Binary Large Objects) specified in [13] can be used.

A.2.8 FSP_ProtParCRC

For the CRC signature calculation of the entire protocol block, the CRC-16 in Table D.1 shall be used. This CRC polynomial has a Hamming distance of ≥ 6 for lengths ≤ 16 octets. A seed value "0" shall be used (see D.3.6).

A.2.9 FSP_IO_StructCRC

An IODD-based non-safety viewer can be used to calculate this 16 bit CRC signature across the FS I/O structure description within the IODD during the development phase. The algorithm for the calculation is shown in Annex D. A seed value "0" shall be used (see D.3.6).

The safety-related interpreter of the FS-Master Tool transfers the entire instance data together with the CRC signature to the FS I/O data mapper as shown in 10.2.3.1.

Table A.4 shows Version "1" of the generic FS I/O data structure description for FS-Devices. With the help of this table, individual instances of FS-Device I/O Process Data can be created via IODD and, amongst others, used for an automatic mapping of IO-Link Safety data to FSCP safety data.

Table A.4 – Generic FS I/O data structure description

Item name	Item length	Definition
IO_DescVersion	1 octet	Version of this generic structure description: 0x01
InputDataRange	1 octet	Length in octets of the entire FS input Process Data including the 3 or 5 octets respectively for the safety code (Control/Status and CRC-16/32)
TotalOfInBits	1 octet	Number of the entire set of input BooleanT (bits)
TotalOfInOctets	1 octet	Number of octets with input BooleanT (including unfilled octets)
TotalOfInInt16	1 octet	Number of input IntegerT(16)
TotalOfInInt32	1 octet	Number of input IntegerT(32)
OutputDataRange	1 octet	Length in octets of the entire FS output Process Data including the 3 or 5 octets respectively for the safety code (Control/Status and CRC-16/32)
TotalOfOutBits	1 octet	Number of the entire set of output BooleanT (bits)
TotalOfOutOctets	1 octet	Number of octets with output BooleanT (including unfilled octets)
TotalOfOutInt16	1 octet	Number of output IntegerT(16)
TotalOfOutInt32	1 octet	Number of output IntegerT(32)
FSP_IO_StructCRC	2 octets	CRC-16 signature value across all items (see Annex D.1)

Figure A.1 shows the instance of the FS I/O data description of the example in Figure A.2.

```
IO_DESCVERSION        01     0x01
INPUT_DATA_RANGE      07     0x07
TOTAL_OF_INBITS       13     0x0D
TOTAL_OF_INOCTETS     02     0x02
TOTAL_OF_ININT16      01     0x01
TOTAL_OF_ININT32      00     0x00
OUTPUT_DATA_RANGE     03     0x03
TOTAL_OF_OUTBITS      00     0x00
TOTAL_OF_OUTOCTETS    00     0x00
TOTAL_OF_OUTINT16     00     0x00
TOTAL_OF_OUTINT32     00     0x00
FSP_IO_STRUCTCRC      2386   0x0952
```

Figure A.1 – Instance of an FS I/O data description

Figure A.2 shows an example with FS input Process Data and no FS output Process Data.

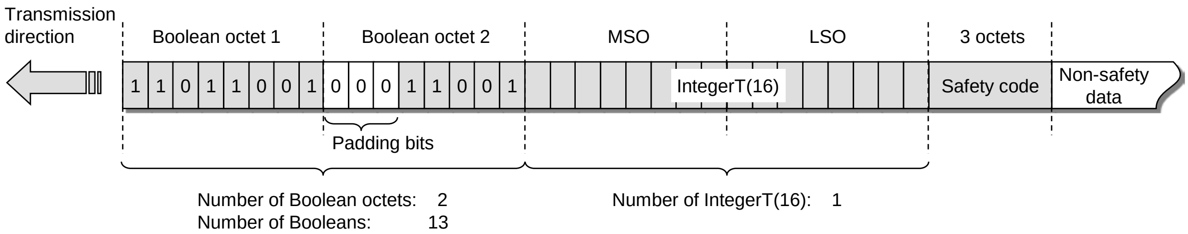


Figure A.2 – Example FS I/O data structure with non-safety data

A.2.10 FS_Password

It is the responsibility of the FS-Master and FS-Master Tool designer to define the password mechanism (e.g. setting/resetting, encryption, protection against easy capturing via line monitors). Maximum size is 32 octets. Encoding shall be ASCII. A mix of upper/lower case characters, numbers, and special characters shall be possible.

A.2.11 Reset_FS_Password

With the help of this password, a reset to factory settings of the FS-Device can be performed including FS_Password. New commissioning can take place with FSP_TechParCRC = "0" (see A.2.7).

A.2.12 FSP_ParamDescCRC

This dummy parameter within the IODD contains the CRC signature calculated across the entire parameter descriptions within the IODD according to the algorithm specified in E.5. The CRC-16 in Table D.1 shall be used. A seed value "0" shall be used since leading "0" parameter values cannot occur (see D.3.6).

The purpose of this parameter is to secure the relevant descriptions of safety parameters within the IODD against data falsification during storage and handling as shown in Figure A.3

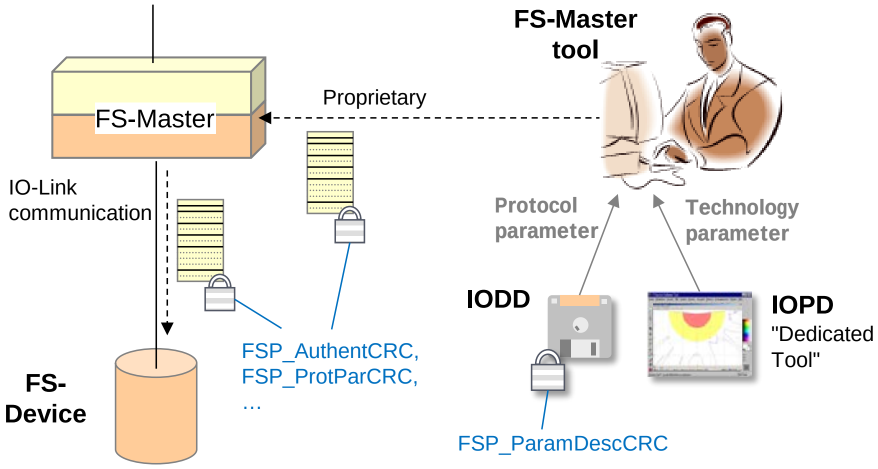


Figure A.3 – Securing of safety parameters

Annex B
(normative, non-safety related)

Extensions to EventCodes

B.1 Additional EventCodes

The Safety Communication Layer (SCL) can create its own EventCodes as shown in Table B.1.

Table B.1 – SCL specific EventCodes

EventCode	Defnition and recommended maintenance action	FS-Device status value	TYPE
0xB000	Transmission error (CRC signature)	2	Warning
0xB001	Transmission error (Counter)	2	Warning
0xB002	Transmission error (Timeout)	3	Error
0xB003	Unexpected authentication code	3	Error
0xB004	Unexpected authentication port	3	Error
0xB005	Incorrect FSP_AuthentCRC	3	Error
0xB006	Incorrect FSP_ProtParCRC	3	Error
0xB007	Incorrect FSP_TechParCRC	3	Error
0xB008	Incorrect FSP_IO_StructCRC	3	Error
0xB009	Watchdog time out of specification (e.g. "0")	3	Error
0xB00A	Reserved: do not use number; do not evaluate number	-	-

Usually, "CRC signature" and/or "Counter" transmission errors are caused by seriously falsified IO-Link messages with SPDUs due to heavy interferences. There is nothing to repair and an operator acknowledgment is sufficient. This very unlikely warning should inform the operator and the responsible production manager about possible changes within a machine requiring an inspection according to the safety manual (see H.6).

Annex C
(normative, safety related)

Extensions to Data Types

C.1 Data types for IO-Link Safety

Table C.1 shows the available data types in IO-Link Safety (see 11.4.7.2).

Table C.1 – Data types for IO-Link Safety

Data type	Coding	Length	See [1]	Device example
BooleanT/bit	BooleanT ("packed form" for efficiency, no WORD structures); assignment of signal names to bits is possible.	1 bit	Clause E.2.2; Table E.22 and Figure E.8	Proximity switch
IntegerT(16)	IntegerT (enumerated or signed)	2 octets	Clause E.2.4; Table E.4, E.7 and Figure E.2	Protection fields of laser scanner
IntegerT(32)	IntegerT (enumerated or signed)	4 octets	Clause E.2.4; Table E.4, E.6, and Figure E.2	Encoder or length measurement (≈ ± 2 km, resolution 1 µm)

C.2 BooleanT (bit)

A BooleanT represents a data type that can have only two different values i.e. TRUE and FALSE. The data type is specified in Table C.2.

Table C.2 – BooleanT for IO-Link Safety

Data type name	Value range	Resolution	Length
BooleanT	TRUE / FALSE	-	1 bit

IO-Link Safety uses solely the so-called packed form via RecordT as shown in Table C.3.

Table C.3 – Example of BooleanT within a RecordT

Subindex	Offset	Data items		Data Type	Name/symbol
1	0	TRUE		BooleanT	Proximity_1
2	1	FALSE		BooleanT	Proximity_2
3	2	FALSE		BooleanT	EmergencyStop_1
4	3	TRUE		BooleanT	EmergencyStop_2
5	4	TRUE		BooleanT	EmergencyStop_3

Figure C.1 demonstrates an example of a BooleanT data structure. Padding bits are "0".

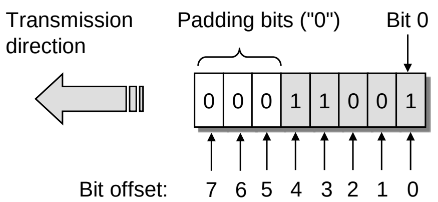


Figure C.1 – Example of a BooleanT data structure

Only RecordT data structures of 8 bit length are permitted. Longer data structures shall use multiple RecordT data structures (see Annex C.5).

NOTE Data structures longer than 8 bit can cause mapping problems with upper level FSCP systems (see 3.4.2)

C.3 IntegerT (16)

An IntegerT(16) is representing a signed number depicted by 16 bits. The number is accommodated within the octet container 2 and right-aligned and extended correctly signed to the chosen number of bits. The data type is specified in Table C.4 for singular use. SN represents the sign with "0" for all positive numbers and zero, and "1" for all negative numbers. Padding bits are filled with the content of the sign bit (SN).

Table C.4 – IntegerT(16)

Data type name	Value range	Resolution	Length
IntegerT(16)	-2^{15} to $2^{15}- 1$	1	2 octets
NOTE 1 High order padding bits are filled with the value of the sign bit (SN). NOTE 2 Most significant octet (MSO) sent first (lowest respective octet number in Table C.5).			

The coding of IntegerT(16) is shown in Table C.5.

Table C.5 – IntegerT(16) coding

Bit	7	6	5	4	3	2	1	0	Container
Octet 1	SN	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2 octets
Octet 2	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	

C.4 IntegerT (32)

An IntegerT(32) is representing a signed number depicted by 32 bits. The number is accommodated within the octet container 4 and right-aligned and extended correctly signed to the chosen number of bits. The data type is specified in Table C.6 for singular use. SN represents the sign with "0" for all positive numbers and zero, and "1" for all negative numbers. Padding bits are filled with the content of the sign bit (SN).

Table C.6 – IntegerT(32)

Data type name	Value range	Resolution	Length
IntegerT(32)	-2^{31} to $2^{31}- 1$	1	4 octets
NOTE 1 High order padding bits are filled with the value of the sign bit (SN). NOTE 2 Most significant octet (MSO) sent first (lowest respective octet number in Table C.7).			

The coding of IntegerT(32) is shown in Table C.7

Table C.7 – IntegerT(32) coding

Bit	7	6	5	4	3	2	1	0	Container
Octet 1	SN	2^{30}	2^{29}	2^{28}	2^{27}	2^{26}	2^{25}	2^{24}	4 octets
Octet 2	2^{23}	2^{22}	2^{21}	2^{20}	2^{19}	2^{18}	2^{17}	2^{16}	
Octet 3	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	
Octet 4	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	

C.5 Safety Code

Size of the Safety Code as shown in Figure C.2 and Figure C.3 can be identified by the

- Parameter "FSP_ProtMode" (see Table A.1), and
- FS I/O structure description (see Table A.1).

Thus, the overall I/O data structure can be identified even if there are non-safety related I/O data associated with the SPDU.

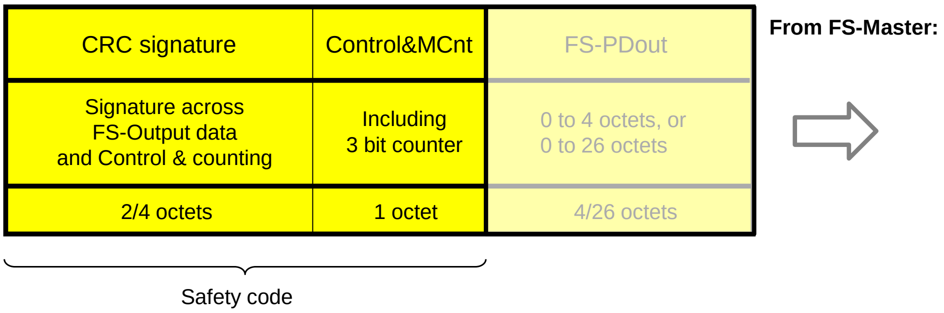


Figure C.2 – Safety Code of an output message

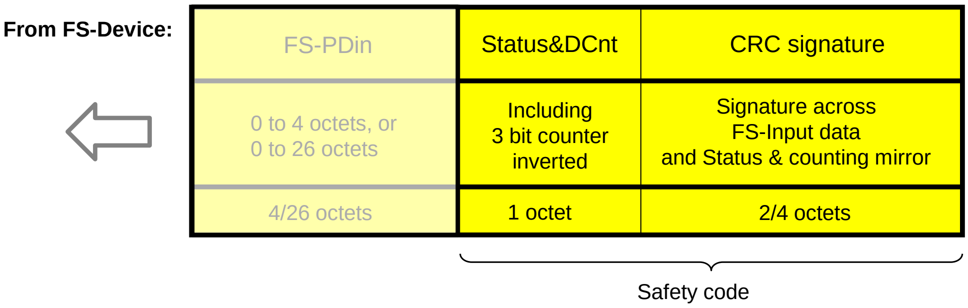


Figure C.3 – Safety Code of an input message

Annex D
(normative, safety related)

CRC generator polynomials

D.1 Overview of CRC generator polynomials

Hamming distance and properness for all required data lengths are important characteristics to select a particular generator polynomial.

If the generator polynomial g(x) = p(x)*(1 + x) is used, where p(x) is a primitive polynomial of degree (r – 1), then the maximum total block length is $2^{(r-1)} - 1$, and the code is able to detect single, double, triple and any odd number of errors (see [19]).

If properness is approved, the residual error probability for the approved data length is 2^{-r}.

It shall be prohibited that the CRC generator polynomial used in the underlying transmission systems, for example IO-Link, matches the CRC generator polynomial used for IO-Link Safety.

Table D.1 shows the CRC-16 and CRC-32 generator polynomials in use for IO-Link Safety:

Table D.1 – CRC generator polynomials for IO-Link Safety

CRC-16/32 polynomial ("Normal" representation)	Data length (bits)	Hamming distance	Properness	Reference	Remark
0x4EAB	≤ 128	≥ 6	≤ 7 octets	[20]	Suitable for functional safety
0xF4ACFB13	≤ 32768	≥ 6	≤128 octets	[20]	
	≤ 65534	≥ 4			
NOTE Representation: "Normal": high order bit omitted					

The CRC-16 can be used

- to secure cyclic Process Data exchange with a total safety PDU length of up to 7 octets, i.e. 4 octets for safety Process Data and
- to secure the transfer of up to 16 octets of FSP parameters at start-up or restart.

The CRC-32 can be used

- to secure cyclic Process Data exchange with a total safety PDU length of up to 32 octets, i.e. 27 octets for safety Process Data and
- to secure the transfer and data integrity of the entire FST parameter set.

Additional parameters and assumptions for the calculation of residual error probabilities/rates can be found in 11.4.6.

D.2 Residual error probabilities

Figure D.1 shows the results of residual error probability (REP) calculations over bit error probabilities (BEP) for safety PDU lengths from 3 to 7 octets.

The REP is less than $0{,}9 \times 10^{-9}$ for BEPs less than the required 10^{-2} at a length of 7 octets.

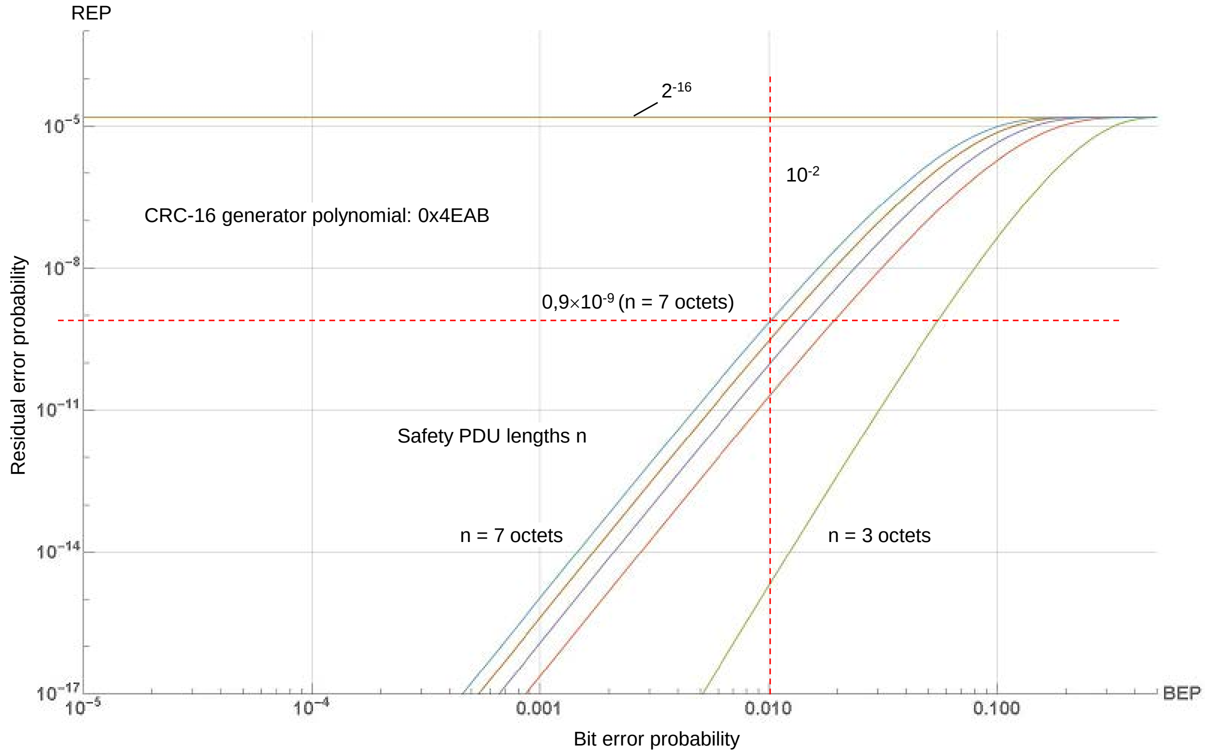


Figure D.1 – CRC-16 generator polynomial

Figure D.2 shows the results of residual error probability (REP) calculations over bit error probabilities (BEP) for safety PDU lengths from 5 to 128 octets.

The REP is less than $0,5 \times 10^{-10}$ for BEPs less than the required 10^{-2} at a length of 26 octets.

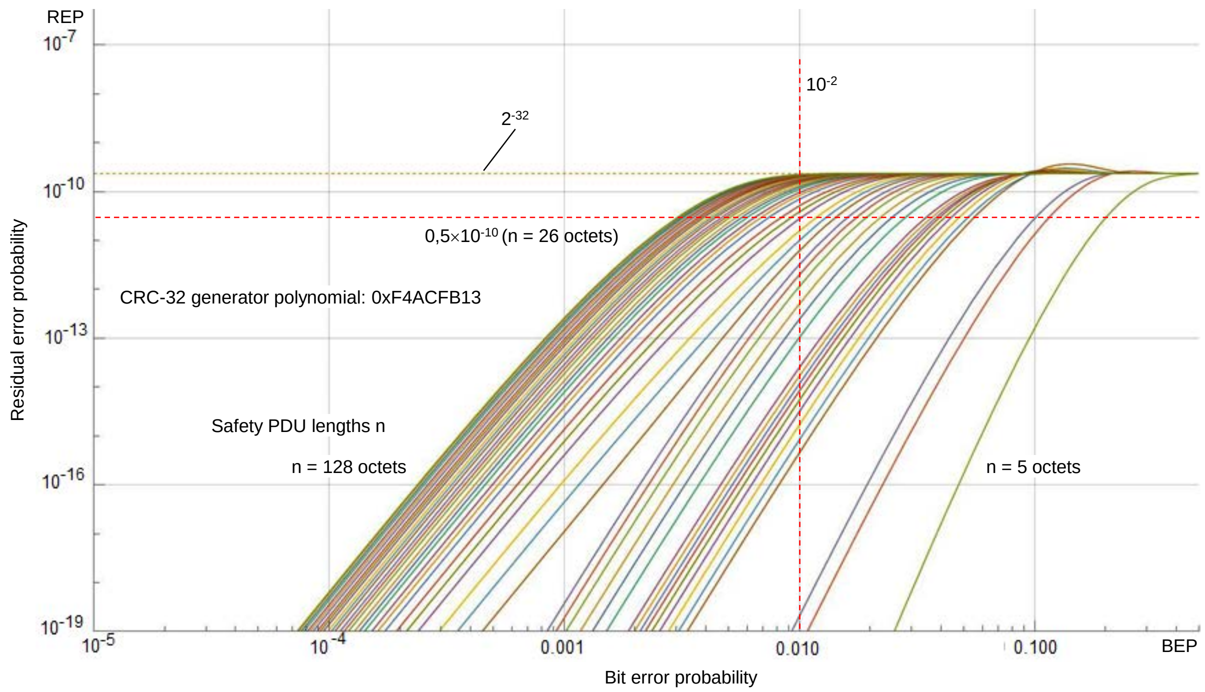


Figure D.2 – CRC-32 generator polynomial

D.3 Implementation considerations

D.3.1 Overview

The designer has two choices to implement the CRC signature calculation. One is based on an algorithm using XOR and shift operations while the other is faster using octet shifts and lookup tables.

D.3.2 Bit shift algorithm (16 bit)

For the 16-bit CRC signature, the value 0x4EAB is used as the generator polynomial. The number of data bits may be odd or even. The value generated after the last octet corresponds to the CRC signature to be transferred.

Figure D.3 shows the algorithm for the innermost loop in "C" programming language.

```
void crc16_calc(unsigned char x, unsigned long *r)
  int i;
  for (i = 1; i <= 8; i++)
    if ((bool)(*r & 0x8000) != (bool)(x & 0x80))
      /* XOR = 1 → shift and process polynomial */
      *r = (*r << 1) ^ 0x4EAB;
      else
      /* XOR = 0 → shift only */
      *r = *r << 1;
    x = x << 1;
  /* for */
```

Figure D.3 – Bit shift algorithm in "C" language (16 bit)

The variables used in Figure D.3 are specified in Table D.2.

Table D.2 – Definition of variables used in Figure D.3

Variable	Definition
x	Data bits including 16 bit CRC signature with "0"
*r	Dereferenced pointer to CRC signature
i	Bitcount 1 to 8

D.3.3 Lookup table (16 bit)

The corresponding function in "C" language is shown in Figure D.4. This function is faster. However, the lookup table requires memory space.

```
r = crctab16 [((r >> 8) ^ *q++) & 0xff] ^(r << 8)
```

Figure D.4 – CRC-16 signature calculation using a lookup table

The variables used in Figure D.4 are specified in Table D.3.

Table D.3 – Definition of variables used in Figure D.4

Variable	Definition
r	CRC signature
q	q represents the pointer to the actual octet value requiring CRC calculation. After reading the value this pointer shall be incremented for the next octet via q++.

The function in Figure D.4 uses the lookup in Table D.4.

Table D.4 – Lookup table for CRC-16 signature calculation

CRC-16 lookup table (0 to 255)							
0000	4EAB	9D56	D3FD	7407	3AAC	E951	A7FA
E80E	A6A5	7558	3BF3	9C09	D2A2	015F	4FF4
9EB7	D01C	03E1	4D4A	EAB0	A41B	77E6	394D
76B9	3812	EBEF	A544	02BE	4C15	9FE8	D143
73C5	3D6E	EE93	A038	07C2	4969	9A94	D43F
9BCB	D560	069D	4836	EFCC	A167	729A	3C31
ED72	A3D9	7024	3E8F	9975	D7DE	0423	4A88
057C	4BD7	982A	D681	717B	3FD0	EC2D	A286
E78A	A921	7ADC	3477	938D	DD26	0EDB	4070
0F84	412F	92D2	DC79	7B83	3528	E6D5	A87E
793D	3796	E46B	AAC0	0D3A	4391	906C	DEC7
9133	DF98	0C65	42CE	E534	AB9F	7862	36C9
944F	DAE4	0919	47B2	E048	AEE3	7D1E	33B5
7C41	32EA	E117	AFBC	0846	46ED	9510	DBBB
0AF8	4453	97AE	D905	7EFF	3054	E3A9	AD02
E2F6	AC5D	7FA0	310B	96F1	D85A	0BA7	450C
81BF	CF14	1CE9	5242	F5B8	BB13	68EE	2645
69B1	271A	F4E7	BA4C	1DB6	531D	80E0	CE4B
1F08	51A3	825E	CCF5	6B0F	25A4	F659	B8F2
F706	B9AD	6A50	24FB	8301	CDAA	1E57	50FC
F27A	BCD1	6F2C	2187	867D	C8D6	1B2B	5580
1A74	54DF	8722	C989	6E73	20D8	F325	BD8E
6CCD	2266	F19B	BF30	18CA	5661	859C	CB37
84C3	CA68	1995	573E	F0C4	BE6F	6D92	2339
6635	289E	FB63	B5C8	1232	5C99	8F64	C1CF
8E3B	C090	136D	5DC6	FA3C	B497	676A	29C1
F882	B629	65D4	2B7F	8C85	C22E	11D3	5F78
108C	5E27	8DDA	C371	648B	2A20	F9DD	B776
15F0	5B5B	88A6	C60D	61F7	2F5C	FCA1	B20A
FDFE	B355	60A8	2E03	89F9	C752	14AF	5A04
8B47	C5EC	1611	58BA	FF40	B1EB	6216	2CBD
6349	2DE2	FE1F	B0B4	174E	59E5	8A18	C4B3

NOTE This table contains 16 bit values in hexadecimal representation for each value (0 to 255) of the argument a in the function crctab16 [a]. The table should be used in ascending order from top left (0) to bottom right (255).

D.3.4 Bit shift algorithm (32 bit)

For the 32-bit CRC signature, the value 0xF4ACFB13 is used as the generator polynomial. The number of data bits may be odd or even. The value generated after the last octet corresponds to the CRC signature to be transferred.

Figure D.5 shows the algorithm for the innermost loop in "C" programming language.

```
void crc32_calc(unsigned char x, unsigned long *r)
  int i;
  for (i = 1; i <= 8; i++)
    if ((bool)(*r & 0x80000000) != (bool)(x & 0x80))
      /* XOR = 1 → shift and process polynomial */
      *r = (*r << 1) ^ 0xF4ACFB13;
      else
      /* XOR = 0 → shift only */
      *r = *r << 1;
    x = x << 1;
  /* for */
```

Figure D.5 – Bit shift algorithm in "C" language (32 bit)

The variables used in Figure D.5 are specified in Table D.5.

Table D.5 – Definition of variables used in Figure D.5

Variable	Definition
x	Data bits including 32 bit CRC signature with "0"
*r	Dereferenced pointer to CRC signature
i	Bit count 1 to 8

D.3.5 Lookup table (32 bit)

The corresponding function in "C" language is shown in Figure D.6. This function is faster. However, the lookup table requires memory space.

```
r = crctab32 [((r >> 24) ^ *q++) & 0xff] ^(r << 8)
```

Figure D.6 – CRC-32 signature calculation using a lookup table

The variables used in Figure D.6 are specified in Table D.6.

Table D.6 – Definition of variables used in Figure D.4

Variable	Definition
r	CRC signature
q	q represents the pointer to the actual octet value requiring CRC calculation. After reading the value this pointer shall be incremented for the next octet via q++.

The function in Figure D.6 uses the lookup table in Table D.7.

Table D.7 – Lookup table for CRC-32 signature calculation

CRC-32 lookup table (0 to 255)							
00000000	F4ACFB13	1DF50D35	E959F626	3BEA1A6A	CF46E179	261F175F	D2B3EC4C
77D434D4	8378CFC7	6A2139E1	9E8DC2F2	4C3E2EBE	B892D5AD	51CB238B	A567D898
EFA869A8	1B0492BB	F25D649D	06F19F8E	D44273C2	20EE88D1	C9B77EF7	3D1B85E4
987C5D7C	6CD0A66F	85895049	7125AB5A	A3964716	573ABC05	BE634A23	4ACFB130
2BFC2843	DF50D350	36092576	C2A5DE65	10163229	E4BAC93A	0DE33F1C	F94FC40F
5C281C97	A884E784	41DD11A2	B571EAB1	67C206FD	936EFDEE	7A370BC8	8E9BF0DB

CRC-32 lookup table (0 to 255)							
C45441EB	30F8BAF8	D9A14CDE	2D0DB7CD	FFBE5B81	0B12A092	E24B56B4	16E7ADA7
B380753F	472C8E2C	AE75780A	5AD98319	886A6F55	7CC69446	959F6260	61339973
57F85086	A354AB95	4A0D5DB3	BEA1A6A0	6C124AEC	98BEB1FF	71E747D9	854BBCCA
202C6452	D4809F41	3DD96967	C9759274	1BC67E38	EF6A852B	0633730D	F29F881E
B850392E	4CFCC23D	A5A5341B	5109CF08	83BA2344	7716D857	9E4F2E71	6AE3D562
CF840DFA	3B28F6E9	D27100CF	26DDFBDC	F46E1790	00C2EC83	E99B1AA5	1D37E1B6
7C0478C5	88A883D6	61F175F0	955D8EE3	47EE62AF	B34299BC	5A1B6F9A	AEB79489
0BD04C11	FF7CB702	16254124	E289BA37	303A567B	C496AD68	2DCF5B4E	D963A05D
93AC116D	6700EA7E	8E591C58	7AF5E74B	A8460B07	5CEAF014	B5B30632	411FFD21
E47825B9	10D4DEAA	F98D288C	0D21D39F	DF923FD3	2B3EC4C0	C26732E6	36CBC9F5
AFF0A10C	5B5C5A1F	B205AC39	46A9572A	941ABB66	60B64075	89EFB653	7D434D40
D82495D8	2C886ECB	C5D198ED	317D63FE	E3CE8FB2	176274A1	FE3B8287	0A977994
4058C8A4	B4F433B7	5DADC591	A9013E82	7BB2D2CE	8F1E29DD	6647DFFB	92EB24E8
378CFC70	C3200763	2A79F145	DED50A56	0C66E61A	F8CA1D09	1193EB2F	E53F103C
840C894F	70A0725C	99F9847A	6D557F69	BFE69325	4B4A6836	A2139E10	56BF6503
F3D8BD9B	07744688	EE2DB0AE	1A814BBD	C832A7F1	3C9E5CE2	D5C7AAC4	216B51D7
6BA4E0E7	9F081BF4	7651EDD2	82FD16C1	504EFA8D	A4E2019E	4DBBF7B8	B9170CAB
1C70D433	E8DC2F20	0185D906	F5292215	279ACE59	D336354A	3A6FC36C	CEC3387F
F808F18A	0CA40A99	E5FDFCBF	115107AC	C3E2EBE0	374E10F3	DE17E6D5	2ABB1DC6
8FDCC55E	7B703E4D	9229C86B	66853378	B436DF34	409A2427	A9C3D201	5D6F2912
17A09822	E30C6331	0A559517	FEF96E04	2C4A8248	D8E6795B	31BF8F7D	C513746E
6074ACF6	94D857E5	7D81A1C3	892D5AD0	5B9EB69C	AF324D8F	466BBBA9	B2C740BA
D3F4D9C9	275822DA	CE01D4FC	3AAD2FEF	E81EC3A3	1CB238B0	F5EBCE96	01473585
A420ED1D	508C160E	B9D5E028	4D791B3B	9FCAF777	6B660C64	823FFA42	76930151
3C5CB061	C8F04B72	21A9BD54	D5054647	07B6AA0B	F31A5118	1A43A73E	EEEF5C2D
4B8884B5	BF247FA6	567D8980	A2D17293	70629EDF	84CE65CC	6D9793EA	993B68F9

NOTE This table contains 32 bit values in hexadecimal representation for each value (0 to 255) of the argument a in the function crctab32 [a]. The table should be used in ascending order from top left (0) to bottom right (255).

D.3.6 Seed values

The algorithm for example in Figure D.3 does not mention explicitly any initial value for the CRC signature variable in "*r". It is implicitly assumed to be "0" by default. This initial value is sometimes called "seed value" in literature.

In 11.4.6 a seed value of "1" is required for the cyclic data exchange of safety PDUs. The reason for that is the possibility for the FS-PDout or FS-PDin data to become completely "0". Since it is a property of CRC-signatures for leading zeros in data strings not to be secured by CRC signatures whenever the seed value is "0", the requirement in 11.4.6 is justified. Any value instead of "0" could be used. However, a "1" is sufficient and faster since all of the operations then are shifting and only the last one consists of shifting and XOR processing.

In A.2.3, A.2.8, A.2.9, A.2.12, and E.5.1, the seed value can be "0" since there are no leading zeros within the data strings.

Annex E
(normative, safety related)

IODD extensions

E.1 General

The IODD of FS-Devices requires extensions for particular FSP parameters and a securing mechanism to protect the content of IODD files from being falsified as mentioned in 11.7.1.

In addition, some of the parameters specified in [1] shall be mandatory instead of optional for this profile (see E.3).

E.2 Schema

There are no extensions required to the existing IODD schema specified in [9].

E.3 IODD constraints

E.3.1 Overview and general rules

Table E.1 shows the constrained Index assignments of data objects (parameters) for IO-Link Safety.

As a general rule, all parameters with Read/Write (R/W) access shall provide a default value within the IODD (for FSP parameters see E.5.2).

Table E.1 – Constrained Index assignment of data objects for IO-Link Safety

Index (dec)	Object name	Access	Length	Data type	M/O/C	Definition/remark
...						
0x0001 (1)	Direct Parameter Page 2	R/W		RecordT	-	Direct Parameter Page 2 shall not be used as well as DirectParameterOverlays
0x0002 (2)	System Command	W	1 octet	UIntegerT	M	Command code definition as specified in B.2.2 in [1] and in E.3.2 in this document
...						
0x000D (13)	Profile Charac-teristic	R	variable	ArrayT of UIntegerT16	M	Profile characteristic as specified in B.2.5 in [1] and in E.3.3 in this document
0x000E (14)	PDInput Descriptor	R	variable	ArrayT of OctetStringT3	M	As specified in B.2.6 in [1] and in E.3.4 in this document
0x000F (15)	PDOutput Descriptor	R	variable	ArrayT of OctetStringT3	M	As specified in B.2.7 in [1] and in E.3.4 in this document
...						
0x0013 (19)	Product ID	R	max. 64 octets	StringT	M	As specified in B.2.11 in [1]
0x0015 (21)	Serial-Number	R	max. 16 octets	StringT	M	As specified in B.2.13 in [1]
0x0016 (22)	Hardware Revision	R	max. 64 octets	StringT	M	As specified in B.2.14 in [1]
0x0017 (23)	Firmware Revision	R	max. 64 octets	StringT	M	As specified in B.2.15 in [1]
0x0018 (24)	Application Specific Tag	R/W	Min. 16, max. 32 octets	StringT	M	As specified in B.2.16 in [1]
...						

Index (dec)	Object name	Access	Length	Data type	M/O/C	Definition/remark
0x0020 (32)	Error Count	R	2 octets	UIntegerT	M	As specified in B.2.17 in [1]
…						
0x0024 (36)	Device Status	R	1 octet	UIntegerT	M	As specified in B.2.18 in [1]
0x0025 (37)	Detailed Device Status	R	variable	ArrayT of OctetStringT3	M	As specified in B.2.19 in [1]
…						
0x0028 (40)	Process-DataInput	R	PD length	Device specific	C	As specified in B.2.20 in [1], if PDin available. See E.3.4.
0x0029 (41)	Process-DataOutput	R	PD length	Device specific	C	As specified in B.2.21 in [1], if PDout available. See E.3.4.
…						
0x4000-0x4FFF (16384-20479)	Profile specific Index					See Table A.1
…						
Key M = mandatory; O = optional; C = conditional						

E.3.2 Specific SystemCommands

Table E.2 shows the specific behavior of the SystemCommand "Restore factory settings" in FS-Devices.

Table E.2 – Specific behavior of "Restore factory settings"

Command (hex)	Command (dec)	Command name	M/O	Definition
…				
0x82	130	Restore factory settings	M	This command shall only be effective whenever the parameter value of FSP_TechParCRC is "0" (commissioning phase)
…				
Key M = mandatory; O = optional				

E.3.3 Profile Characteristic

The ProfileID for IO-Link Safety is 32800 or 0x8020.

E.3.4 ProcessDataInput and ProcessDataOutput

Only the references are required in case of PDin or PDout. This description can be omitted if there is only Safety Code to be transmitted. The sample IODD in E.5.7 shows details.

E.4 IODD conventions

E.4.1 Naming

While this document and [1] use "parameter" for any data object of a Device and FS-Device, IODDs in [9] use "variable" instead and thus all those data objects are indicated via the prefix "V_". The following rules apply:

1) Naming of non-safety parameters shall be "V_xxx". Prefixes "V_FSP", "V_FST" shall be omitted for FS-Devices.

2) Naming of FST technology safety parameters shall be "V_FST_xxx".
3) Naming of FSP safety parameters shall be "V_FSP_xxx".

These namings conventions shall only be used for IO-Link Safety.

E.4.2 Process Data (PD)

The following rules apply for Process Data:

1) PDin and PDout shall be described as record.
2) Subindices shall be used within the records to differentiate between safety PD and non-safety PD.
3) Subindices 1 to 126 shall be used to describe safety PD starting with the highest bit offset.
4) Safety Code (see C.5) shall not be described in detail within the IODD. However, Subindex 127 shall be used to describe the Safety Code by means of an OctetStringT (3 or 5 octets) as a dummy to indicate the length of the Safety Code.
5) Subindices 128 to 255 shall be used to describe non-safety PD.
6) Multiple PD structure definitions selected by conditions are not permitted.

E.4.3 IODD conventions for user interface

The following rules apply for user interface:

1) The IODD shall contain different headlines (menu IDs) for the parameter block types "Standard", "FST", and "FSP" in this order.
2) The following abbreviations shall be used for the user role menu IDs: Observer ("OR"), Maintenance ("MR"), Specialist ("SR").

   The menu IDs shall be structured and named as follows:
   "ME_OR_Param_Standard"
   "ME_MR_Param_Standard"
   "ME_SR_Param_Standard"
   "ME_OR_Param_FST"
   "ME_MR_Param_FST"
   "ME_SR_Param_FST"
   "ME_OR_Param_FSP"
   "ME_MR_Param_FSP"
   "ME_SR_Param_FSP"

   Menus can be omitted for example in case of the observer role. They can be referenced multiple if for example the same menu is used for the maintenance role and the specialist role.

E.4.4 Master Tool features

The following rules on how to present the IODD to the user are highly recommended:

1) IODD interpreter in Master Tools should show headlines not only for PDin and PDout but also for safety and non-safety PD. These headlines should use yellow colors.
2) In case of PD observation via ISDU access the variable names should be the same as with cyclic PDs.

E.5 Securing

E.5.1 General

An IODD-based non-safety viewer calculates this 32 bit CRC signature across the FSP parameter description within the IODD. The algorithm for the calculation is shown in this Annex. The safety-related interpreter of the FS-Master Tool checks the correctness of the imported IODD data. Parameter names associated to Index/Subindex are known in the FS IODD interpreter and can be checked in a safe manner.

An IODD checker is not safety-related and thus not sufficient.

Only one IODD per DeviceID is permitted. A particular FS-Device (hardware) can have two DeviceIDs for example a current DeviceID and a DeviceID of a previous software version.

Figure E.1 shows the algorithm to build the FSP_ParamDescCRC signature. The algorithm shall be used across the Authenticity and the Protocol block (see Table A.1). A seed value "0" shall be used (see D.3.6).

1. General rule: All numerical values are serialized in **big-endian octet order** (most significant octet first).
2. Serialize the **Index** (16 bit unsigned integer) of the FS parameter.
3. Serialize the **bitLength** (16 bit unsigned integer) of the RecordT.
4. Sort the RecordItems in ascending order by Subindex.
5. For each **RecordItem** (including the last one) serialize:
   a) The **Subindex** (8 bit unsigned integer)
   b) The **bitOffset** (16 bit unsigned integer)
   c) The **Datatype** (8 bit unsigned integer): 1=UIntegerT(8), 2=UIntegerT(16), 3=UIntegerT(32)
   d) If and only if a **DefaultValue** is given in the IODD: The DefaultValue (8/16/32 bit unsigned integer according to Datatype).
   e) If and only if **SingleValues** or a **ValueRange** is given in the IODD: The allowed values. A list of SingleValues is serialized as a sequence of these values, in ascending order. A ValueRange is serialized by the sequence of the minimum and maximum value. Whether SingleValues and/or a ValueRange are allowed depends on the specific RecordItem. See Table E.4.
6. Calculate the 2 octet CRC across the octet stream using the CRC polynomial 0x4EAB.

Figure E.1 – Algorithm to build the FSP parameter CRC signatures

E.5.2 DefaultValues for FSP

The DefaultValues for FSP_Authenticity1/2, FSP_Port, FSP_AuthentCRC, FSP_TechParCRC, and FSP_ProtParCRC shall be "0". Table E.3 demonstrates the user actions to replace the default values by actual values.

Table E.3 – User actions to replace DefaultValues

Parameter	User actions
FSP_Authenticity1/2	During commissioning, the Authenticity values can be acquired from the gateway and displayed by the Master Tool. SCL will not start with the default value.
FSP_Port	The user shall replace the default "0" by an allowed number with the help of the Master Tool during commissioning. SCL will not start with the default value.
FSP_AuthentCRC	Master Tool calculates this CRC signature

Parameter	User actions
FSP_TechParCRC	The user parameterizes the FS-Device during commissioning or maintenance and uses a Dedicated Tool to calculate the actual value (see 11.7.8 and 11.7.9)
FSP_ProtParCRC	Master Tool calculates this CRC signature

E.5.3 FSP_Authenticity

The values of the authenticity parameters cannot be defined within the IODD. They are maintained by the FS-Master Tool.

E.5.4 FSP_Protocol

The limited variability of the protocol parameters requires the securing mechanism specified in E.5.1.

Table E.4 lists the RecordItems of FSP_Protocol to be serialized.

Table E.4 – RecordItems of FSP_Protocol where allowed values shall be serialized

RecordItem	Serialized as
FSP_ProtVersion	List of 8-bit unsigned integer containing the allowed values, in ascending order
FSP_ProtMode	List of 8-bit unsigned integer containing the allowed values, in ascending order
FSP_Watchdog	Minimum value and maximum value of the contiguous range of allowed values
Any other	All values according to the data type are allowed, therefore nothing is serialized

E.5.5 FSP_IO_Description

The FSP_IO_Description parameters do not need a particular securing mechanism since these instance values are straight forward. The IODD designer can calculate the CRC signature already and place it into the IODD (see A.2.9).

E.5.6 Sample serialization for FSP_ParamDescCRC

Table E.5 shows a sample serialization for the calculation of the FSP_ParamDescCRC signature in E.5.7. A seed value "0" shall be used since there are no leading zeros (see D.3.6).

Table E.5 – Sample serialization for FSP_ParamDescCRC

Offset	Serialization	IODD items	Expected values
0000	42 00	index	42 00 *(≠ 0)*
0002	00 58	bitLength of index	00 58
0004	01	subindex	01 *(Authenticity 1)*
0005	00 38	bitOffset	00 38
0007	03	xsi:type=UIntegerT, bitLength=32	03
0008	00 00 00 00	RecordItemInfo/@defaultValue	00 00 00 00
000C	02	subindex	02 *(Authenticity 2)*
000D	00 18	bitOffset	00 18
000F	03	xsi:type=UIntegerT, bitLength=32	03
0010	00 00 00 00	RecordItemInfo/@defaultValue	00 00 00 00
0014	03	subindex	03 *(Port)*
0015	00 10	bitOffset	00 10
0017	01	xsi:type=UIntegerT, bitLength=8	01
0018	00	RecordItemInfo/@defaultValue	00
0019	04	subindex	04 *(AuthentCRC)*

Offset	Serialization	IODD items	Expected values
001A	00 00	bitOffset	00 00
001C	02	xsi:type=UIntegerT, bitLength=16	02
001D	00 00	RecordItemInfo/@defaultValue	00 00 *(Dummy CRC)*
001F	42 01	index	42 01
0021	00 60	bitLength of index	00 60
0023	01	subindex	01 *(ProtVersion)*
0024	00 58	bitOffset	00 58
0026	01	xsi:type=UIntegerT, bitLength=8	01
0027	01	RecordItemInfo/@defaultValue	01
0028	01	SingleValue/@value	01 *(example:16 bit)*
0029	02	subindex	02 *(ProtMode)*
002A	00 50	bitOffset	00 50
002C	01	xsi:type=UIntegerT, bitLength=8	01
002D	01	RecordItemInfo/@defaultValue	*Vendor defined*
002E	01	SingleValue/@value	01
002F	03	subindex	03 *(Watchdog)*
0030	00 40	bitOffset	00 40
0032	02	xsi:type=UIntegerT, bitLength=16	02
0033	00 64	RecordItemInfo/@defaultValue	*(Vendor defined)*
0035	00 64	ValueRange/@lowerValue	00 64 *(example: 100)*
0037	13 88	ValueRange/@upperValue	13 88 *(example: 5000)*
0039	04	subindex	04 *(IO_StructCRC)*
003A	00 30	bitOffset	00 30
003C	02	xsi:type=UIntegerT, bitLength=16	02
003D	09 52	RecordItemInfo/@defaultValue (see A.2.9)	*(Vendor defined)*
003F	05	Subindex	05 *(TechParCRC)*
0040	00 10	bitOffset	00 10
0042	03	xsi:type=UIntegerT, bitLength=32	03
0043	00 00 00 00	RecordItemInfo/@defaultValue	00 00 00 00 *(Vendor)*
0047	06	subindex	06 *(ProtParCRC)*
0048	00 00	bitOffset	00 00
004A	02	xsi:type=UIntegerT, bitLength=16	02
004B	00 00	RecordItemInfo/@defaultValue	00 00 *Dummy CRC*
Calculated FSP_ParamDescCRC signature value is: 7520 (0x1D60)			See E.5.7

The sample serialization in Table E.5 shows 77 octets to be secured via the CRC-16 polynomial listed in Table D.1. This is only sufficient due to the fact that most of the values are expected values within the FS-Master Tool importing the IODD. Only a few values are variable and "*Vendor defined*" and require securing (see offsets: 0028, 002D,0033 to 0037, 003D, and 0043). The remaining values can be compared with preset values.

The "*Dummy CRC*" are placeholders to be replaced by the FS-Master Tool once the user assigned the actual parameter values.

E.5.7 FST and FSP parameters and Data Storage

FST parameters shall be described within the IODD. A "packed" parameter transfer via one ISDU that is not described within the IODD is possible for Data Storage as long as the result in the Device/FS-Device is the same as with discrete ISDUs (see 11.7.6). A manufacturer-/vendor is responsible to guarantee this behavior.

FSP parameters (authenticity and protocol) shall be described within the IODD also and are part of Data Storage.

E.5.8 Sample IODD of an FS-Device

The following XML code represents the sample code of an FS-Device IODD. A complete IODD file with name *IO-Link-SafetyDevice-20170225-IODD1.1.xml* can be downloaded from the IO-Link websites.

This sample IODD contains already calculated CRC signature values:

```
<?xml version="1.0" encoding="UTF-8"?>
<IODevice xmlns="http://www.io-link.com/IODD/2010/10" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.io-link.com/IODD/2010/10 IODD1.1.xsd">
  <DocumentInfo copyright="IO-Link Community" releaseDate="2017-02-25" version="V1.0"/>
  <ProfileHeader>
    <ProfileIdentification>IO Device Profile</ProfileIdentification>
    <ProfileRevision>1.1</ProfileRevision>
    <ProfileName>Device Profile for IO Devices</ProfileName>
    <ProfileSource>IO-Link Consortium</ProfileSource>
    <ProfileClassID>Device</ProfileClassID>
    <ISO15745Reference>
      <ISO15745Part>1</ISO15745Part>
      <ISO15745Edition>1</ISO15745Edition>
      <ProfileTechnology>IODD</ProfileTechnology>
    </ISO15745Reference>
  </ProfileHeader>
  <ProfileBody>
    <DeviceIdentity deviceId="160" vendorId="65535" vendorName="IO-Link Community">
      <VendorText textId="T_VendorText"/>
      <VendorUrl textId="T_VendorUrl"/>
      <VendorLogo name="IO-Link-logo.png"/>
      <DeviceName textId="T_DeviceName"/>
      <DeviceFamily textId="T_DeviceFamily"/>
      <DeviceVariantCollection>
        <DeviceVariant productId="SafetyDeviceVariant" deviceIcon="IO-Link-SafetyDevice-icon.png" deviceSymbol="IO-Link-SafetyDevice-pic.png">
          <Name textId="TN_SafetyDeviceVariant"/>
          <Description textId="TD_SafetyDeviceVariant"/>
        </DeviceVariant>
      </DeviceVariantCollection>
    </DeviceIdentity>
    <DeviceFunction>
      <Features blockParameter="true" dataStorage="true" profileCharacteristic="32800">
        <SupportedAccessLocks parameter="true" dataStorage="true" localParameterization="false" localUserInterface="false"/>
      </Features>
      <DatatypeCollection>
        <!-- Data types for IO-Link Safety parameter. See chapter A.1. -->
        <Datatype id="D_FSP_Authenticity" xsi:type="RecordT" bitLength="88">
          <RecordItem subindex="1" bitOffset="56">
            <SimpleDatatype xsi:type="UIntegerT" bitLength="32"/>
            <Name textId="TN_FSCP_Authenticity_1"/>
            <Description textId="TD_FSCP_Authenticity_1"/>
          </RecordItem>
          <RecordItem subindex="2" bitOffset="24">
            <SimpleDatatype xsi:type="UIntegerT" bitLength="32"/>
            <Name textId="TN_FSCP_Authenticity_2"/>
            <Description textId="TD_FSCP_Authenticity_2"/>
          </RecordItem>
          <RecordItem subindex="3" bitOffset="16">
            <SimpleDatatype xsi:type="UIntegerT" bitLength="8"/>
            <Name textId="TN_FSP_Port"/>
            <Description textId="TD_FSP_Port"/>
          </RecordItem>
          <RecordItem subindex="4" bitOffset="0">
            <SimpleDatatype xsi:type="UIntegerT" bitLength="16"/>
            <Name textId="TN_FSP_AuthentCRC"/>
            <Description textId="TD_FSP_AuthentCRC"/>
```

```xml
      </RecordItem>
    </Datatype>
    <Datatype id="D_FSP_Password" xsi:type="StringT" fixedLength="32" encoding="US-ASCII"/>
  </DatatypeCollection>
  <VariableCollection>
    <StdVariableRef id="V_DirectParameters_1"/>
    <!--0-->
    <StdVariableRef id="V_DirectParameters_2"/>
    <!--1 - TBD delete this once IODD Checker has been changed -->
    <StdVariableRef id="V_SystemCommand">
      <!--2-->
      <StdSingleValueRef value="130"/>
      <!-- RestoreFactorySettings -->
    </StdVariableRef>
    <StdVariableRef id="V_DeviceAccessLocks">
      <!--12-->
      <StdRecordItemRef subindex="1" defaultValue="false"/>
      <!-- TBD delete this once IODD Checker has been changed -->
      <StdRecordItemRef subindex="2" defaultValue="false"/>
      <!-- TBD delete this once IODD Checker has been changed -->
    </StdVariableRef>
    <StdVariableRef id="V_VendorName" defaultValue="IO-Link Community"/>
    <!--16-->
    <StdVariableRef id="V_VendorText" defaultValue="http://www.io-link.com"/>
    <!--17 optional -->
    <StdVariableRef id="V_ProductName" defaultValue="SafetyDevice"/>
    <!--18-->
    <StdVariableRef id="V_ProductID" defaultValue="SafetyDeviceVariant"/>
    <!--19-->
    <StdVariableRef id="V_ProductText" defaultValue="Sample IO-Link Safety"/>
    <!--20 optional -->
    <StdVariableRef id="V_SerialNumber"/>
    <!--21-->
    <StdVariableRef id="V_HardwareRevision"/>
    <!--22-->
    <StdVariableRef id="V_FirmwareRevision"/>
    <!--23-->
    <StdVariableRef id="V_ApplicationSpecificTag" defaultValue="IO-Link Safety"/>
    <!--24-->
    <StdVariableRef id="V_ErrorCount"/>
    <!--32-->
    <StdVariableRef id="V_DeviceStatus"/>
    <!--36-->
    <StdVariableRef id="V_DetailedDeviceStatus" fixedLengthRestriction="8"/>
    <!--37-->
    <StdVariableRef id="V_ProcessDataInput"/>
    <!--40-->
    <!-- V_ProcessDataOutput 41 - only required when "real" output is present (not only F message trailer) -->
    <!-- standard (=non-safety) Parameter appear here, e.g. -->
    <Variable index="64" id="V_NonSafetyParameter" accessRights="rw">
      <Datatype xsi:type="IntegerT" bitLength="16"/>
      <Name textId="TN_NonSafetyParameter"/>
    </Variable>
    <!-- FS Technology Parameter appear here, e.g. -->
    <Variable index="65" id="V_FST_DiscrepancyTime" accessRights="rw" defaultValue="0">
      <Datatype xsi:type="UIntegerT" bitLength="16"/>
      <Name textId="TN_FST_DiscrepancyTime"/>
    </Variable>
    <Variable index="66" id="V_FST_Filter" accessRights="rw" defaultValue="0">
      <Datatype xsi:type="UIntegerT" bitLength="16"/>
      <Name textId="TN_FST_Filter"/>
    </Variable>
    <!-- IO-Link Safety parameter. See chapter A.1. -->
    <Variable index="16896" id="V_FSP_Authenticity" accessRights="rw">
      <DatatypeRef datatypeId="D_FSP_Authenticity"/>
      <RecordItemInfo subindex="1" defaultValue="0"/>
      <!-- FSCP_Authenticity_1: 0= invalid -->
      <RecordItemInfo subindex="2" defaultValue="0"/>
      <!-- FSCP_Authenticity_2: 0= invalid -->
      <RecordItemInfo subindex="3" defaultValue="0"/>
      <!-- FSP_Port: 0= invalid -->
      <RecordItemInfo subindex="4" defaultValue="0"/>
      <!-- FSP_AuthentCRC: 0= invalid -->
      <Name textId="TN_FSP_Authenticity"/>
      <Description textId="TD_FSP_Authenticity"/>
    </Variable>
    <Variable index="16897" id="V_FSP_Protocol" accessRights="rw">
```

```xml
    <Datatype xsi:type="RecordT" bitLength="96">
      <RecordItem subindex="1" bitOffset="88">
        <SimpleDatatype xsi:type="UIntegerT" bitLength="8">
          <SingleValue value="0"/>
          <!-- fixed - current protocol version -->
        </SimpleDatatype>
        <Name textId="TN_FSP_ProtVer"/>
        <Description textId="TD_FSP_ProtVer"/>
      </RecordItem>
      <RecordItem subindex="2" bitOffset="80">
        <SimpleDatatype xsi:type="UIntegerT" bitLength="8">
          <!-- Which of these SingleValues is supported is device specific -->
          <SingleValue value="1"/>
          <!-- 16 bit CRC -->
          <!-- SingleValue value="2" - 32 bit CRC -->
        </SimpleDatatype>
        <Name textId="TN_FSP_ProtMode"/>
        <Description textId="TD_FSP_ProtMode"/>
      </RecordItem>
      <RecordItem subindex="3" bitOffset="64">
        <SimpleDatatype xsi:type="UIntegerT" bitLength="16">
          <!-- Which ValueRange is supported is device specific (but the lowerValue must be >0) -->
          <ValueRange lowerValue="100" upperValue="5000"/>
        </SimpleDatatype>
        <Name textId="TN_FSP_Watchdog"/>
        <Description textId="TD_FSP_Watchdog"/>
      </RecordItem>
      <RecordItem subindex="4" bitOffset="48">
        <SimpleDatatype xsi:type="UIntegerT" bitLength="16"/>
        <Name textId="TN_FSP_IO_StructCRC"/>
        <Description textId="TD_FSP_IO_StructCRC"/>
      </RecordItem>
      <RecordItem subindex="5" bitOffset="16">
        <SimpleDatatype xsi:type="UIntegerT" bitLength="32"/>
        <Name textId="TN_FSP_TechParCRC"/>
        <Description textId="TD_FSP_TechParCRC"/>
      </RecordItem>
      <RecordItem subindex="6" bitOffset="0">
        <SimpleDatatype xsi:type="UIntegerT" bitLength="16"/>
        <Name textId="TN_FSP_ProtParCRC"/>
        <Description textId="TD_FSP_ProtParCRC"/>
      </RecordItem>
    </Datatype>
    <RecordItemInfo subindex="1" defaultValue="0"/>
    <!-- FSP_ProtVer: 0= valid -->
    <RecordItemInfo subindex="2" defaultValue="1"/>
    <!-- FSP_ProtMode:1 (16 bit CRC)= valid -->
    <RecordItemInfo subindex="3" defaultValue="100"/>
    <!-- FSP_Watchdog: 100= valid -->
    <RecordItemInfo subindex="4" defaultValue="444"/>
    <!-- FSP_IO_StructCRC: = valid -->
    <!-- TBD value -->
    <RecordItemInfo subindex="5" defaultValue="0"/>
    <!-- FSP_TechParCRC: 0= invalid -->
    <RecordItemInfo subindex="6" defaultValue="0"/>
    <!-- FSP_ProtParCRC: 0= invalid -->
    <Name textId="TN_FSP_Protocol"/>
    <Description textId="TD_FSP_Protocol"/>
  </Variable>
  <Variable index="16912" id="V_FSP_Password" accessRights="wo">
    <DatatypeRef datatypeId="D_FSP_Password"/>
    <Name textId="TN_FSP_Password"/>
    <Description textId="TD_FSP_Password"/>
  </Variable>
  <Variable index="16913" id="V_FSP_Reset_Password" accessRights="wo">
    <DatatypeRef datatypeId="D_FSP_Password"/>
    <Name textId="TN_FSP_Reset_Password"/>
    <Description textId="TD_FSP_Reset_Password"/>
  </Variable>
  <Variable index="16914" id="V_FSP_ParamDescCRC" accessRights="ro" defaultValue="444">
    <!-- TBD: correct defaultValue -->
    <Datatype xsi:type="UIntegerT" bitLength="16"/>
    <Name textId="TN_FSP_ParamDescCRC"/>
    <Description textId="TD_FSP_ParamDescCRC"/>
  </Variable>
</VariableCollection>
<ProcessDataCollection>
```

```xml
<!-- See chapter 11.4.3 Safety PDUs, Figure A.65 and Figure A.66 -->
<ProcessData id="P_ProcessData">
  <ProcessDataIn bitLength="88" id="PI_ProcessDataIn">
    <!-- Safety process data has subindex 1..126 -->
    <Datatype xsi:type="RecordT" bitLength="88">
      <!-- boolean octet 1 -->
      <RecordItem subindex="1" bitOffset="80">
        <SimpleDatatype xsi:type="BooleanT"/>
        <Name textId="TN_PDin-Bool1"/>
      </RecordItem>
      <RecordItem subindex="2" bitOffset="81">
        <SimpleDatatype xsi:type="BooleanT"/>
        <Name textId="TN_PDin-Bool2"/>
      </RecordItem>
      <RecordItem subindex="3" bitOffset="82">
        <SimpleDatatype xsi:type="BooleanT"/>
        <Name textId="TN_PDin-Bool3"/>
      </RecordItem>
      <RecordItem subindex="4" bitOffset="83">
        <SimpleDatatype xsi:type="BooleanT"/>
        <Name textId="TN_PDin-Bool4"/>
      </RecordItem>
      <RecordItem subindex="5" bitOffset="84">
        <SimpleDatatype xsi:type="BooleanT"/>
        <Name textId="TN_PDin-Bool5"/>
      </RecordItem>
      <RecordItem subindex="6" bitOffset="85">
        <SimpleDatatype xsi:type="BooleanT"/>
        <Name textId="TN_PDin-Bool6"/>
      </RecordItem>
      <RecordItem subindex="7" bitOffset="86">
        <SimpleDatatype xsi:type="BooleanT"/>
        <Name textId="TN_PDin-Bool7"/>
      </RecordItem>
      <RecordItem subindex="8" bitOffset="87">
        <SimpleDatatype xsi:type="BooleanT"/>
        <Name textId="TN_PDin-Bool8"/>
      </RecordItem>
      <!-- boolean octet 2 -->
      <!-- There may be no gaps between the booleans, but the last octet may contain less than eight booleans. -->
      <RecordItem subindex="9" bitOffset="72">
        <SimpleDatatype xsi:type="BooleanT"/>
        <Name textId="TN_PDin-Bool9"/>
      </RecordItem>
      <RecordItem subindex="10" bitOffset="73">
        <SimpleDatatype xsi:type="BooleanT"/>
        <Name textId="TN_PDin-Bool10"/>
      </RecordItem>
      <RecordItem subindex="11" bitOffset="74">
        <SimpleDatatype xsi:type="BooleanT"/>
        <Name textId="TN_PDin-Bool11"/>
      </RecordItem>
      <RecordItem subindex="12" bitOffset="75">
        <SimpleDatatype xsi:type="BooleanT"/>
        <Name textId="TN_PDin-Bool12"/>
      </RecordItem>
      <RecordItem subindex="13" bitOffset="76">
        <SimpleDatatype xsi:type="BooleanT"/>
        <Name textId="TN_PDin-Bool13"/>
      </RecordItem>
      <!-- Integer (octets 3 and 4) -->
      <RecordItem subindex="14" bitOffset="56">
        <SimpleDatatype xsi:type="IntegerT" bitLength="16"/>
        <Name textId="TN_PDin-Int1"/>
      </RecordItem>
      <!-- Status&DCnt and CRC has fixed subindex 127, octets 5-7 -->
      <RecordItem subindex="127" bitOffset="32">
        <SimpleDatatype xsi:type="OctetStringT" fixedLength="3"/>
        <Name textId="TN_PD_FSTrailer"/>
        <Description textId="TD_PD_FSTrailer"/>
      </RecordItem>
      <!-- Non-safety process data has subindex 128..255 -->
      <!-- UInteger (octets 8-11) -->
      <RecordItem subindex="128" bitOffset="0">
        <SimpleDatatype xsi:type="UIntegerT" bitLength="32"/>
        <Name textId="TN_PD_Rev"/>
        <Description textId="TD_PD_Rev"/>
```

```xml
        </RecordItem>
      </Datatype>
      <Name textId="TN_ProcessDataIn"/>
    </ProcessDataIn>
    <ProcessDataOut bitLength="24" id="PO_ProcessDataOut">
      <Datatype xsi:type="RecordT" bitLength="24">
        <!-- Control&MCnt and CRC -->
        <RecordItem subindex="1" bitOffset="0">
          <SimpleDatatype xsi:type="OctetStringT" fixedLength="3"/>
          <Name textId="TN_PD_FSTrailer"/>
          <Description textId="TD_PD_FSTrailer"/>
        </RecordItem>
      </Datatype>
      <Name textId="TN_ProcessDataOut"/>
    </ProcessDataOut>
  </ProcessData>
</ProcessDataCollection>
<EventCollection>
  <!-- SCL (Safety Communication Layer) EventCodes. See chapter B.1. -->
  <Event code="45056" type="Warning">
    <Name textId="TN_TransmissionError_CRCSignature"/>
  </Event>
  <Event code="45057" type="Warning">
    <Name textId="TN_TransmissionError_Counter"/>
  </Event>
  <Event code="45058" type="Error">
    <Name textId="TN_TransmissionError_Timeout"/>
  </Event>
  <Event code="45059" type="Error">
    <Name textId="TN_UnexpectedAuthenticationCode"/>
  </Event>
  <Event code="45060" type="Error">
    <Name textId="TN_UnexpectedAuthenticationPort"/>
  </Event>
  <Event code="45061" type="Error">
    <Name textId="TN_IncorrectFSP_AuthentCRC"/>
  </Event>
  <Event code="45062" type="Error">
    <Name textId="TN_IncorrectFSP_ProtParCRC"/>
  </Event>
  <Event code="45063" type="Error">
    <Name textId="TN_IncorrectFSP_TechParCRC"/>
  </Event>
  <Event code="45064" type="Error">
    <Name textId="TN_IncorrectFSP_IO_StructCRC"/>
  </Event>
  <Event code="45065" type="Error">
    <Name textId="TN_WatchdogTimeOutOfSpec"/>
  </Event>
  <Event code="6200" type="Error">
    <!-- for device test -->
    <Name textId="TN_Event1"/>
  </Event>
  <Event code="6201" type="Error">
    <!-- for device test -->
    <Name textId="TN_Event2"/>
  </Event>
</EventCollection>
<UserInterface>
  <MenuCollection>
    <Menu id="M_Identification">
      <VariableRef variableId="V_VendorName"/>
      <VariableRef variableId="V_VendorText"/>
      <VariableRef variableId="V_ProductName"/>
      <VariableRef variableId="V_ProductID"/>
      <VariableRef variableId="V_ProductText"/>
      <VariableRef variableId="V_SerialNumber"/>
      <VariableRef variableId="V_HardwareRevision"/>
      <VariableRef variableId="V_FirmwareRevision"/>
    </Menu>
    <Menu id="M_OR_Parameter">
      <RecordItemRef variableId="V_DeviceAccessLocks" subindex="1" accessRightRestriction="ro"/>
      <VariableRef variableId="V_ApplicationSpecificTag"/>
      <VariableRef variableId="V_NonSafetyParameter" accessRightRestriction="ro"/>
    </Menu>
    <Menu id="M_MR_Param_Standard">
      <Name textId="TN_MR_Param_Standard"/>
```

```xml
      <RecordItemRef variableId="V_DeviceAccessLocks" subindex="1"/>
      <VariableRef variableId="V_ApplicationSpecificTag"/>
      <VariableRef variableId="V_NonSafetyParameter"/>
    </Menu>
    <Menu id="M_MR_Param_FST">
      <Name textId="TN_MR_Param_FST"/>
      <VariableRef variableId="V_FST_DiscrepancyTime" unitCode="1056" accessRightRestriction="ro"/>
      <VariableRef variableId="V_FST_Filter" unitCode="1056" accessRightRestriction="ro"/>
    </Menu>
    <Menu id="M_MR_Param_FSP">
      <Name textId="TN_MR_Param_FSP"/>
      <VariableRef variableId="V_FSP_Authenticity" accessRightRestriction="ro"/>
      <VariableRef variableId="V_FSP_Protocol" accessRightRestriction="ro"/>
    </Menu>
    <Menu id="M_SR_Param_Standard">
      <Name textId="TN_SR_Param_Standard"/>
      <RecordItemRef variableId="V_DeviceAccessLocks" subindex="1"/>
      <VariableRef variableId="V_ApplicationSpecificTag"/>
      <VariableRef variableId="V_NonSafetyParameter"/>
    </Menu>
    <Menu id="M_SR_Param_FST">
      <Name textId="TN_SR_Param_FST"/>
      <VariableRef variableId="V_FST_DiscrepancyTime" unitCode="1056"/>
      <VariableRef variableId="V_FST_Filter" unitCode="1056"/>
    </Menu>
    <Menu id="M_SR_Param_FSP">
      <Name textId="TN_SR_Param_FSP"/>
      <VariableRef variableId="V_FSP_Authenticity"/>
      <VariableRef variableId="V_FSP_Protocol"/>
      <VariableRef variableId="V_FSP_Password"/>
      <VariableRef variableId="V_FSP_Reset_Password"/>
    </Menu>
    <Menu id="M_MR_Parameter">
      <MenuRef menuId="M_MR_Param_Standard"/>
      <MenuRef menuId="M_MR_Param_FST"/>
      <MenuRef menuId="M_MR_Param_FSP"/>
    </Menu>
    <Menu id="M_SR_Parameter">
      <MenuRef menuId="M_SR_Param_Standard"/>
      <MenuRef menuId="M_SR_Param_FST"/>
      <MenuRef menuId="M_SR_Param_FSP"/>
    </Menu>
    <Menu id="M_StandardProcessData">
      <Name textId="TN_StandardProcessData"/>
      <RecordItemRef variableId="V_ProcessDataInput" subindex="128"/>
    </Menu>
    <Menu id="M_FS_ProcessData">
      <Name textId="TN_FS_ProcessData"/>
      <RecordItemRef variableId="V_ProcessDataInput" subindex="1"/>
      <RecordItemRef variableId="V_ProcessDataInput" subindex="2"/>
      <RecordItemRef variableId="V_ProcessDataInput" subindex="3"/>
      <RecordItemRef variableId="V_ProcessDataInput" subindex="4"/>
      <RecordItemRef variableId="V_ProcessDataInput" subindex="5"/>
      <RecordItemRef variableId="V_ProcessDataInput" subindex="6"/>
      <RecordItemRef variableId="V_ProcessDataInput" subindex="7"/>
      <RecordItemRef variableId="V_ProcessDataInput" subindex="8"/>
      <RecordItemRef variableId="V_ProcessDataInput" subindex="9"/>
      <RecordItemRef variableId="V_ProcessDataInput" subindex="10"/>
      <RecordItemRef variableId="V_ProcessDataInput" subindex="11"/>
      <RecordItemRef variableId="V_ProcessDataInput" subindex="12"/>
      <RecordItemRef variableId="V_ProcessDataInput" subindex="13"/>
      <RecordItemRef variableId="V_ProcessDataInput" subindex="14"/>
    </Menu>
    <Menu id="M_Observation">
      <MenuRef menuId="M_StandardProcessData"/>
      <MenuRef menuId="M_FS_ProcessData"/>
    </Menu>
    <Menu id="M_Diagnosis">
      <VariableRef variableId="V_ErrorCount"/>
      <VariableRef variableId="V_DeviceStatus"/>
      <VariableRef variableId="V_DetailedDeviceStatus"/>
    </Menu>
  </MenuCollection>
  <ObserverRoleMenuSet>
    <IdentificationMenu menuId="M_Identification"/>
    <ParameterMenu menuId="M_OR_Parameter"/>
    <ObservationMenu menuId="M_Observation"/>
```

```xml
        <DiagnosisMenu menuId="M_Diagnosis"/>
      </ObserverRoleMenuSet>
      <MaintenanceRoleMenuSet>
        <IdentificationMenu menuId="M_Identification"/>
        <ParameterMenu menuId="M_MR_Parameter"/>
        <ObservationMenu menuId="M_Observation"/>
        <DiagnosisMenu menuId="M_Diagnosis"/>
      </MaintenanceRoleMenuSet>
      <SpecialistRoleMenuSet>
        <IdentificationMenu menuId="M_Identification"/>
        <ParameterMenu menuId="M_SR_Parameter"/>
        <ObservationMenu menuId="M_Observation"/>
        <DiagnosisMenu menuId="M_Diagnosis"/>
      </SpecialistRoleMenuSet>
    </UserInterface>
  </DeviceFunction>
</ProfileBody>
<CommNetworkProfile xsi:type="IOLinkCommNetworkProfileT" iolinkRevision="V1.1">
  <TransportLayers>
    <PhysicalLayer bitrate="COM3" minCycleTime="2000" sioSupported="true" mSequenceCapability="43">
      <Connection xsi:type="M12-4ConnectionT" connectionSymbol="IO-Link-SafetyDevice-con-pic.png">
        <ProductRef productId="SafetyDeviceVariant"/>
        <Wire1 function="L+"/>
        <Wire2 function="Other"/>
        <Wire3 function="L-"/>
        <Wire4 function="C/Q"/>
      </Connection>
    </PhysicalLayer>
  </TransportLayers>
  <Test>
    <Config1 index="64" testValue="0x55,0x99"/>
    <Config2 index="1024" testValue="0x00"/>
    <Config3 index="24" testValue="0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x20"/>
    <Config7 index="155">
      <EventTrigger disappearValue="2" appearValue="1"/>
      <EventTrigger disappearValue="4" appearValue="3"/>
    </Config7>
  </Test>
</CommNetworkProfile>
<ExternalTextCollection>
  <PrimaryLanguage xml:lang="en">
    <Text id="T_VendorText" value="Breakthrough in Communication"/>
    <Text id="T_VendorUrl" value="http://www.io-link.com"/>
    <Text id="T_DeviceName" value="Safety Device"/>
    <Text id="T_DeviceFamily" value="Safety Device Family"/>
    <Text id="TN_SafetyDeviceVariant" value="Safety Device"/>
    <Text id="TD_SafetyDeviceVariant" value="Sample for a device with IO-Link Safety"/>
    <!-- Non-Safety parameter -->
    <Text id="TN_NonSafetyParameter" value="Sample Parameter"/>
    <!-- FS Technology parameter -->
    <Text id="TN_FST_DiscrepancyTime" value="Discrepancy Time"/>
    <Text id="TN_FST_Filter" value="Filter"/>
    <!-- IO-Link Safety parameter -->
    <Text id="TN_FSP_Authenticity" value="Authenticity"/>
    <Text id="TD_FSP_Authenticity" value="Authenticity parameters"/>
    <Text id="TN_FSCP_Authenticity_1" value="FSCP_Authenticity_1"/>
    <Text id="TD_FSCP_Authenticity_1" value="&quot;A-Code&quot; from the upper level FSCP system"/>
    <Text id="TN_FSCP_Authenticity_2" value="FSCP_Authenticity_2"/>
    <Text id="TD_FSCP_Authenticity_2" value="Extended &quot;A-Code&quot; from the upper level FSCP system"/>
    <Text id="TN_FSP_Port" value="FSP_Port"/>
    <Text id="TD_FSP_Port" value="PortNumber identifiying the particular FS-Device"/>
    <Text id="TN_FSP_AuthentCRC" value="FSP_AuthentCRC"/>
    <Text id="TD_FSP_AuthentCRC" value="CRC-16 across authenticity parameters"/>
    <Text id="TN_FSP_Protocol" value="Protocol"/>
    <Text id="TD_FSP_Protocol" value="Protocol parameters"/>
    <Text id="TN_FSP_ProtVer" value="FSP_ProtVer"/>
    <Text id="TD_FSP_ProtVer" value="Protocol version (0=current version)"/>
    <Text id="TN_FSP_ProtMode" value="FSP_ProtMode"/>
    <Text id="TD_FSP_ProtMode" value="Protocol mode (1=16 bit CRC, 2=32 bit CRC)"/>
    <Text id="TN_FSP_Watchdog" value="FSP_Watchdog"/>
    <Text id="TD_FSP_Watchdog" value="Monitoring of IO update"/>
    <Text id="TN_FSP_IO_StructCRC" value="FSP_IO_StructCRC"/>
    <Text id="TD_FSP_IO_StructCRC" value="CRC-16 across IO structure description block"/>
    <Text id="TN_FSP_TechParCRC" value="FSP_TechParCRC"/>
    <Text id="TD_FSP_TechParCRC" value="Securing code across FST (technology specific parameter)"/>
    <Text id="TN_FSP_ProtParCRC" value="FSP_ProtParCRC"/>
    <Text id="TD_FSP_ProtParCRC" value="CRC-16 across protocol parameters"/>
```

```xml
      <Text id="TN_FSP_Password" value="FS_Password"/>
      <Text id="TD_FSP_Password" value="Password for the access protection of FSP parameter and Dedicated Tools"/>
      <Text id="TN_FSP_Reset_Password" value="Reset_FS_Password"/>
      <Text id="TD_FSP_Reset_Password" value="Password to reset the FST parameter to factory settings and to reset implicitly the FS-Password"/>
      <Text id="TN_FSP_ParamDescCRC" value="FSP_ParamDescCRC"/>
      <Text id="TD_FSP_ParamDescCRC" value="A dummy variable to store the CRC across the safety parameters within the IODD in the defaultValue"/>
      <!-- Process data -->
      <Text id="TN_ProcessDataIn" value="Process Data In"/>
      <Text id="TN_PDin-Bool1" value="FS process data in Boolean 1"/>
      <Text id="TN_PDin-Bool2" value="FS process data in Boolean 2"/>
      <Text id="TN_PDin-Bool3" value="FS process data in Boolean 3"/>
      <Text id="TN_PDin-Bool4" value="FS process data in Boolean 4"/>
      <Text id="TN_PDin-Bool5" value="FS process data in Boolean 5"/>
      <Text id="TN_PDin-Bool6" value="FS process data in Boolean 6"/>
      <Text id="TN_PDin-Bool7" value="FS process data in Boolean 7"/>
      <Text id="TN_PDin-Bool8" value="FS process data in Boolean 8"/>
      <Text id="TN_PDin-Bool9" value="FS process data in Boolean 9"/>
      <Text id="TN_PDin-Bool10" value="FS process data in Boolean 10"/>
      <Text id="TN_PDin-Bool11" value="FS process data in Boolean 11"/>
      <Text id="TN_PDin-Bool12" value="FS process data in Boolean 12"/>
      <Text id="TN_PDin-Bool13" value="FS process data in Boolean 13"/>
      <Text id="TN_PDin-Int1" value="FS process data in Int 1"/>
      <Text id="TN_PD_FSTrailer" value="F-Message Trailer"/>
      <Text id="TD_PD_FSTrailer" value="Control/Status octet and CRC"/>
      <Text id="TN_PD_Rev" value="Revolutions"/>
      <Text id="TD_PD_Rev" value="Rotational speed"/>
      <Text id="TN_ProcessDataOut" value="Process Data Out"/>
      <!-- Events -->
      <Text id="TN_TransmissionError_CRCSignature" value="Transmission error (CRC signature)"/>
      <Text id="TN_TransmissionError_Counter" value="Transmission error (Counter)"/>
      <Text id="TN_TransmissionError_Timeout" value="Transmission error (Timeout)"/>
      <Text id="TN_UnexpectedAuthenticationCode" value="Unexpected authentication code"/>
      <Text id="TN_UnexpectedAuthenticationPort" value="Unexpected authentication port"/>
      <Text id="TN_IncorrectFSP_AuthentCRC" value="Incorrect FSP_AuthentCRC"/>
      <Text id="TN_IncorrectFSP_ProtParCRC" value="Incorrect FSP_ProtParCRC"/>
      <Text id="TN_IncorrectFSP_TechParCRC" value="Incorrect FSP_TechParCRC"/>
      <Text id="TN_IncorrectFSP_IO_StructCRC" value="Incorrect FSP_IO_StructCRC"/>
      <Text id="TN_WatchdogTimeOutOfSpec" value="Watchdog time out of specification"/>
      <!-- Menu -->
      <Text id="TN_MR_Param_Standard" value="Standard (non-safety) parameter"/>
      <Text id="TN_MR_Param_FST" value="Fail-safe technology parameter"/>
      <Text id="TN_MR_Param_FSP" value="Fail-safe protocol parameter"/>
      <Text id="TN_SR_Param_Standard" value="Standard (non-safety) parameter"/>
      <Text id="TN_SR_Param_FST" value="Fail-safe technology parameter"/>
      <Text id="TN_SR_Param_FSP" value="Fail-safe protocol parameter"/>
      <Text id="TN_StandardProcessData" value="Standard (non-safety) process data in"/>
      <Text id="TN_FS_ProcessData" value="Fail-safe process data in"/>
      <!-- for device test -->
      <Text id="TN_Event1" value="Event 1"/>
      <Text id="TN_Event2" value="Event 2"/>
    </PrimaryLanguage>
  </ExternalTextCollection>
  <Stamp crc="1946410459">
    <Checker name="IODD-Checker V1.1.3" version="V1.1.3.0"/>
  </Stamp>
</IODevice>
```

Annex F
(normative, non-safety related)

Device Tool Interface (DTI) for IO-Link

F.1 Purpose of DTI

For integration of IO-Link Devices in a Master Tool, IODD files shall be used provided by the Device manufacturer. Syntax and semantics of these files are standardized (see [9]) such that the Devices can be integrated independently from the vendor/manufacturer.

However, some applications/standards such as functional safety require a so-called Dedicated Tool for e.g. parameter setting and validation, at least as a complement to the IODD method. This Dedicated Tool shall communicate with its Device and is responsible for the data integrity according to [3]. In the following, the term "Device Tool" is used within this document. Without any additional standardized technology, such an IO-Link system would force the user

- to know which Device Tool is required for a particular Device,
- to enter the communication parameters of the Device both in the Master Tool and in the Device Tool and to keep the parameters consistent,
- to store consistent configuration and parameterization data from both the Master Tool and the Device Tool at one single place to archive project data.

In addition, it would face the Device manufacturer

- with the necessity to implement the communication functionality for each supported field bus system, and
- with the problem of nested communication whenever the target Device is located in a different network and only a proprietary gateway interconnects the networks..

A solution is the Device Tool Interface (DTI) technology specified herein after. It can be used for safety (FS-Master/FS-Device) as well as for non-safety (Master/Device) IO-Link devices.

F.2 Base model

The Device Tool Interface (DTI) comprises three main parts according to Figure 59:

- An invocation interface between Master Tool and Device Tool
- A backward interface between Master Tool and Device Tool ("Backchannel")
- A communication interface between Device Tool and a Communication Server

The combination of these three parts leads to the following user interaction.

A Master Tool is supposed to be already installed on a PC running Microsoft Windows operating system. A Device is configured with the help of the corresponding IODD file of the Device manufacturer. This step includes assignment of port addresses and adjustment of the Device parameters defined in the IODD.

Now, the DTI standard allows for associating Device Tool identification with IO-Link Device identification. The Master Tool uses DTI specific mechanisms to find the Device Tool for a given Device. It provides for example in the context menu of a selected Device an entry that can be used to invoke the Device Tool. As soon as the Device Tool is active, it identifies the selected Device. The user can instantly establish a communication with the Device without entering address information and alike and assign parameter values. Assigned values can be returned to the Master Tool using the Backchannel.

For the communication server part, DTI relies on technology specified in [17]. DTI comprises mechanisms to store and maintain Device data objects (project data).

F.3 Invocation interface

F.3.1 Overview

The invocation interface is used to transfer information from the representation of the Device in the Master Tool to the Device Tool. In order to achieve a high flexibility and to be able to identify different versions of the interface, both the description of the Device Tool capabilities and the invocation parameters are stored in XML based documents. For the assignment from Master Tool to Device Tool the system registry of the Microsoft Windows operating system is used.

Figure F.1 shows the principle of the DTI invocation interface part.

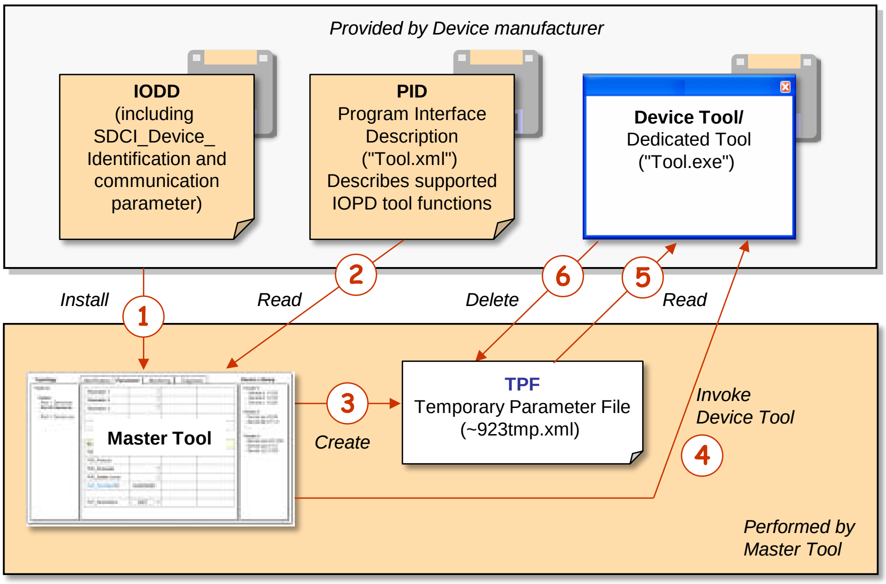


Figure F.1 – Principle of DTI invocation interface

Precondition for the mechanism is the availability of the Master Tool and all used Device Tools on one and the same PC.

For the Tool invocation the following steps are required:

(1) As usual, the IODD file is imported into the Master Tool. The Device is configured and communication settings are made. With the help of (SDCI) Device Identification data the Master Tool is able to find the installed Device Tool and the directory path to the "Program Interface Description" (PID) file. Annex F.3.2 describes this procedure in detail.

(2) The Master Tool reads the content of the PID file. This file contains information about the interface version and the supported Tool functions. The structure of the PID file is described in Annex F.3.3.

(3) Before launching the Device Tool, the Master Tool creates a new "Temporary Parameter File" (TPF) that contains all invocation parameters. See F.3.4 for details.

(4) The Master Tool launches the Device Tool and passes the name of the TPF. See F.3.4.

(5) The Device Tool reads and interprets the content of the TPF file.

(6) The Device Tool deletes the TPF file after processing. See F.3.4.

F.3.2 Detection of Device Tool

F.3.2.1 Registry structure

In order for DTI to identify the type of an IO-Link Device, a specific, unique, and unambiguous "SDCI_Device_Identifier" is used in the PC system registry and within the Temporary Parameter File (TPF).

Figure F.2 shows the structure of the DTI part of the registry. Each class in the diagram represents a registry key. Each attribute in the diagram represents a string value of the registry key. The semantics of the attributes is defined in Table F.1 and Table F.2.

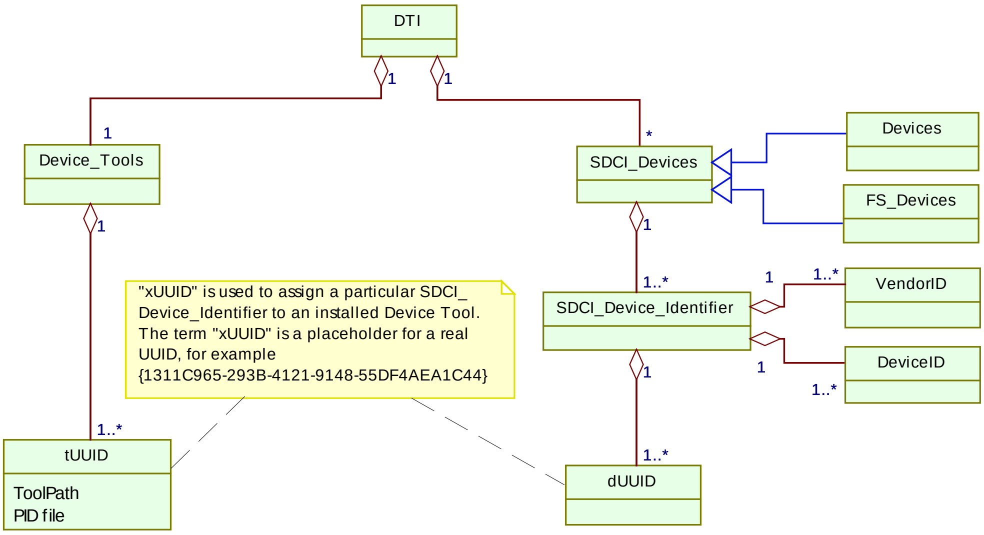


Figure F.2 – Structure of the registry

Since for an SDCI_Device_Identifier an unlimited number of "UUID" elements can be inserted, the Master Tool shall handle all Tools of these "UUID" elements.

F.3.2.2 Device Tool specific registry entries

Each version of a Device Tool is represented by one UUID in the system registry.

The installation program of a Device Tool (32 bit or 64 bit) shall insert this UUID as key under its appropriate registry path:

HKEY_LOCAL_MACHINE\SOFTWARE\IO-Link Community\DTI\Device_Tools or

HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\IO-Link Community\DTI\Device_Tools

A Master Tool shall check both registry paths.

Within this key, two attributes with string values shall be used:

- "PIDfile", containing the absolute path and name of the installed PID file, and
- "ToolPath", containing the absolute path and name of the executable Device Tool file including its file extension (.exe)

Figure F.3 illustrates registry entries for SDCI Devices and Device Tools.

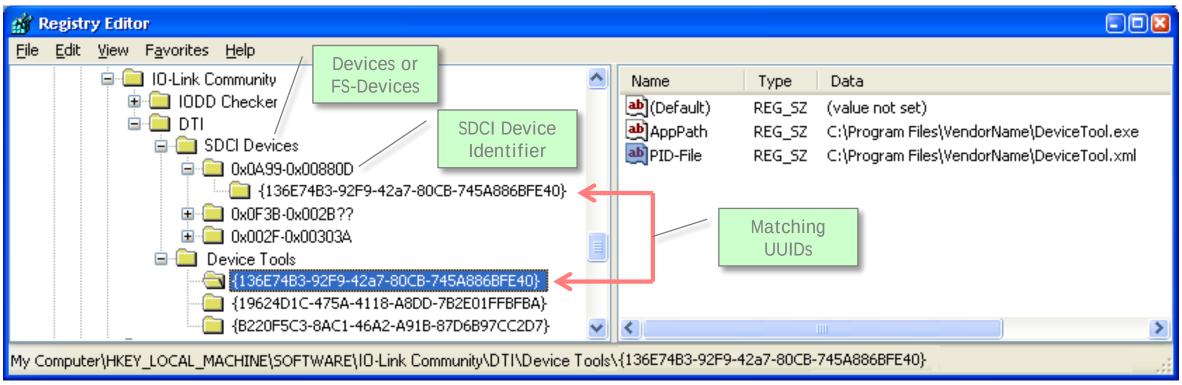


Figure F.3 – Example of a DTI registry

If different versions of a Device Tool for the same Device type exist (same SDCI_Device_Identifier), each version requires a separate UUID in the registry. In the PID files of the Device Tools, different version information shall be provided in the attribute "ToolDescription" of the element "ToolDescription" (see Table F.1). This leads to multiple items in the context menu of the Master Tool, differing in the description text.

NOTE The advantage of a separate entry of the “ToolPath” keyword is a simpler installation procedure for the Device Tool. It can install the PID file without a need to modify this file.

The installation program of a Device Tool shall also insert each UUID as key under the registry path

HKEY_LOCAL_MACHINE\SOFTWARE\IO-Link Community\DTI\SDCI Devices\<SDCI Device Identifier>

IO-Link Devices are identified unambiguously via the following items:

- VendorID (assigned by IO-Link Community)
- DeviceID (assigned by Device/FS-Device manufacturer)

This information is part of the IO-Link Device Description (IODD), which allows the Master Tool to work with the Device (data, parameter) without establishing an online connection to the Device. The IDs can be found at the following locations within an IODD:

(1) //ISO15745Profile/ProfileBody/DeviceIdentity/@vendorId

(2) //ISO15745Profile/ProfileBody/DeviceIdentity/@deviceId

With the help of the registry, the Master Tool is able to read the required information about the Device Tool (in case of safety: Dedicated Tool). Location and structure for the entries shall be commonly agreed upon.

All entries shall be provided by the Device Tool under the following registry path:

HKEY_LOCAL_MACHINE\SOFTWARE\IO-Link Community\DTI\SDCI Devices

Within this path one or more keys can be inserted with the following field structure:

0xvvvv-0xdddddd

The meaning of the fields is:

vvvv: Four-character VendorID in hexadecimal coding

dddddd: Six-character DeviceID in hexadecimal coding.

The question mark character "?" can be used in the DeviceID as wildcard to replace one single character. The number of question marks is only limited by the size of the field. If wildcards are used, the Device Tool is responsible for the check whether it supports the selected object.

The assignment to the Tool is made by a string value within this key. The UUID shall be used as name for the string value. The number of string values is not limited, which in turn means an unlimited number of Tools that can be assigned to the same Device.

Examples for valid keys (see Figure F.3):

0x0A99-0x00880D	The Tool can be launched in the context of a Device with a DeviceID 0x00880D from the vendor with the VendorID 0x0A99.
0x0F3B-0x002B??	The Tool can be started in the context of Devices with a DeviceID in the range of 0x002B00 to 0x002BFF from the vendor with the VendorID 0x0F3B.

F.3.2.3 Processing of the Registry Data

The installation program of the Device Tool is responsible to insert the keys in the system registry as defined in Annex F.3.2.2.

Figure F.4 shows an activity diagram illustrating the detection of a Device Tool in the registry via "SDCI_Device_Identifier".

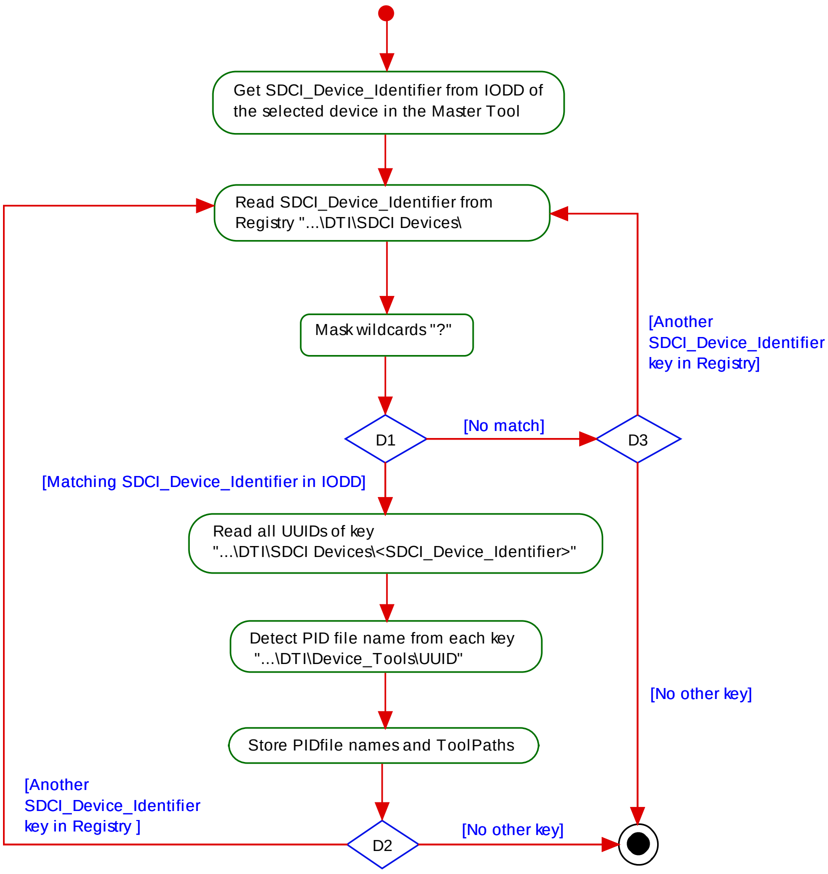


Figure F.4 – Detection of a Device Tool in registry

NOTE All registry keys in Figure F.4 are relative to the path HKEY_LOCAL_MACHINE\SOFTWARE\IO-Link Community

In a first step, the Master Tool gets the SDCI Device Identifier from the IODD of the selected object in the Master Tool. Then all sub keys in the system registry path ...DTI\SDCI Devices shall be compared with this SDCI Device Identifier. If a sub key matches (excepting wildcards), the UUID sub key of this key is used to find the PID file name in the registry path DTI\Device Tools\<UUID>. Since the same PID file name can be found in different locations in the registry, the context menu of the Master Tool shall only show the Device Tools with different PID file names. As a last step, the information in the PID file is used to build the menu items of the Master Tool (Figure F.5).

F.3.3 Program Interface Description – PID

F.3.3.1 General

The Program Interface Description (PID) file describes the properties of the Device Tool and contains data which are required by the Master Tool to build menu items in its graphical user interface (GUI). The PID file is an XML document. The corresponding XML schema is defined in F.9.2. UTF-8 shall be used for character encoding.

This PID file shall be provided by the manufacturer of a Device/Device Tool and installed by the installation program associated with the Device Tool. This installation program shall also insert the name and installation path in the system registry (see F.3.2).

The PID file allows the Master Tool to extend its GUI menu structure by the name of the Device Tool such that the user is able to launch the Device Tool for example from the context menu of a selected Device as illustrated exemplary in Figure F.5.

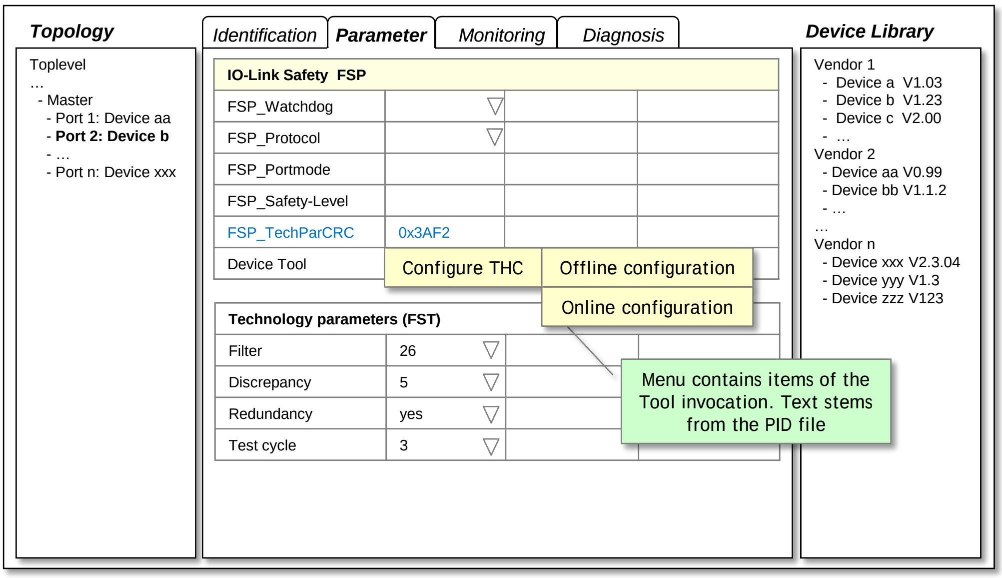


Figure F.5 – Menu for Device Tool invocation

F.3.3.2 Structure of the PID file

The PID file is an XML based document and structured as described in Figure F.6.

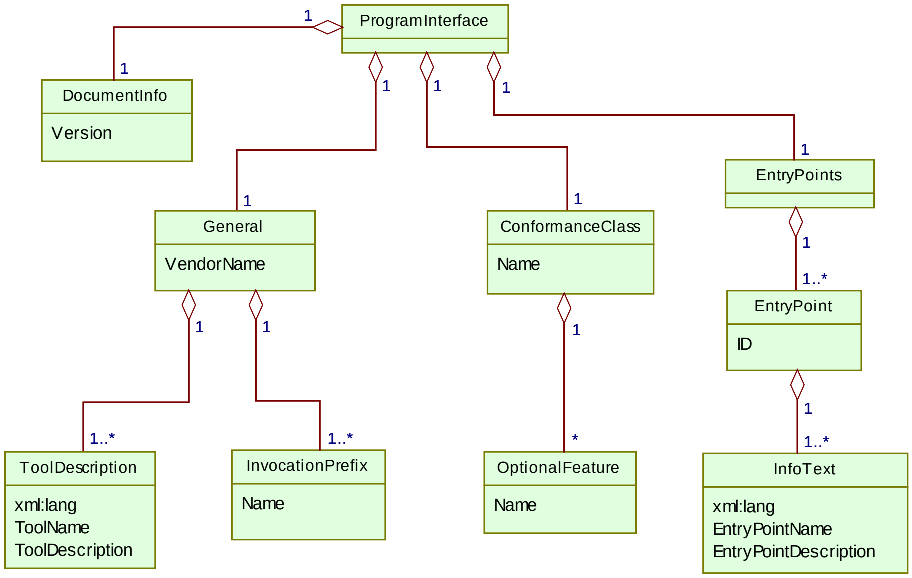


Figure F.6 – Structure of the PID file

The corresponding XML schema can be found in F.9.2. Namespace URI for this file is "http://www.io-link.com/DTI/2016/06/PID".

The elements of Figure F.6 are specified in Table F.1. The column "SV" indicates the schema version a particular attribute has been introduced.

3125 **Table F.1 – Description of PID file elements**

Element	Attribute	Type	M/O	SV	Description
ProgramInterface	–	–	–	1.0	Root element
DocumentInfo	Version	xsd:string	M	1.0	Contains the schema version of PID interface definition. Also determines the newest TPF version supported by this tool. The value shall comply with the following regular expression: \d+(\.\d+)* In this version, the string "1.1" shall be used.
General	VendorName	xsd:string	M	1.0	Contains the name of the Device vendor
ToolDescription	xml:lang	xsd:language	M	1.0	Defines the language of the text. The "2-letter coding" or the "3-letter coding" as defined in ISO 639 shall be used.
	ToolName	xsd:string	M	1.0	Describes the function of the Device Tool. This text shall be used to extend the GUI menu items of the Master Tool. Default element in English language shall always be present.
	ToolDescription	xsd:string	O	1.0	Contains a short description of the Device Tool.
Invocation Prefix	–	–	–	1.0	With this element, the command line arguments of the called Device Tool can be modified. If a Device Tool is able to interpret different command line arguments, usually a prefix is used to define the semantic of an argument. If an InvocationPrefix is present in the PID file, the Master Tool shall insert a blank character as delimiter between the InvocationPrefix string and the file name of the TPF. To interpret the command line argument as a file name for a DTI call, a Device Tool shall be launched as follows: *DeviceTool.exe -i "c:\tmp\TPF01.xml"* In this case, the prefix "-i" shall be entered in the PID file.
	Name	xsd:string	O	1.0	Defines which command line prefix is used when the tool is launched. If this attribute is not present, only the file name of the TPF is used as command line argument. NOTE Since the datatype "string" is used, blank characters (ASCII 32 dec) are allowed. XML Entities are allowed and shall be converted by the Master Tool.
ConformanceClass	Name	xsd:string	M	1.0	Contains the name of the conformance class (F.8.1). One of the following values is allowed: "C1", "C2", or "C3"
OptionalFeature	Name	xsd:string	M	1.0	Name of the implemented feature of the Master Tool as described in Table F.8.
EntryPoints	–	–	–	1.0	This optional element shall be used, if a Device Tool has more than one entry point.
EntryPoint	ID	xsd:string	M	1.0	This element represents an entry point of the Device Tool. Entry points are used to generate additional sub menu items in the "ToolDescription" context menu of the Master Tool. Using entry

Element	Attribute	Type	M/O	SV	Description
					points a Device Tool can provide direct access to Tool specific views or functions. The attribute "ID" identifies an Entry-Point. It shall be unique within a PID file.
InfoText	–	–	–	1.0	The element "InfoText" is used to define language dependent text information for description of the entry point. This information can be used to extend the GUI menu items of the Master Tool. An InfoText element in English language shall always be present here.
	xml:lang	xsd:string	M	1.0	Defines the language of the text. The "2-letter coding" or the "3-letter coding" as defined in ISO 639 shall be used.
	EntryPointName	xsd:string	M	1.0	Describes the function of the entry point. This text shall be used to extend the GUI menu items of the Master Tool.
	EntryPointDescription	xsd:string	O	1.0	Contains a short description of the entry point.

F.3.3.3 Example PID file

The following XML code shows an example content of a PID file with EntryPoints.

```
<?xml version="1.0" encoding="UTF-8"?>
<ProgramInterface xmlns="http://www.io-link.com/DTI/2017/02/PID" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:prim="http://www.io-link.com/DTI/2017/02/Primitives" xsi:schemaLocation="http://www.io-link.com/DTI/2017/02/PID
iosafe_pid_schema_20170225.xsd">
	<DocumentInfo version="V1.0"/>
	<General vendorName="IO-LinkCompany">
		<ToolDescription name="Configure THC" description="IO-Link-16 Safety Device" lang="en"/>
		<ToolDescription name="Konfiguriere THC" description="IO-Link-16 Safety Device" lang="de"/>
		<InvocationPrefix name="/"/>
	</General>
	<EntryPoints>
		<EntryPoint id="1">
			<InfoText name="Offline Configuration" description="Offline Configuration" lang="en"/>
			<InfoText name="Offline Konfiguration" description="Offline Konfiguration" lang="de"/>
		</EntryPoint>
		<EntryPoint id="2">
			<InfoText name="Online Configuration" description="Online Configuration" lang="en"/>
			<InfoText name="Online Konfiguration" description="Online Konfiguration" lang="de"/>
		</EntryPoint>
	</EntryPoints>
	<ConformanceClass name="C3"/>

</ProgramInterface>
```

F.3.4 Temporary Parameter File – TPF

F.3.4.1 General

Due to the large number of parameters to be transferred from the Master Tool to the Device Tool, a parameter transfer by command line arguments is not a good solution. The necessary syntax would become too complex to cover all aspects.

Instead, all required parameters are included into an XML file, called Temporary Parameter File (TPF) by the Master Tool and thus, the name of the XML file is passed as the only command line argument. If the Device Tool requires a command line switch, this information can be extracted from the PID file. See "InvocationPrefix" in Table F.1 for details.

The XML schema for the TPF is defined in F.9.3. For character encoding, UTF-8 shall be used. The Master Tool shall use the newest TPF schema version supported by both the Master Tool and the Device Tool.

After the TPF is interpreted, the Device Tool shall delete the TPF file.

F.3.4.2 Structure of a TPF

The structure of the TPF is defined by the XML schema shown in F.9.3. This schema is built in a generic manner, which means, a new parameter does not require the schema itself to be updated. Thus, new parameters can be introduced without a new definition of the TPF structure.

Namespace URI for this file is "http://www.io-link.com/DTI/2017/02/TPF".

Figure F.7 shows the structure of a TPF.

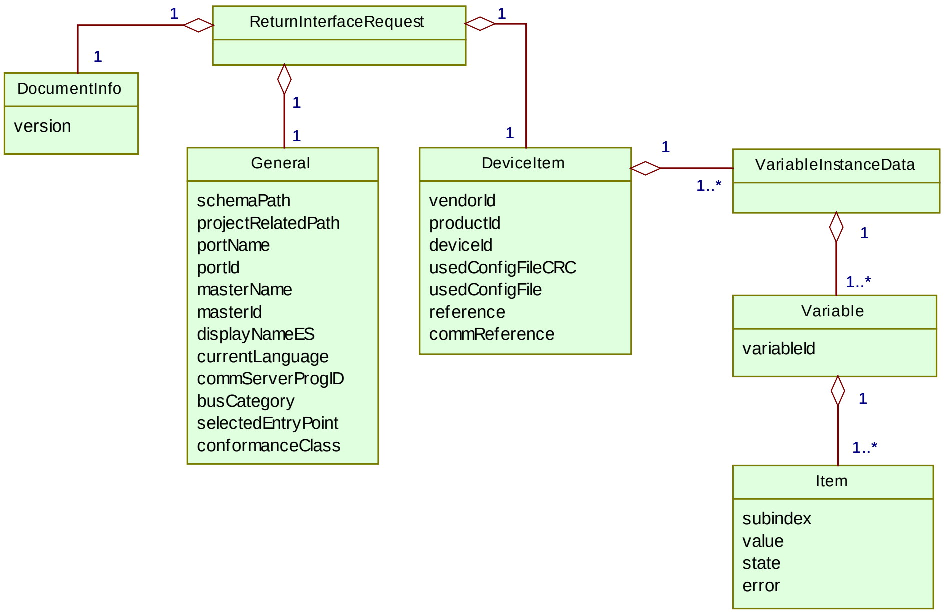


Figure F.7 – Structure of a TPF

The elements of Figure F.7 are specified in Table F.2. The column "SV" indicates the schema version a particular attribute has been introduced.

Table F.2 – Elements of a TPF

Element	Attribute	Type	M/O	SV	Description
InvocationInterface	–	–	M	1.0	Root element
DocumentInfo	Version	xsd:string	M	1.0	Contains the schema version of the TPF interface definition. The value shall comply with the following regular expression: \d+(\.\d+)* One of the following values is allowed: "1.0" Used for TPF based on version 1.0 schema files
General	schemaPath	xsd:string	M	1.0	This attribute defines the path where the schema files for FDT communication schemas and TPF/PID file are stored. • This schema files shall be installed on this path by the Master Tool • The path does not change during runtime of the Master Tool • The path can be used from a Device Tool to initialize the XML

Element	Attribute	Type	M/O	SV	Description
					parser. NOTE Even if no schema validation is used, some XML parsers need the location of the schema files for initialization. In this case, a Device Tool does not need to install an own set of schema files – it should use the schema files in the path defined by this attribute.
	projectRelatedPath	xsd:string	M	1.0	The attribute "ProjectRelatedPath" contains information about a directory which is assigned to the project context of the Master Tool. A Device Tool should use this path for storage of its Device data. The format and structure of this data is defined by the Device Tool itself. Within this directory, additional subdirectories can be created. The Master Tool is responsible to keep all data in the directory tree in its project context. That means, if the project is copied or archived, also this data shall be copied or archived. The attribute "ProjectRelatedPath" contains a unique path (directory) for each combination of Master project and DTI Device Tool. For example, different directories are used for the same tool, if two Master Tool projects are used. The file name in "ProjectRelatedPath" shall consist of the drive letter and an absolute path expression. Alternatively the UNC notation can be used instead of the drive letter.
	portName	xsd:string	M	1.0	Name of used FS-Master port
	portId	xsd:string	M	1.0	ID of used FS-Master port 1 to n
	masterName	xsd:string	M	1.0	User defined name of FS-Master
	displayNameES	xsd:string	M	1.0	Display name of the Master Tool in the language specified in attribute "currentLanguage". The Device Tool can use this name in error messages or user dialogs to provide more understandable texts.
	currentLanguage	xsd:string	M	1.0	Defines which language shall be used by the Device Tool for TPF. The "2-letter coding" or the "3-letter coding" as defined in ISO 639 can be used. If a Device Tool does not support the selected language, the tool shall use its default language.
	commServerProgID	xsd:string	O	1.0	This attribute contains the ProgID of the Communication Server provided by the Master Tool manufacturer. It allows the Device Tool to use the Communication Server functionality. See F.5.6 for details. If this attribute is not provided, the Master Tool does not support a Communication Server.
	busCategory	xsd:string	M	1.1	This attribute is used to specify the used communication protocol. It also can be used to find a corresponding Communication Server.

Element	Attribute	Type	M/O	SV	Description
					Default value is "2C4CD8B8-D509-4ECB-94A7-019F12569C8B"
	selectedEntryPoint	xsd:string	O	1.0	Defines, which entry point of the Device Tool was selected in the Master Tool when the Device Tool was launched. This attribute shall contain only values defined in the attribute "ID" of any element "EntryPoint" of the corresponding PID file. This attribute allows the Device Tool to show an entry point specific GUI when it was launched. If the PID file does not contain any EntryPoint elements, this attribute shall not be used in the TPF.
	conformanceClass	xsd:string	M	1.0	Contains the name of the conformance class of the Master Tool. One of the following values is allowed: "C2" or "C3". See Table F.7.
DeviceItem	vendorId	xsd:string	M	1.0	See Table B.1 in [1]
	productId	xsd:string	M	1.0	See Table B.8 in [1]
	deviceId	xsd:string	M	1.0	See Table B.1 in [1]
	usedConfigFileCRC	xsd:string	M	1.0	IODD stamp
	usedConfigFile	xsd:string	M	1.0	The keyword usedConfigFile contains the file name of the used description file (e.g. IODD). The file name shall consist of the drive letter, an absolute path expression and the file extension. Alternatively the UNC notation can be used instead of the drive letter. The Device Tool It is not allowed to modify the content of the description file.
	reference	xsd:string	M	1.0	Used to identify FS-Device within engineering project
	commReference	xsd:string	M	1.0	This attribute is used with the Communication Server (CS) to address a Device instance unambiguously within the PC. The unique nature of this attribute shall be ensured by the Master Tool. The structure of the attribute is only defined by the Master Tool. It is not allowed to interpret the syntax of this keyword in the Device Tool. LineFeed characters (ASCII 10 dec) are not allowed in the string. This attribute shall be provided for all Device instances of a TPF, if the Device Tool wants to use the CS interface (Conformance Class 3 (C3)) and the commReference is different from the DeviceReference.
VariableInstanceData	–	–	M	1.0	Element "VariableInstanceData" is a container for "Variable" elements (= parameter).
Variable	variableId	xsd:string	M	1.0	Contains the parameter ID
Item	subindex	xsd:string	M	1.0	See [1]
	value	xsd:string	M	1.0	Contains the parameter value. In absence of a parameter-specific

Element	Attribute	Type	M/O	SV	Description
					rule for the representation of the value: Numerical values shall use the decimal coding without left-hand zeros. Negative values shall have a hyphen (ASCII 45 dec) prefix. Separator for floating point values is a dot (ASCII 46 dec). Other separators are not permitted.
	state	xsd:string	M	1.0	Contains parameter status
	error	xsd:string	M	1.0	Contains parameter error

F.3.4.3 Example of a TPF

The following XML code shows the content of an exemplary TPF file.

```
<?xml version="1.0" encoding="UTF-8"?>
<InvocationInterface xmlns="http://www.io-link.com/DTI/2017/02/TPF" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:prim="http://www.io-link.com/DTI/2017/02/Primitives" xsi:schemaLocation="http://www.io-
link.com/DTI/2017/02/TPF IOsafe_TPF_Schema_20170225.xsd">
 <General currentLanguage="en" commServerProgID="DTI.MyCommunicationServer"
projectRelatedPath="\\ServerName\ShareName\Projects" masterId="444444" masterName="CPU-1" portId="0" portName="P1-
4" schemaPath="d:\dti\schema" displayNameEs="MyMTName" busCategory="IOLink" selectedEntryPoint="1"
conformanceClass="C3"/>
 <DeviceItem reference="Project1/Network2/Device3/1897212" commReference="Controller3/Gateway7/Unit4" vendorId="335"
deviceId="6553616" productId="SafetyDeviceVariant" usedConfigFile="d:\IODDfiles\IO-Link-SafetyDevice-20170225-
IODD1.1.xml" usedConfigFileCRC="1946410459">
  <VariableInstanceData>
   <Variable variableId="V_DirectParameters_1">
    <Item subindex="0" state="empty" error="0" value=""/>
    <Item subindex="1" state="empty" error="0" value=""/>
    <Item subindex="2" state="empty" error="0" value=""/>
    <Item subindex="3" state="empty" error="0" value=""/>
    <Item subindex="4" state="empty" error="0" value=""/>
    <Item subindex="5" state="initial" error="0" value="17"/>
    <Item subindex="6" state="empty" error="0" value=""/>
    <Item subindex="7" state="empty" error="0" value=""/>
    <Item subindex="8" state="empty" error="0" value=""/>
    <Item subindex="9" state="empty" error="0" value=""/>
    <Item subindex="10" state="empty" error="0" value=""/>
    <Item subindex="11" state="empty" error="0" value=""/>
    <Item subindex="12" state="empty" error="0" value=""/>
    <Item subindex="13" state="empty" error="0" value=""/>
    <Item subindex="14" state="empty" error="0" value=""/>
    <Item subindex="15" state="empty" error="0" value=""/>
   </Variable>
   <Variable variableId="V_DeviceAccessLocks">
    <Item subindex="1" state="initial" error="0" value="false"/>
    <Item subindex="2" state="initial" error="0" value="false"/>
   </Variable>
   <Variable variableId="V_VendorName">
    <Item subindex="0" state="initial" error="0" value="IO-Link Community"/>
   </Variable>
   <Variable variableId="V_VendorText">
    <Item subindex="0" state="initial" error="0" value="http://www.io-link.com"/>
   </Variable>
   <Variable variableId="V_ProductName">
    <Item subindex="0" state="initial" error="0" value="SafetyDevice"/>
   </Variable>
   <Variable variableId="V_ProductID">
    <Item subindex="0" state="initial" error="0" value="SafetyDeviceVariant"/>
   </Variable>
   <Variable variableId="V_ProductText">
    <Item subindex="0" state="initial" error="0" value="Sample IO-Link Safety"/>
   </Variable>
   <Variable variableId="V_SerialNumber">
    <Item subindex="0" state="empty" error="0" value=""/>
   </Variable>
   <Variable variableId="V_HardwareRevision">
    <Item subindex="0" state="empty" error="0" value=""/>
   </Variable>
   <Variable variableId="V_FirmwareRevision">
    <Item subindex="0" state="empty" error="0" value=""/>
```

```xml
      </Variable>
      <Variable variableId="V_ApplicationSpecificTag">
        <Item subindex="0" state="initial" error="0" value="IO-Link Safety"/>
      </Variable>
      <Variable variableId="V_ErrorCount">
        <Item subindex="0" state="empty" error="0" value=""/>
      </Variable>
      <Variable variableId="V_DeviceStatus">
        <Item subindex="0" state="empty" error="0" value=""/>
      </Variable>
      <Variable variableId="V_DetailedDeviceStatus">
        <Item subindex="1" state="empty" error="0" value=""/>
        <Item subindex="2" state="empty" error="0" value=""/>
        <Item subindex="3" state="empty" error="0" value=""/>
        <Item subindex="4" state="empty" error="0" value=""/>
        <Item subindex="5" state="empty" error="0" value=""/>
        <Item subindex="6" state="empty" error="0" value=""/>
        <Item subindex="7" state="empty" error="0" value=""/>
        <Item subindex="8" state="empty" error="0" value=""/>
      </Variable>
      <Variable variableId="V_ProcessDataInput">
        <Item subindex="1" state="empty" error="0" value=""/>
        <Item subindex="2" state="empty" error="0" value=""/>
        <Item subindex="3" state="empty" error="0" value=""/>
        <Item subindex="4" state="empty" error="0" value=""/>
        <Item subindex="5" state="empty" error="0" value=""/>
        <Item subindex="6" state="empty" error="0" value=""/>
        <Item subindex="7" state="empty" error="0" value=""/>
        <Item subindex="8" state="empty" error="0" value=""/>
        <Item subindex="9" state="empty" error="0" value=""/>
        <Item subindex="10" state="empty" error="0" value=""/>
        <Item subindex="11" state="empty" error="0" value=""/>
        <Item subindex="12" state="empty" error="0" value=""/>
        <Item subindex="13" state="empty" error="0" value=""/>
        <Item subindex="14" state="empty" error="0" value=""/>
        <Item subindex="127" state="empty" error="0" value=""/>
        <Item subindex="128" state="empty" error="0" value=""/>
      </Variable>
      <Variable variableId="V_NonSafetyParameter">
        <Item subindex="0" state="initial" error="0" value="0"/>
      </Variable>
      <Variable variableId="V_FST_DiscrepancyTime">
        <Item subindex="0" state="initial" error="0" value="0"/>
      </Variable>
      <Variable variableId="V_FST_Filter">
        <Item subindex="0" state="initial" error="0" value="0"/>
      </Variable>
      <Variable variableId="V_FSP_Authenticity">
        <Item subindex="1" state="initial" error="0" value="0"/>
        <Item subindex="2" state="initial" error="0" value="0"/>
        <Item subindex="3" state="initial" error="0" value="0"/>
        <Item subindex="4" state="initial" error="0" value="0"/>
      </Variable>
      <Variable variableId="V_FSP_Protocol">
        <Item subindex="1" state="initial" error="0" value="0"/>
        <Item subindex="2" state="initial" error="0" value="1"/>
        <Item subindex="3" state="initial" error="0" value="100"/>
        <Item subindex="4" state="initial" error="0" value="444"/>
        <Item subindex="5" state="initial" error="0" value="0"/>
        <Item subindex="6" state="initial" error="0" value="0"/>
      </Variable>
      <Variable variableId="V_FSP_ParamDescCRC">
        <Item subindex="0" state="initial" error="0" value="444"/>
      </Variable>
    </VariableInstanceData>
  </DeviceItem>
</InvocationInterface>
```


F.3.5 Temporary Backchannel File – TBF

F.3.5.1 General

The TBF should be transfered by a new transaction of the communication server. This transaction is initiated by the Device Tool and can be performed automatically or upon user request. Transaction acknowledgements (TAF) should be implemented indicating reception of the instance values by the Master Tool or indicating a transaction fault (see F.3.6).

F.3.5.2 Structure of the TBF

The structure of the TBF is defined by the XML schema shown in F.9.4. This schema is built in a generic manner, which means, a new parameter does not require the schema itself to be updated. Thus, new parameters can be introduced without a new definition of the TBF structure.

Namespace URI for this file is "http://www.io-link.com/DTI/2017/02/TBF".

Figure F.8 shows the structure of the TBF.

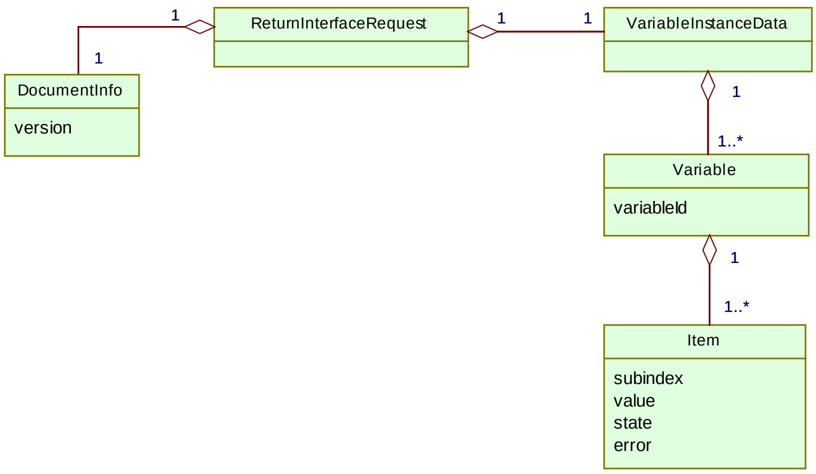


Figure F.8 – Structure of the TBF

The elements of Figure F.8 are specified in Table F.3. The column "SV" indicates the schema version a particular attribute has been introduced.

Table F.3 – Elements of the TBF

Element	Attribute	Type	M/O	SV	Description
ReturnInterfaceRequest	–	–	M	1.0	Root element
DocumentInfo	version	xsd:string	M	1.0	Contains the schema version of the TBF interface definition. The value shall comply with the following regular expression: \d+(\.\d+)* One of the following values is allowed: "1.0" Used for TBF based on version 1.0 schema files
VariableInstanceData	–	–	M	1.0	The element "VariableInstanceData" is a container for "Variable" elements (= parameter).
Variable	variableId	xsd:string	M	1.0	Contains the parameter ID
Item	subindex	xsd:string	M	1.0	See [1]
	value	xsd:string	M	1.0	Contains the parameter value. In absence of a parameter-specific rule for the representation of the value: Numerical values shall use the decimal coding without left-hand zeros. Negative values shall have a hyphen (ASCII 45 dec) prefix. Separator for floating point values is a dot (ASCII 46 dec). Other separators are not permitted.
	state	xsd:string	M	1.0	Contains parameter status

Element	Attribute	Type	M/O	SV	Description
	error	xsd:string	M	1.0	Contains parameter error

F.3.5.3 Example of a TBF

The following XML code shows the content of an exemplary TBF file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ReturnInterfaceRequest xmlns="http://www.io-link.com/DTI/2017/02/TBF" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:prim="http://www.io-link.com/DTI/2017/02/Primitives" xsi:schemaLocation="http://www.io-link.com/DTI/2017/02/TBF IOsafe_TBF_Schema_20170225.xsd">
  <VariableInstanceData>
    <Variable variableId="V_DeviceAccessLocks">
      <Item subindex="1" state="initial" error="0" value="false"/>
      <Item subindex="2" state="initial" error="0" value="false"/>
    </Variable>
    <Variable variableId="V_ApplicationSpecificTag">
      <Item subindex="0" state="initial" error="0" value="IO-Link Safety"/>
    </Variable>
    <Variable variableId="V_NonSafetyParameter">
      <Item subindex="0" state="initial" error="0" value="0"/>
    </Variable>
    <Variable variableId="V_FST_DiscrepancyTime">
      <Item subindex="0" state="initial" error="0" value="0"/>
    </Variable>
    <Variable variableId="V_FST_Filter">
      <Item subindex="0" state="initial" error="0" value="0"/>
    </Variable>
    <Variable variableId="V_FSP_Authenticity">
      <Item subindex="1" state="initial" error="0" value="0"/>
      <Item subindex="2" state="initial" error="0" value="0"/>
      <Item subindex="3" state="initial" error="0" value="0"/>
      <Item subindex="4" state="initial" error="0" value="0"/>
    </Variable>
    <Variable variableId="V_FSP_Protocol">
      <Item subindex="1" state="initial" error="0" value="0"/>
      <Item subindex="2" state="initial" error="0" value="1"/>
      <Item subindex="3" state="initial" error="0" value="100"/>
      <Item subindex="4" state="initial" error="0" value="444"/>
      <Item subindex="5" state="initial" error="0" value="0"/>
      <Item subindex="6" state="initial" error="0" value="0"/>
    </Variable>
    <Variable variableId="V_FSP_ParamDescCRC">
      <Item subindex="0" state="initial" error="0" value="444"/>
    </Variable>
  </VariableInstanceData>
</ReturnInterfaceRequest>
```

F.3.6 Temporary Acknowledgment File – TAF

F.3.6.1 General

Transaction acknowledgements should be implemented indicating reception of the instance values by the Master Tool or indicating a transaction fault. The same mechanism is used as with the TBF (see F.3.5).

F.3.6.2 Structure of the TAF

The structure of the TAF corresponds to the TBF structure in F.3.5.2. However, the root name has changed to "ReturnInterfaceResponse".

F.3.6.3 Example of a TAF

The following XML code shows the content of an exemplary TAF file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ReturnInterfaceResponse xmlns="http://www.io-link.com/DTI/2017/02/TBF" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:prim="http://www.io-link.com/DTI/2017/02/Primitives" xsi:schemaLocation="http://www.io-link.com/DTI/2017/02/TBF IOsafe_TBF_Schema_20170225.xsd">
  <Response value="true"/>
</ReturnInterfaceResponse>
```

F.3.7 Invocation behavior

F.3.7.1 Conventions on Device Tool invocation

Since the directory path of the TPF can contain "blank" characters, the Device Tool shall use the double quote character (") at the beginning and the end of the string when the ".exe" file is invoked.

It is not required for the invoking Master Tool to monitor the status of the launched Device Tools. Even in case an instance of a Device Tool is already running, the Master Tool will generate a new Device Tool invocation whenever the user launches the same tool again.

Therefore, it is the task of the Device Tool to handle multiple invocations. Table F.4 lists invocation cases and possible behaviors.

Table F.4 – Invocation cases and behaviors

Case	Behavior
Device Tool is launched once	No conflicts
Device Tool is already running and works on the same Device instance as in a prior session.	– The Tool should be brought to the foreground of the GUI desktop – Invocation of another instance of the Device Tool shall be avoided
Device Tool is already running and works on another Device instance as provided by the DTI call. The provided DeviceReference is *known* in the Device Tool.	The behavior depends on the design of the Device Tool: – Another Tool instance is launched and opens its Device data – The active GUI is brought to the foreground of the desktop in order to show the Device data of the selected Device
Device Tool is already running and works on another Device instance as provided by the DTI call. The provided DeviceReference is *not known* in the Device Tool.	The behavior depends on the design of the Device Tool: – Another Tool instance is launched and creates a new Device instance – The active GUI is brought to the foreground of the desktop in order to create a new Device instance of the selected Device

If a Device Tool is invoked via DTI, this Tool should not call another Device Tool because the Communication Server cannot interconnect (no nested communication defined for a DTI Communication Server).

F.3.7.2 Handling of the TPF

The name of the TPF will be provided to the Device Tool as a command line parameter. This name shall consist of a drive letter, an absolute path expression and the file extension. Alternatively, the UNC notation can be used instead of the drive letter. The Master Tool is responsible to create the file and unlock it before the Device Tool is invoked in such a manner that the Device Tool has full access to the file. The file name itself is only temporary and a new file name is generated with each Tool invocation.

After interpretation of the content of the TPF file, the Device Tool shall delete this file. Since the Master Tool can also delete this file when it is restarted, it is recommended for the Device Tool to make a "private" copy of the file when the Device Tool is launched.

F.4 Device data objects (DDO)

F.4.1 General

There is no design goal for DTI, to harmonize the different object models of the Device Tools and the Master Tools as well as for engineering systems due to the tremendous variety and complexity. Instead of a common object model, the Device reference is the bridge between a DDO (e.g. parameter instance) in the Master Tool and a DDO in the Device Tool.

F.4.2 Creating DDOs

Since a Device Tool is invoked within the context of a Device in the Master Tool, the DDO shall be initially created in the Master Tool. This is performed via the IODD. For DTI, no extension in the description files is required. With the help of the system registry a Master Tool can find an appropriate Device Tool to handle the newly created DDO.

Figure F.9 illustrates the activities during Device Tool invocation.

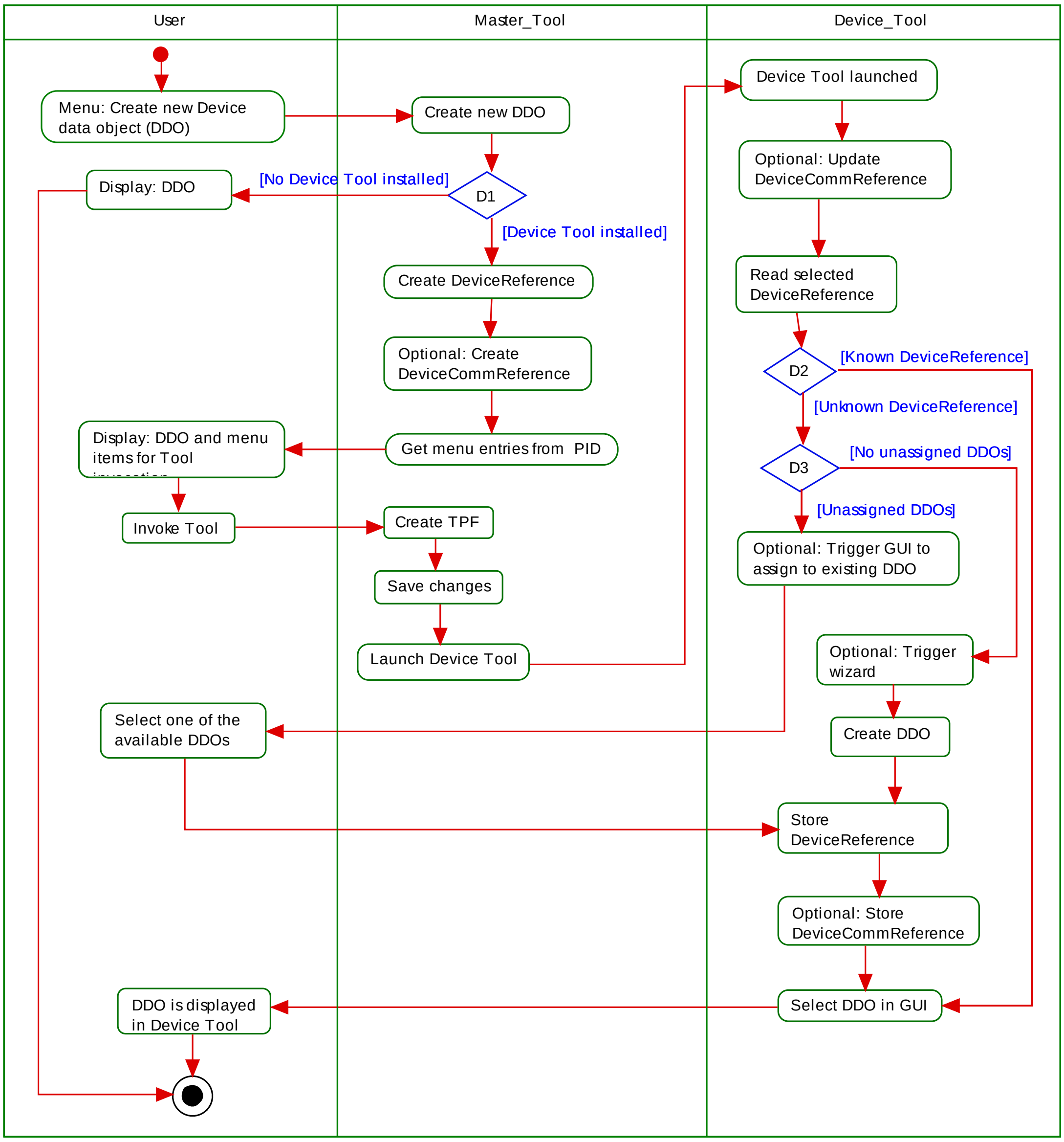


Figure F.9 – Activity diagram for the DDO handling

The Master Tool shall generate a Device reference for each new instance of a Device, whose SDCI Device Identification is registered in the registry as described in Annex F.3.2. This reference shall be unique at least within the Master Tool project. It shall be used in the keyword "DeviceReference" of the TPF and shall not be changed for the lifetime of the Device.

If the Master Tool supports Conformance Class 3 (see Annex F.8), it can additionally generate a Device communication reference for each new Device instance. This reference shall be unique within the PC. It shall be used in the keyword "DeviceCommReference" of the TPF and shall not be changed for the lifetime of the Device except when copying an entire Master Tool project or retrieving a Master Tool project. When the copying is done outside of the Master Tool (for example via the Windows Explorer), the Master Tool shall detect the copy when opening the project the next time and then issue new, unique Device communication references.

It is the decision of the Master Tool whether the DDO reference is generated whenever a new instance is created or upon the first call of the Device Tool after the creation of the DDO. When a new instance of a DDO is created in the Master Tool, there is no corresponding DDO in the Device Tool at the first Tool invocation. In this case, the Device Tool shall create an own instance of the DDO in its own DDO administration. If the user must enter some more data, the Device Tool can start a wizard in order to guide the user. After this step, the reference shall be stored in the Device Tool project so that the Tool can select the right DDO when it is launched again with the same reference.

If a DDO is created initially in the Device Tool, the corresponding DDO in the Master Tool cannot be created automatically. In this case, the user shall create a new DDO in the Master Tool manually. If the Device Tool is now launched in the context of the Master Tool, the Device Tool can show a list of unassigned DDOs of the same type and let the user decide which DDO of the Device Tool corresponds to the newly created DDO in the Master Tool.

F.4.3 Copying DDOs

When a DDO is copied in the Master Tool, only the IODD parameter settings are copied. For the new DDO instance, a new DDO reference (DeviceReference, DeviceCommReference) shall be generated by the Master Tool. The DDO is not copied in the Device Tool. At the next nvocation, a Device Tool can react on this new DDO reference. From the point of view of the Device Tool, there is no difference between a copied DDO and a newly created DDO.

If a complete project is copied in the Master Tool, the DDO references shall not change. Only the DeviceCommReferences will be changed by the Master Tool to enable different routing info. The Master Tool shall copy all files in the "ProjectRelatedPath" directory to the new destination. If a Device Tool is launched from a copied project, it will find all Device Tool specific data as within the original project.

F.4.4 Moving DDOs

If a DDO is moved in the Master Tool to another location within the same project, the Device reference shall not change.

In order to react in the Device Tool upon moved Devices besides the selected Device, the option "UsesMultipleDeviceInformation" shall be used.

F.4.5 Deleting DDOs

If a DDO is deleted in the Master Tool, the corresponding DDOs in the Device Tool should normally also be deleted. This cannot be done automatically due to a missing unique storage model (save, undo...) for all Tools (see Annex F.4.1).

The Master Tool provides a list of used Device references in the TPF. This list can be interpreted by the Device Tool to find out, which DDOs of the same PLC in the Device Tool project are no more part of the TPF. If one or more DDOs are missing in the TPF, the Device Tool can now ask the user which DDOs to delete automatically or to keep internally as unassigned DDOs for a later reuse. Since this behavior of the Device Tool is optional, it shall be described in its PID file with feature name “SupportsObjectDeletion”.

If a Device Tool does not implement this functionality, the Master Tool shall display a message informing the user that these changes shall be made manually in the Device Tool.

F.5 Communication Interface

F.5.1 General

As already explained in Annex F.1, there is no seamless communication solution for stand-alone Device Tools such as "Dedicated Tools" for functional safety in IO-Link so far. The only possibility in the past has been a separate point-to-point communication connection, for example RS232, USB, or alike, between a Device and a PC running the Device Tool software. Each of these connections requires appropriate driver software with different programming API for the Device and for the different PC communication interfaces.

This leads to the problem that a Device Tool either can work only with one particular communication interface or that the Device Tool has to implement different APIs for Device driver integration.

Another problem in a plant is that the network structure often requires communication across network boundaries (Routing). Due to the many fieldbuses and different communication protocols, it is very cumbersome to achieve an integrated network with routing functions for Device Tools down to the associated Device (see Figure F.10).

The second major part of DTI solves two problems:

- All Devices/FS-Devices and their Device Tools/Dedicated Tools can rely on one particular communication interface.
- The chosen communication technology is standardized in IEC 62453 and solves the routing problem across network boundaries.

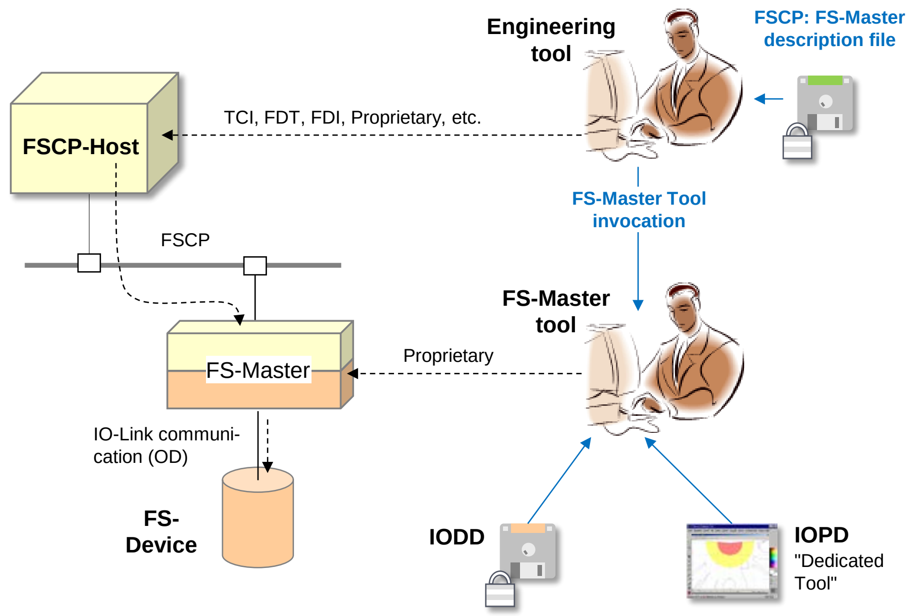


Figure F.10 – Communication routes between Device Tool and Device

F.5.2 Principle of DTI communications

The communication interface consists of a component which provides a unique interface (API) to the Device Tool. This component is able to provide communication functionality for different field busses and also proprietary network protocols. The communication parameters which are necessary to establish a connection are entered in the Master Tool and passed to the Device Tool when it is launched.

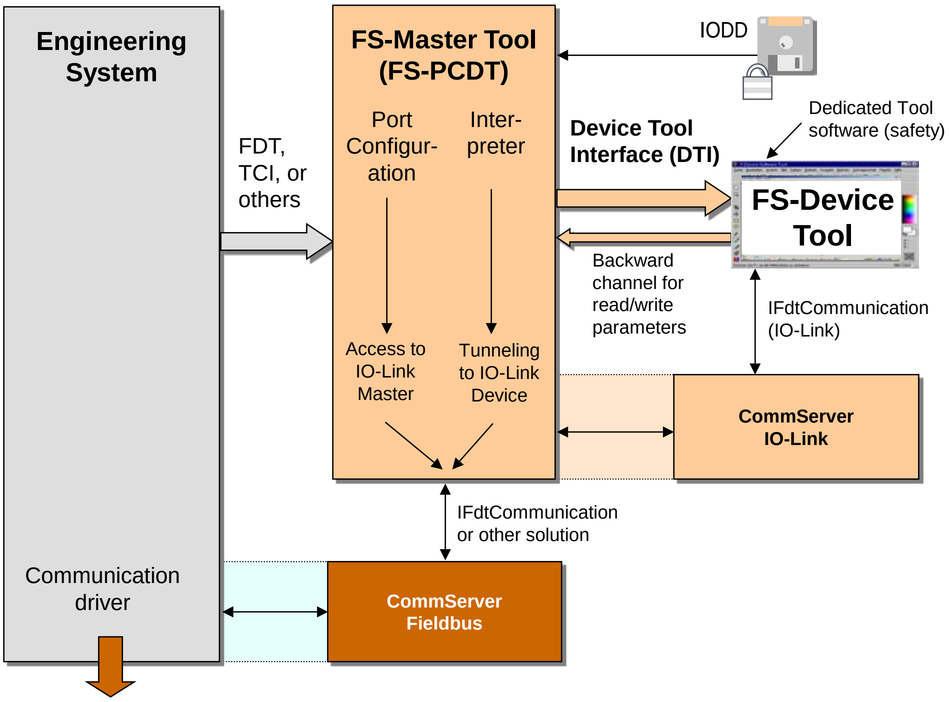


Figure F.11 – Routing across networks and IO-Link

Figure F.10 shows fieldbus or proprietary networks between the PC and the Device. Figure F.11 shows the mapping to software and Communication Servers. In this case, the Communication Server (Fieldbus) requires information about the network protocol. This routing information is generated by the Engineering System and transferred to the Communication Server (Fieldbus). Due to the fact that manufacturer specific data has to be exchanged, the Communication Server and the Engineering System must be provided by the same manufacturer.

The routing information for the second Communication Server (IO-Link) is generated by the Master Tool and transferred to this CS. When the Device Tool is started, only a communication reference to the Device is passed. This reference is forwarded from the Device Tool to the Communication Server. With the help of the routing information from the Engineering System, the Communication Server is able to create physical network addresses and to establish a connection to the Device. Figure F.12 shows the relationships between the components involved.

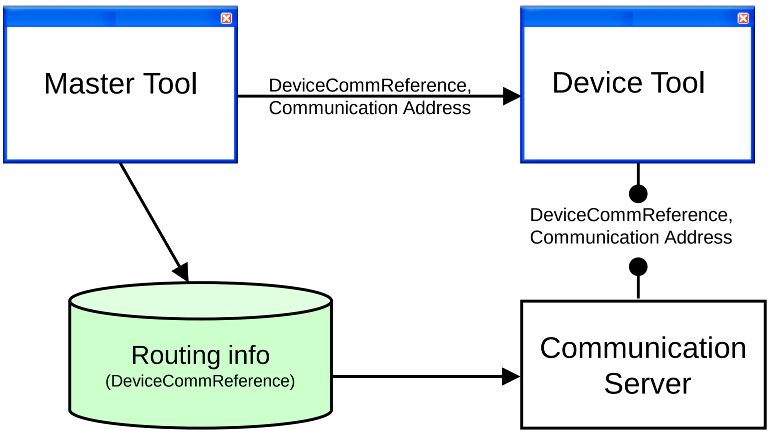


Figure F.12 – Communication Server

It is always possible for a Device Tool to use its native communication interfaces (for example serial RS232) as an alternative besides the Communication Server.

F.5.3 Gateways

A Communication Server allows a communication connection across network boundaries (see Figure F.11).

The Engineering System, all Device Tools and the Communication Server are located on the same PC which is connected e.g. via an Ethernet adapter to a network. The target Devices can be found behind a gateway which can work in different ways. From the Device Tool point of view, it is irrelevant where the Device is located because the network structure is handled by the Communication Server.

The Communication Server is potentially able to manage all gateway types which are supported by the Engineering System itself. The gateway functionalities itself are encapsulated by the Communication Server. Only gateway types known by the Communication Server can be supported (no nested communication).

If a device can be reached through multiple paths in the network, it is up to the Engineering System to decide, which network path is used for communication.

F.5.4 Configuration of the Communication Server

In order to build the network communication addresses from the Device communication reference, the Communication Server requires configuration data from the Engineering System/Master Tool. The structure of configuration data itself and the way how the data is sent to the Communication Server is manufacturer specific and will not be standardized.

F.5.5 Definition of the Communication Interface

The Communication Server implements the interface "IFdtCommunication" and uses the "IFdtCommunication-Events" and "IFdtCommunicationEvents2" as described in IEC 62453. All other DTM interfaces which are described in IEC 62453 are not relevant for the Communication Server. Due to this constraint, a Communication Server cannot be used in an FDT environment as communication DTM.

F.5.6 Sequence for establishing a communication relation

An interaction of Engineering System/Master Tool, Device Tool and Communication Server (CS) is required to establish a communication relation.

The sequence is as follows:

At first, a Device is integrated into the Master Tool with the corresponding configuration file (IODD). Within the Engineering System, communication addresses and bus parameter are adjusted. Together with other network data, topology data for the network is the result.

Furthermore, the Master Tool shall build a unique Device communication reference. This reference is passed to the Device Tool when it is launched with the help of the TPF (keyword “DeviceCommReference”). The Device Tool is now able to establish a connection to the Device using the Communication Server and Device communication reference.

The Communication Server itself interprets the Device communication reference and converts it to network addresses. Therefore it uses the configuration data from the Master Tool. Because it is up to the CS to decide if the Device communication reference or the communication address itself is used, the Device Tool shall always pass both attributes in the ConnectRequest XML document.

If no routing functionality is required, the CS does not require the proprietary configuration. In order to connect, the CS can use the communication address itself from the Master Tool.

Figure F.13 shows how a communication connection is established.

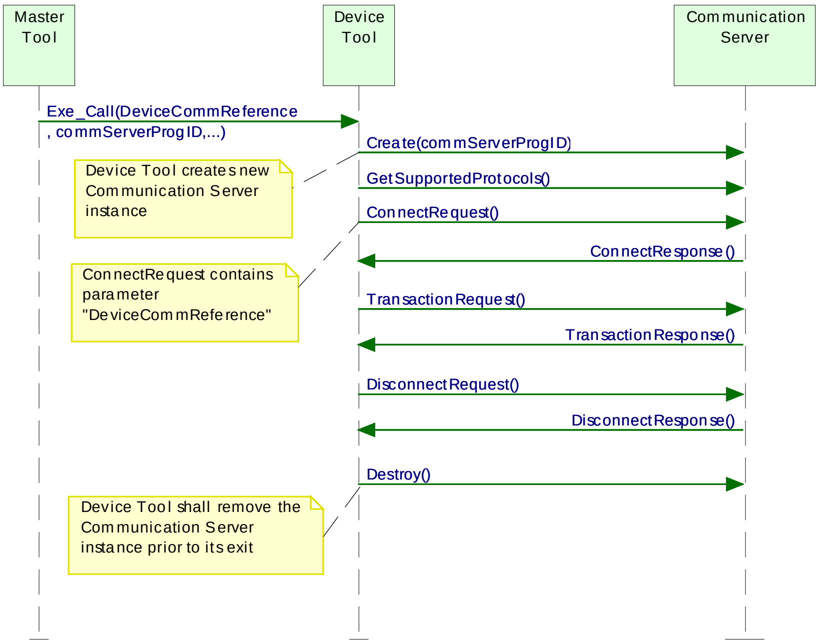


Figure F.13 – Sequence chart for establishing communication

The passed ProgID (Keyword commServer-ProgID) can be used to create a new instance of the Communication Server by the Device Tool. There is a 1:1 relationship between Device Tool and Communication Server instance. The Communication Server instance is able to connect to one or more Devices.

Figure F.14 shows a code fragment in C++ as an example on how to create a new instance.

commServer-ProgID of TPF

```
r1 = CLSIDFromProgID(L"TCI.MyCommunicationServer", &clsid);
r1 = CoCreateInstance (clsid, NULL, CLSCTX_INPROC_SERVER,
     __uuidof (IFdtCommunication), (void**)&m_pITciCommunicationServer);
```

Figure F.14 – Create Communication Server instance

It is recommended to create the Communication Server instance as "in process server" (CLSCTX_INPROC_SERVER) due to performance issues.

After a new instance of the Communication Server is created, all methods of the interface "IFdtCommunication" can be called. At first a Device Tool shall call the "GetSupportedProtocols" method to find out if the required protocol is supported by the CS. If not, the Device Tool shall inform the user. A new connection is established with help of the function "ConnectRequest". Among others, as invocation parameter a pointer to the callback interface (Interface IFdtCommunicationEvents) is passed. This means that a Device Tool shall implement this interface.

The Device Tool is responsible to release the Communication Server instance when the Tool exits. If the Communication Server instance was created in the process of the Device Tool, as recommended before, this is done automatically since the instance is terminated with the process of the Device Tool.

F.5.7 Usage of the Communication Server in stand-alone mode

If a Device Tool is not called from a Master Tool with DTI, it shall find out the ProgID of the Communication Server by itself. In this case the "Component Categories" of the system registry can be used (HKEY_CLASSES_ROOT\Component Categories).

The following values are defined for the DTI Communication Server:

Symbolic Name of CatID: CATID_DTI_CS

UUID of CatID: {7DDC60A6-1FD4-45a2-917F-0F8FC371BC57}

A Device Tool is able to find out the ProgID of the Communication Server with the help of the Standard Component Categories Manager. If more than one component is assigned to this category, the user of the Device Tool shall select one of the Communication Servers.

If a Communication Server does not support the "Stand-Alone" mode (i.e. a Communication Server instance cannot be created by a Device Tool), a system registry entry should not be made.

A Device Tool that supports Conformance Class 3 and is intended for "Stand-Alone" mode shall store the DeviceCommReferences together with its DDOs. Whenever the DeviceCommReference is changed by the Master Tool while copying the entire project or while retrieving the project, the Device Tool shall check and – if changed – update the DeviceCommReference when called from the Master Tool with DTI. There are two general possibilities:

1) The Device Tool checks and updates the DeviceCommReference of a particular Device immediately before connection.

NOTE After copy/retrieval of a Master Tool project, the user should call the Device Tool via DTI and connect to the particular Device(s) prior to the connection to this/these Device(s) later on in "Stand-Alone" mode.

2) The Device Tool checks and updates the DeviceCommReferences of all Devices immediately after being called by the Master Tool via DTI.

NOTE After copy/retrieval of a Master Tool project, the user should call the Device Tool via DTI. Then, all Devices can be connected later in “Stand-Alone” mode.

F.5.8 IO-Link specifics

The IO-Link schema defined in [16] shall be used as communication schema.

Table F.5 shows the mapping between the TPF keywords and the attributes in the communication schema.

Table F.5 – Communication Schema mapping

Attribute of ConnectRequest element (FDTIOLinkCommunicationSchema.xml)	Parameter Keyword in TPF file	Remarks
fdt:nodeId	–	Unused
systemTag	"DeviceCommReference" attribute of element "Device".	

The communication parameters passed during the Device Tool invocation shall be used as input for the Connect Request XML document to be used in the connect method. Additionally, the device communication reference (Keyword "DeviceCommReference" in Table F.5) shall be entered in the Connect Request XML document as attribute “systemTag”. Figure F.15 shows an example.

```
<?xml version="1.0"?>
<FDT xmlns="x-schema:FDTIOLinkCommunicationSchema.xml"
        xmlns:fdt="x-schema:FDTDataTypesSchema.xml">
        <ConnectRequest systemTag="Controller3/Gateway2/Unit1"/>
</FDT>
```

DeviceCommReference of TPF

Figure F.15 – Example of a Connect Request XML document for IO-Link

F.5.9 Changing communication settings

If it is necessary to change the communication address (Master, port?) in the Master Tool, the Device Tool needs information about the new communication address. This shall be done via relaunching the Device Tool by the user of the Master Tool. During relaunch, the new communication parameters are passed to the Device Tool. With these communication parameters a new communication relation can be established to the Device.

If the Device communication reference is used instead of the communication address between Device Tool and Communication Server, no relaunch of the Tool is required, because the Device communication reference does not change whenever the communications address changes. In this case, the Communication Server itself can reconnect to the Device with the new communication address (Master, port).

For an existing connection, changed communication parameters in the Master Tool project shall not have any impact. Changed communication parameters shall be used when a connection is (re)established.

F.6 Reaction on incorrect Tool behavior

Table F.6 describes the system reaction if a Master Tool or Device Tool works incorrectly.

Table F.6 – Reaction on incorrect Tool behavior

Fault	Description	System reaction
XML structure of PID file not valid	The PID file of a Device Tool does not validate with the XML Schema in Annex F.9.1	The Master Tool should only show an error message if required schema elements or attributes are missing. All unknown elements or attributes shall be ignored.
XML structure of TPF file not valid	The TPF file generated by the Master Tool does not validate with the XML Schema in Annex F.9.3	The Device Tool should only show an error message if required schema elements or attributes are missing. All unknown elements or attributes shall be ignored.
Device Tool cannot be invoked	When the operation system is instructed to create a new process (Tool invocation) the function returns an error code. Reason could be that the path of the exe file in the system registry is incorrect.	Master Tool shall show an error message (Tool cannot be invoked) with the name and path of the exe file.
CommunicationServer object cannot be created. See F.5.6	The "CoCreateInstance" function returns an error code when an object with the ProgID of the TPF should be instantiated.	The Device Tool should show an error message.
TPF file not deleted by the Device Tool	The TPF file was not removed by the Device Tool as described in Annex F.3.1	Master Tool should delete the TPF file when it is launched (garbage collection). If the file cannot be deleted, the Master Tool should not show an error message.
DeviceCommReference not valid (Communication channel cannot be established). See Annex F.5.	Device Tool is using a not existing DeviceCommReference in the Master Tool.	The Device Tool should show an error message.

F.7 Compatibility

F.7.1 Schema validation

XML documents can easily be validated with the help of standard parsers and schema files. If the structure of an XML document does not follow the rules defined in the corresponding schema, the XML parser rejects the document. This is not very practical if Tools with different versions of DTI files shall work together since a newer XML document cannot be processed by previous software.

In order to implement a robust model, the Master Tool and the Device Tools shall ignore any XML attributes or elements not recognizable in a valid XML document. This means that XML schema validation shall not be used. The schema files in Annex F.9 are for information purposes only.

The installation program of the Device Tool can always install the newest PID file version. The Master Tool shall ignore any unknown XML attributes or elements.

F.7.2 Version policy

If it is necessary to modify the structure definition of a TPF with the result that a new version of the invocation interface is defined, the Master Tool shall ensure that the right version of the TPF is created. That means it shall use an earlier version of the structure if the Device Tool is only able to support the earlier version.

The PID file version of the Device Tool determines the newest supported version of the corresponding Device Tool. See Annex F.3.3 for details.

If a Device Tool supports a newer version than the Master Tool, the Master Tool uses its newest TPF version. In this case the Device Tool shall work with the old schema version.

F.8 Scalability

F.8.1 Scalability of a Device Tool

The manufacturer of a Device Tool can choose to support different function levels of DTI as shown in Table F.7.

Table F.7 – DTI conformance classes

Conformance Class	Description
C1 (Navigation)	Setup program creates system registry entries as described in Annex F.3.2. This allows the user to invoke the Device Tool from the context of a selected Device in the Master Tool without any impact on an existing Device Tool itself.
C2 (Parameter transfer)	The Device Tool uses the information of the TPF. In this case, for example, the Tool is able to read FST parameter instances or to use a communication address for its proprietary communication channel. This way, the user can be relieved from multiple entries. The implementation effort is limited to evaluation of the TPF file for internal initialization of the Device Tool.
C3 (DTI communication with optional backchannel)	The full functionality is available if the Device Tool uses the DTI Communication Server. This component enables the Tool to manage all network boundaries implemented by the Master Tool. In this case the Device Tool shall support the IFdtCommunication/IFdtCommunicationEvents/IFdtCommunicationEvents2 interface. In case of the backchannel option, the Master Tool uses the information of the TBF. In this case, for example, the Tool is able to read FST parameter instances or to use the I/O Process Data description. This way, the user can be relieved from multiple entries. The implementation effort is limited to evaluation of the TBF file for internal processing of the Master Tool.

Table F.8 shows the DTI relevant features of a Device Tool.

Table F.8 – DTI feature levels of Device Tools

Function	Annex	Conformance Class	Feature Name for PID file
Make system registry entries	F.3.2	C1	–
Provide PID file during installation procedure	F.3.3	C1	–
Avoid multiple program instances		C2	–
Interpret TPF	F.3.4	C2	–
Delete TPF	F.3.7.2	C2	–
Supports deletion of DDOs not	F.4.5	C2 – optional feature	SupportsObjectDeletion

Function	Annex	Conformance Class	Feature Name for PID file
in TPF			
Use the Communication Server interface		C3	–

F.8.2 Scalability of a Master Tool

A Master Tool shall support all DTI feature levels/conformance classes.

F.8.3 Interactions at conformance class combinations

Table F.9 defines how a Master Tool and a Device Tool shall interact depending on their conformance class.

Table F.9 – Interactions at conformance class combinations

Master Tool	Device Tool	Interaction
C2 or C3	C1	Device Tool is launched, no parameters are passed. The Master shall not generate a TPF because it would not be deleted by the Device Tool.
C2 or C3	C2	Device Tool is launched, Parameters are passed through TPF.
C2	C3	Device Tool is launched, Parameters are passed through TPF.
C3	C3	Device Tool is launched, Parameters are passed through TPF. Communication via Communication Server is possible.

F.9 Schema definitions

F.9.1 General

The schema definitions in this Annex F.9 are for information only (see Annex F.7.1).

F.9.2 Schema of the PID

Figure F.16 shows the XML schema of the Program Interface Description file.

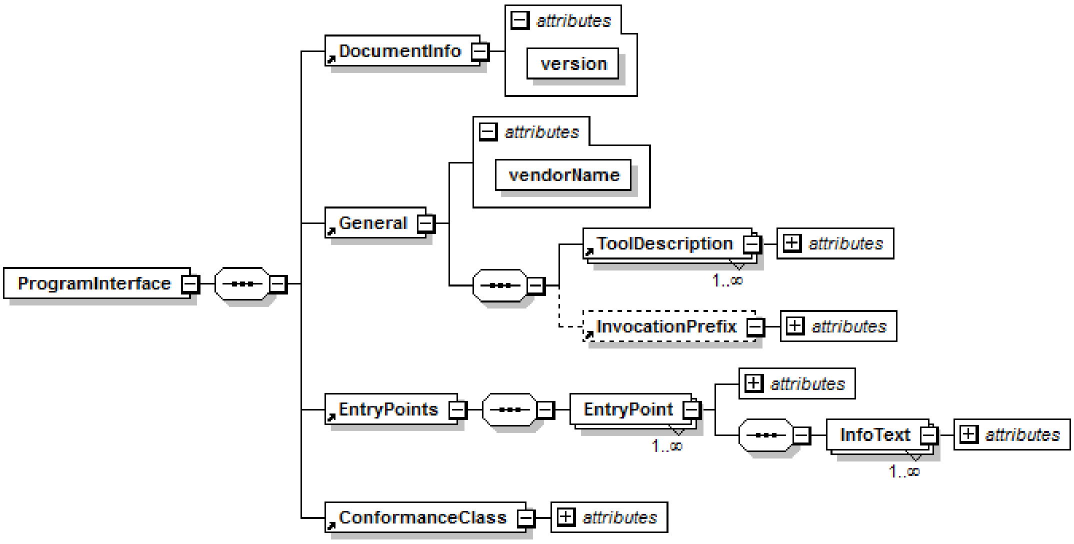


Figure F.16 – XML schema of the PID file

Figure F.16 is based on the XML code as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```xml
<xsd:schema xmlns="http://www.io-link.com/DTI/2017/02/PID" xmlns:prim="http://www.io-link.com/DTI/2017/02/Primitives"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" targetNamespace="http://www.io-link.com/DTI/2017/02/PID"
elementFormDefault="qualified" attributeFormDefault="unqualified" version="1.0">
 <xsd:import namespace="http://www.w3.org/XML/1998/namespace"/>
 <xsd:import namespace="http://www.io-link.com/DTI/2017/02/Primitives" schemaLocation="DTI-Primitives1.0.xsd"/>
 <xsd:element name="DocumentInfo">
  <xsd:complexType>
   <xsd:attribute name="version" use="required">
    <xsd:simpleType>
     <xsd:restriction base="xsd:string">
      <xsd:pattern value="V\d+(\.\d+){1,7}"/>
     </xsd:restriction>
    </xsd:simpleType>
   </xsd:attribute>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="ToolDescription">
  <xsd:complexType>
   <xsd:attribute name="lang" type="xsd:string" use="required"/>
   <xsd:attribute name="name" type="xsd:string" use="required"/>
   <xsd:attribute name="description" type="xsd:string" use="required"/>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="InvocationPrefix">
  <xsd:complexType>
   <xsd:attribute name="name" use="required">
    <xsd:simpleType>
     <xsd:restriction base="xsd:string"/>
    </xsd:simpleType>
   </xsd:attribute>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="General">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element ref="ToolDescription" maxOccurs="unbounded"/>
    <xsd:element ref="InvocationPrefix" minOccurs="0"/>
   </xsd:sequence>
   <xsd:attribute name="vendorName" type="xsd:string" use="required"/>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="EntryPoints">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="EntryPoint" maxOccurs="unbounded">
     <xsd:complexType>
      <xsd:complexContent>
       <xsd:extension base="prim:ObjectT">
        <xsd:sequence>
         <xsd:element name="InfoText" maxOccurs="unbounded">
          <xsd:complexType>
           <xsd:attribute name="lang" type="xsd:string" use="required"/>
           <xsd:attribute name="name" type="xsd:string" use="required"/>
           <xsd:attribute name="description" type="xsd:string" use="required"/>
          </xsd:complexType>
         </xsd:element>
        </xsd:sequence>
        <xsd:attribute name="id" type="prim:IdT" use="required"/>
       </xsd:extension>
      </xsd:complexContent>
     </xsd:complexType>
    </xsd:element>
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
 <xsd:element name="ConformanceClass">
  <xsd:complexType>
   <xsd:attribute name="name" use="required">
    <xsd:simpleType>
     <xsd:restriction base="xsd:string">
      <xsd:enumeration value="C1"/>
      <xsd:enumeration value="C2"/>
      <xsd:enumeration value="C3"/>
     </xsd:restriction>
    </xsd:simpleType>
   </xsd:attribute>
  </xsd:complexType>
```

```
    </xsd:element>
    <xsd:element name="ProgramInterface">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element ref="DocumentInfo"/>
          <xsd:element ref="General"/>
          <xsd:element ref="EntryPoints"/>
          <xsd:element ref="ConformanceClass"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
</xsd:schema>
```

F.9.3 Schema of the TPF

Figure F.17 shows the XML schema of the Temporary Parameter File.

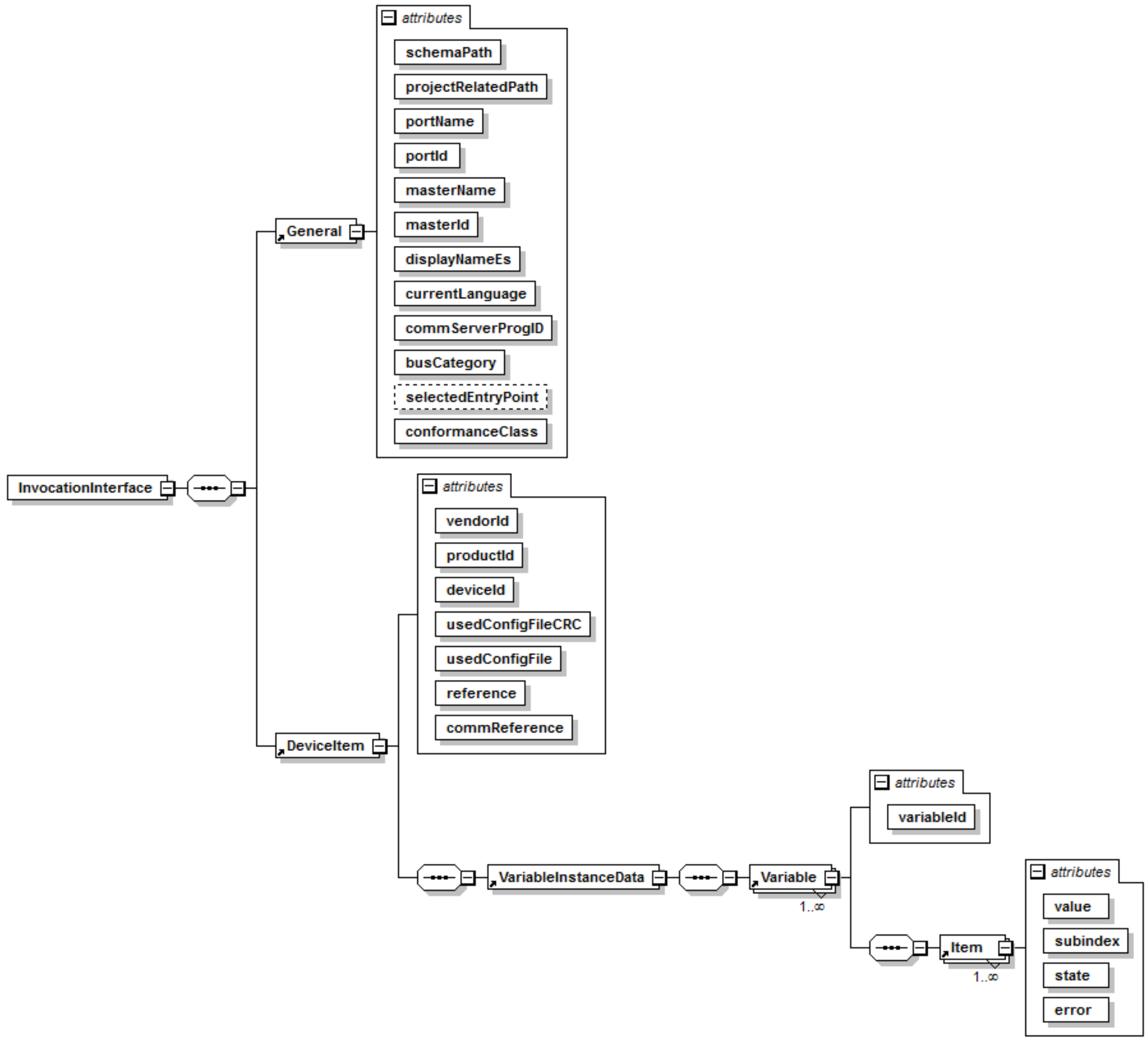


Figure F.17 – XML schema of the TPF

Figure F.17 is based on the XML code as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<InvocationInterface xmlns="http://www.io-link.com/DTI/2017/02/TPF" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:prim="http://www.io-link.com/DTI/2017/02/Primitives" xsi:schemaLocation="http://www.io-
link.com/DTI/2017/02/TPF IOsafe_TPF_Schema_20170225.xsd">
  <General currentLanguage="en" commServerProgID="DTI.MyCommunicationServer"
projectRelatedPath="\\ServerName\ShareName\Projects" masterId="444444" masterName="CPU-1" portId="0" portName="P1-
4" schemaPath="d:\dti\schema" displayNameEs="MyMTName" busCategory="IOLink" selectedEntryPoint="1"
conformanceClass="C3"/>
```

```xml
 <DeviceItem reference="Project1/Network2/Device3/1897212" commReference="Controller3/Gateway7/Unit4" vendorId="335" deviceId="6553616" productId="SafetyDeviceVariant" usedConfigFile="d:\IODDfiles\IO-Link-SafetyDevice-20170225-IODD1.1.xml" usedConfigFileCRC="1946410459">
  <VariableInstanceData>
   <Variable variableId="V_DirectParameters_1">
    <Item subindex="0" state="empty" error="0" value=""/>
    <Item subindex="1" state="empty" error="0" value=""/>
    <Item subindex="2" state="empty" error="0" value=""/>
    <Item subindex="3" state="empty" error="0" value=""/>
    <Item subindex="4" state="empty" error="0" value=""/>
    <Item subindex="5" state="initial" error="0" value="17"/>
    <Item subindex="6" state="empty" error="0" value=""/>
    <Item subindex="7" state="empty" error="0" value=""/>
    <Item subindex="8" state="empty" error="0" value=""/>
    <Item subindex="9" state="empty" error="0" value=""/>
    <Item subindex="10" state="empty" error="0" value=""/>
    <Item subindex="11" state="empty" error="0" value=""/>
    <Item subindex="12" state="empty" error="0" value=""/>
    <Item subindex="13" state="empty" error="0" value=""/>
    <Item subindex="14" state="empty" error="0" value=""/>
    <Item subindex="15" state="empty" error="0" value=""/>
   </Variable>
   <Variable variableId="V_DeviceAccessLocks">
    <Item subindex="1" state="initial" error="0" value="false"/>
    <Item subindex="2" state="initial" error="0" value="false"/>
   </Variable>
   <Variable variableId="V_VendorName">
    <Item subindex="0" state="initial" error="0" value="IO-Link Community"/>
   </Variable>
   <Variable variableId="V_VendorText">
    <Item subindex="0" state="initial" error="0" value="http://www.io-link.com"/>
   </Variable>
   <Variable variableId="V_ProductName">
    <Item subindex="0" state="initial" error="0" value="SafetyDevice"/>
   </Variable>
   <Variable variableId="V_ProductID">
    <Item subindex="0" state="initial" error="0" value="SafetyDeviceVariant"/>
   </Variable>
   <Variable variableId="V_ProductText">
    <Item subindex="0" state="initial" error="0" value="Sample IO-Link Safety"/>
   </Variable>
   <Variable variableId="V_SerialNumber">
    <Item subindex="0" state="empty" error="0" value=""/>
   </Variable>
   <Variable variableId="V_HardwareRevision">
    <Item subindex="0" state="empty" error="0" value=""/>
   </Variable>
   <Variable variableId="V_FirmwareRevision">
    <Item subindex="0" state="empty" error="0" value=""/>
   </Variable>
   <Variable variableId="V_ApplicationSpecificTag">
    <Item subindex="0" state="initial" error="0" value="IO-Link Safety"/>
   </Variable>
   <Variable variableId="V_ErrorCount">
    <Item subindex="0" state="empty" error="0" value=""/>
   </Variable>
   <Variable variableId="V_DeviceStatus">
    <Item subindex="0" state="empty" error="0" value=""/>
   </Variable>
   <Variable variableId="V_DetailedDeviceStatus">
    <Item subindex="1" state="empty" error="0" value=""/>
    <Item subindex="2" state="empty" error="0" value=""/>
    <Item subindex="3" state="empty" error="0" value=""/>
    <Item subindex="4" state="empty" error="0" value=""/>
    <Item subindex="5" state="empty" error="0" value=""/>
    <Item subindex="6" state="empty" error="0" value=""/>
    <Item subindex="7" state="empty" error="0" value=""/>
    <Item subindex="8" state="empty" error="0" value=""/>
   </Variable>
   <Variable variableId="V_ProcessDataInput">
    <Item subindex="1" state="empty" error="0" value=""/>
    <Item subindex="2" state="empty" error="0" value=""/>
    <Item subindex="3" state="empty" error="0" value=""/>
    <Item subindex="4" state="empty" error="0" value=""/>
    <Item subindex="5" state="empty" error="0" value=""/>
    <Item subindex="6" state="empty" error="0" value=""/>
    <Item subindex="7" state="empty" error="0" value=""/>
```

```
        <Item subindex="8" state="empty" error="0" value=""/>
        <Item subindex="9" state="empty" error="0" value=""/>
        <Item subindex="10" state="empty" error="0" value=""/>
        <Item subindex="11" state="empty" error="0" value=""/>
        <Item subindex="12" state="empty" error="0" value=""/>
        <Item subindex="13" state="empty" error="0" value=""/>
        <Item subindex="14" state="empty" error="0" value=""/>
        <Item subindex="127" state="empty" error="0" value=""/>
        <Item subindex="128" state="empty" error="0" value=""/>
      </Variable>
      <Variable variableId="V_NonSafetyParameter">
        <Item subindex="0" state="initial" error="0" value="0"/>
      </Variable>
      <Variable variableId="V_FST_DiscrepancyTime">
        <Item subindex="0" state="initial" error="0" value="0"/>
      </Variable>
      <Variable variableId="V_FST_Filter">
        <Item subindex="0" state="initial" error="0" value="0"/>
      </Variable>
      <Variable variableId="V_FSP_Authenticity">
        <Item subindex="1" state="initial" error="0" value="0"/>
        <Item subindex="2" state="initial" error="0" value="0"/>
        <Item subindex="3" state="initial" error="0" value="0"/>
        <Item subindex="4" state="initial" error="0" value="0"/>
      </Variable>
      <Variable variableId="V_FSP_Protocol">
        <Item subindex="1" state="initial" error="0" value="0"/>
        <Item subindex="2" state="initial" error="0" value="1"/>
        <Item subindex="3" state="initial" error="0" value="100"/>
        <Item subindex="4" state="initial" error="0" value="444"/>
        <Item subindex="5" state="initial" error="0" value="0"/>
        <Item subindex="6" state="initial" error="0" value="0"/>
      </Variable>
      <Variable variableId="V_FSP_ParamDescCRC">
        <Item subindex="0" state="initial" error="0" value="444"/>
      </Variable>
    </VariableInstanceData>
  </DeviceItem>
</InvocationInterface>
```

F.9.4 Schema of the TBF

Figure F.18 shows the XML schema of the Temporary Backchannel File.

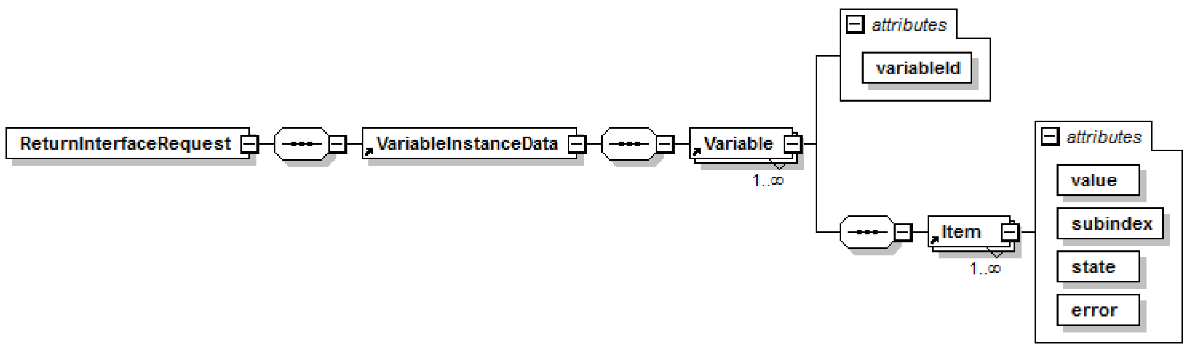


Figure F.18 – XML schema of a TBF

Figure F.18 is based on the XML code as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns="http://www.io-link.com/DTI/2017/02/TBF" xmlns:prim="http://www.io-link.com/DTI/2017/02/Primitives"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" targetNamespace="http://www.io-link.com/DTI/2017/02/TBF">
  <xsd:import namespace="http://www.io-link.com/DTI/2017/02/Primitives" schemaLocation="DTI-Primitives1.0.xsd"/>
  <xsd:element name="VariableInstanceData">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="Variable" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="Variable">
```

```
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element ref="Item" maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:attribute name="variableId" type="xsd:string" use="required"/>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="Item">
      <xsd:complexType>
        <xsd:attribute name="value" type="xsd:string" use="required"/>
        <xsd:attribute name="subindex" use="required">
          <xsd:simpleType>
            <xsd:restriction base="xsd:unsignedShort">
              <xsd:maxInclusive value="255"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:attribute>
        <xsd:attribute name="state" use="required">
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:enumeration value="empty"/>
              <xsd:enumeration value="initial"/>
              <xsd:enumeration value="device"/>
              <xsd:enumeration value="read error"/>
              <xsd:enumeration value="write error"/>
              <xsd:enumeration value="valid"/>
              <!--xsd:enumeration value="changed"/-->
              <!-- should be transferred to device or stored in database before DTI invocation -->
              <!-- could be changed to empty before DTI invocation -->
              <!-- could be changed to empty or valid before DTI invocation -->
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:attribute>
        <xsd:attribute name="error" type="xsd:integer" use="required"/>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="Response">
      <xsd:complexType>
        <xsd:attribute name="value" type="xsd:boolean" use="required"/>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="ReturnInterfaceRequest">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element ref="VariableInstanceData"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="ReturnInterfaceResponse">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element ref="Response"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
    <xsd:group name="ReturnInterface">
      <xsd:choice>
        <xsd:element ref="ReturnInterfaceRequest"/>
        <xsd:element ref="ReturnInterfaceResponse"/>
      </xsd:choice>
    </xsd:group>
  </xsd:schema>
```

F.9.5 Schema of the TAF

The schema of the TAF corresponds to the schema of the TBF in F.9.4.

F.9.6 Schema of DTI primitives

The DTI primitives are defined in the XML code as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns="http://www.io-link.com/DTI/2017/02/Primitives" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.io-link.com/DTI/2017/02/Primitives">
  <xsd:annotation>
```

```
  <xsd:documentation>In this schema, only the necessary types and attributes for DTI are used from the Common Primitives Schema.</xsd:documentation>
  <xsd:appinfo>
    <schemainfo versiondate="20170225"/>
  </xsd:appinfo>
 </xsd:annotation>
 <!-- SIMPLE TYPES -->
 <xsd:simpleType name="IdT">
  <xsd:annotation>
    <xsd:documentation>Base Type for Object identifiers</xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="xsd:string"/>
 </xsd:simpleType>
 <xsd:simpleType name="GuidT">
  <xsd:annotation>
    <xsd:documentation>GUID</xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="xsd:string">
    <xsd:pattern value="\{[0-9A-Fa-f]{8}\-[0-9A-Fa-f]{4}\-[0-9A-Fa-f]{4}\-[0-9A-Fa-f]{4}\-[0-9A-Fa-f]{12}\}"/>
    <xsd:pattern value="[0-9A-Fa-f]{8}\-[0-9A-Fa-f]{4}\-[0-9A-Fa-f]{4}\-[0-9A-Fa-f]{4}\-[0-9A-Fa-f]{12}"/>
  </xsd:restriction>
 </xsd:simpleType>
 <!-- ____________________________________________________________ -->
 <!-- COMPLEX TYPES -->
 <!-- Main Types -->
 <xsd:complexType name="DocumentT">
  <xsd:annotation>
    <xsd:documentation>Type for all top level elements</xsd:documentation>
  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="DocumentInfo" type="DocumentInfoT"/>
  </xsd:sequence>
 </xsd:complexType>
 <xsd:complexType name="DocumentInfoT">
  <xsd:attribute name="Version" type="xsd:string" use="required" fixed="1.1"/>
 </xsd:complexType>
 <!-- ELEMENT DECLARATIONS -->
 <!-- ____________________________________________________________ -->
 <!-- Text Definition Elements-->
 <xsd:complexType name="ObjectT">
  <xsd:annotation>
    <xsd:documentation>Base type</xsd:documentation>
  </xsd:annotation>
 </xsd:complexType>
 <xsd:complexType name="FeatureT">
  <xsd:annotation>
    <xsd:documentation>Base type</xsd:documentation>
  </xsd:annotation>
  <xsd:attribute name="Name" type="xsd:string" use="optional"/>
 </xsd:complexType>
 <xsd:complexType name="ParameterT" mixed="true">
  <xsd:attribute name="Name" type="xsd:string" use="required"/>
 </xsd:complexType>
 <!-- ____________________________________________________________ -->
 <!--Specialized Parameters-->
 <xsd:complexType name="StringParameterT">
  <xsd:complexContent>
    <xsd:extension base="ParameterT">
      <xsd:attribute name="Value" type="xsd:string" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
 </xsd:complexType>
 <!-- ELEMENT DECLARATIONS -->
 <xsd:element name="Document" type="DocumentT">
  <xsd:unique name="OBJ-ID">
    <xsd:selector xpath=".//*"/>
    <xsd:field xpath="@ID"/>
  </xsd:unique>
 </xsd:element>
 <xsd:element name="Object" type="ObjectT"/>
 <xsd:element name="Parameter" type="ParameterT"/>
 <xsd:element name="StringParameter" type="StringParameterT" substitutionGroup="Parameter"/>
 <xsd:element name="Feature" type="FeatureT"/>
 <xsd:simpleType name="ConformanceClassEnumT">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="C1"/>
    <xsd:enumeration value="C2"/>
```

```
      <xsd:enumeration value="C3"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:schema>
```

Annex G
(normative)

Main scenarios of IO-Link Safety

Table G.1 shows main scenarios, the initial key parameters and the associated system activities. Its purpose is to provide a brief overview and it contains references to clauses with detailed descriptions.

Table G.1 – Main scenarios of IO-Link Safety

Scenario	Initial parameters	System activities
OSSD operation	Authenticity = 0 Port = 0 FSP_TechParCRC ≠ 0	1. Modify FST parameter via "USB Master" (option; see 9.4.4.2); 2. Adapt FSP_TechParCRC (see 11.7.8) 3. Plug, validate & play (default)
Commissioning (Test = monitored operation)	Authenticity = 0 Port = 0 FSP_TechParCRC ≠ 0	1. Set FSP_TechParCRC = 0 temporarily (FS-Device and FSP record) 2. Assign Authenticity and Port (FS-Device and FSP record) 3. Assign protocol parameter and FST parameter 4. Run in test mode (Write FSP record: Authenticity + FSP_TechParCRC not evaluated; other protocol parameters adopted; Data Storage upload not required) 5. FS-Master Tool responsible to indicate test mode or to prevent from running in test mode without Tool connection.
Commissioning (Arm and validate)	Authenticity = FSCP code Port = port number FSP_TechParCRC = 0	1. Assign actual FSP_TechParCRC (FS-Device and FSP record) 2. Upload parameters to Data Storage (FSP and FST), see clause 9.4.5.4 3. Run in armed mode (Write FSP record: Authenticity + FSP_TechParCRC compared but not adopted; other protocol parameters adopted), see 11.7.6 4. Validate according to safety manual of FS-Device.
Replacement by FS-Device with factory settings w/o tools	Authenticity = 0 Port = 0 FSP_TechParCRC ≠ 0	1. Download and adopt parameters from Data Storage (FSP and FST) if Authenticity and Port = 0, see 9.4.6.1 and 9.4.6.2 2. Run in armed mode (Write FSP record: Authenticity + FSP_TechParCRC compared but not adopted; other protocol parameters adopted), see 11.7.6 3. Validate according to safety manual of FS-Device.
Misconnection of configured FS-Devices	Authenticity = FSCP code Port = port number FSP_TechParCRC ≠ 0	1. No adoption of downloaded parameters from Data Storage (FSP and FST) since Authenticity and Port ≠ 0 2. Run in armed mode (Write FSP record: Authenticity + FSP_TechParCRC compared; nothing adopted), see 11.7.6 3. Error message: "Misconnection" (0xB003 or 0xB004, see Annex B) 4. Other protocol parameters not adopted.

"Local modification" of FST parameters as described in 9.4.5.4 and Table 13 is possible. However, the FSP_TechParCRC shall be assigned with the help of FS-Master Tool.

Annex H
(normative)

System requirements

H.1 Indicators

H.1.1 General

Indicators for FS-Devices are not mandatory since for example proximity sensors may be too small for LEDs (light emitting diode).

FS-Masters and FS-Devices may be used in a mix of different technologies such as

- Fieldbus safety modules for inputs (e.g. F-DI module) or outputs (e.g. F-DO module);
- Safety devices such as light curtains connected to fieldbuses via FSCPs;
- IO-Link Masters and Devices.

Thus, it is the designer's responsibility to layout the indication of the signal status, modes, or operations for FS-Masters and FS-Devices.

H.1.2 OSSDe

In case an FS-Master port is running in OSSDe mode it behaves similar to an F-DI module port. One possibility of indication is using the same indication as with the SIO mode.

H.1.3 Safety communication

In case an FS-Master port is running in SCL mode, the normal non-safety operation indication can be used also.

H.1.4 Acknowledgment request

A machine is not allowed to restart automatically after a stop. Usually, after repair or clearance, the signal/service "ChFAckReq" is switched ON as specified in 11.10.4 and 11.10.5. It is highly recommended to indicate this signal on an FS-Master port and optionally on FS-Devices where it is likely to cause a trip due to high frequency or duration of exposure to a safety function.

H.2 Installation guidelines, electrical safety, and security

IO-Link installation guidelines shall be considered (see [21]).

Only FS-Masters and FS-Devices providing a short form functional safety assessment report according to IEC 61508 or ISO 13849-1 together with a certificate of the assessment body are permitted. The short form report shall indicate all considered clauses and paragraphs of the used relevant standards and the corresponding assessment results.

Wireless connection between FS-Master and FS-Device is only permitted if interdependency with other wireless connections can be precluded, for example via inductive couplers.

No components in the link between FS-Master and FS-Device are permitted that are storing, inserting, or delaying messages.

Manufacturer/vendor of FS-Masters and/or FS-Devices shall define installation constraints for the operation of OSSD devices or FS-Devices in OSSDe mode within their safety manuals.

Requirements of IEC 61010-2-201 (see [4]) and IEC 60204-1 with respect to electrical safety (SELV/PELV) shall be observed.

The zones and conduit concept of IEC 62443 applies for security and/or the rules of the applicable FSCP system.

H.3 Safety function response time

Safety manuals of FS-Master shall provide information on how to determine the safety function response time for OSSDe and for communication modes.

H.4 Duration of demands

Short demands of FS-Devices may not trip a safety function due to its chain of independent communication cycles across the network. Therefore, a demand shall last for at least two SCL (SPDU) cycles.

H.5 Maintenance and repair

FS-Devices can be replaced at runtime. Restart of the corresponding safety function is only permitted if there is no hazardous process state and after an operator acknowledgment.

H.6 Safety manual

FS-Masters and FS-Devices shall provide safety manuals according to the relevant national and international standards, for example IEC 61784-3-0, Edition 3.

Manufacturer/vendor of FS-Masters and/or FS-Devices shall specify appropriate mitigation means in the safety manual for the deployment of IO-Link Safety components in harsh industrial environment such as in EMC zones B and C according to IEC 61131-2.

Manufacturer/vendor of FS-Masters and/or FS-Devices shall define all constraints for the operation of OSSD devices or FS-Devices in OSSDe mode within their safety manuals.

Manufacturer/vendor of FS-Masters and/or FS-Devices shall define all constraints for the operation of FS-Devices in communication mode within their safety manuals such as limitations with respect to storing elements, inductive or optical couplers, and alike.

Manufacturer/vendor of FS-Masters and/or FS-Devices shall define the maintenance rules with respect to the PFH-Monitor (see Table 30).

Annex I
(normative)

Assessment

I.1 General

Functional safety assessments can only be performed if hardware and software are provided. Thus, the actual assessment of IO-Link Safety can only comprise a concept approval as a precondition for the conformity of implementations. This can result in precertified development kits to save time and effort.

I.2 Safety policy

In order to prevent and protect the manufacturers and vendors of FS-Masters and FS-Devices from possibly misleading understandings or wrong expectations and gross negligence actions regarding safety-related developments and applications the following shall be observed and explained in each training, seminar, workshop and consultancy.

- Any non-safety-related device automatically will not be applicable for safety-related applications just by using fieldbus or IO-Link communication and a safety communication layer.
- In order to enable a product for safety-related applications, appropriate development processes according to safety standards shall be observed (see IEC 61508, IEC 60204-1, IEC 62061, ISO 13849) and/or an assessment from a competent assessment body shall be achieved.
- The manufacturer of a safety product is responsible for the correct implementation of the safety communication layer technology, the correctness and completeness of the product documentation and information.
- Additional important information about current corrigendums through concluded change requests shall be considered for implementation and assessment. This information can be obtained from the IO-Link Community.

I.3 Obligations

As a rule, the international safety standards are accepted (ratified) globally. However, since safety technology in automation is relevant to occupational safety and the concomitant insurance risks in a country, recognition of the rules pointed out here is still a sovereign right. The national "Authorities" decide on the recognition of assessment reports.

I.4 Concept approval

For the approval of the safety concepts of IO-Link Safety the following has been provided by the community:

- This document (specification of IO-Link Safety)
- Documentation of the modelling, the model checking, and the simulation including fault injection of the IO-Link safety communication layer (SCL)
- Document "Safety considerations" with Functional Safety Management, calculation of relevant Residual Error Rates, and software tool chain FMEA
- Document "Document Management and Working Group rules"

Annex J
(normative)

Details of "Classic" port class B

J.1 "Classic" power supply option

The IO-Link connection system provides dedicated power lines in addition to the signal line as shown in Figure J.1. The communication section of a Device/FS-Device shall always be powered by the Master/FS-Master using the power lines defined in the 3-wire connection system (Power1) in [1]. Its maximum supply current is defined in 5.9 and Table 7.

The technology/application part of a Device/FS-Device can be powered by one of three ways:

- via the power lines of the 3-wire connection system (class A ports), using Power1;
- via the extra power lines of the 5-wire connection system (class B ports), using an extra power supply (Power2) at the Master/FS-Master;
- via a power supply at the Device/FS-Device (design specific) that shall be nonreactive to Power 1.

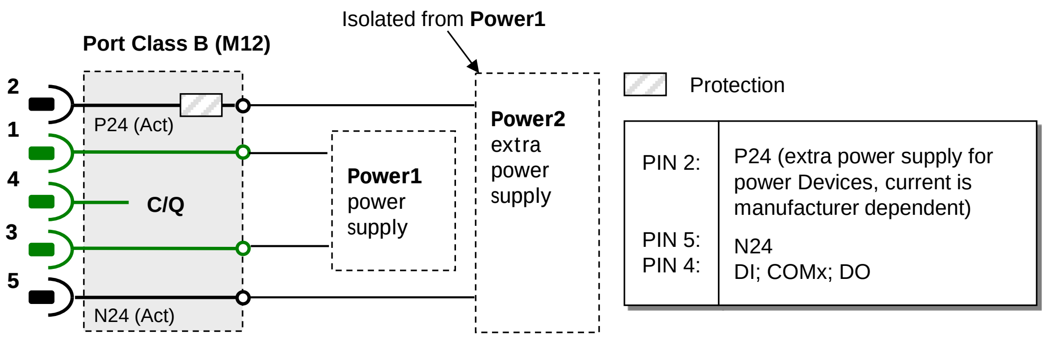


Figure J.1 – "Classic" port Class B definitions

Figure J.1 shows also an extra power supply (Power2) intended for Devices/FS-Devices requiring more supply current for their individual technology/application such as actuators. Class B ports shall be marked to distinguish from Class A ports due to risks deriving from incompatibilities on pin 2 and pin 5.

The maximum current available from this extra power supply is specified in Table J.1.

Table J.1 – Electric characteristic of Power2

Property	Designation	Minimum	Typical	Maximum	Unit	Remark
$VPN24_M$	Extra DC supply voltage for Devices	20a)	24	30	V	
$IPN24_M$	Extra DC supply current for Devices	1,6b)	n/a	3,5c)	A	

a) A minimum voltage shall be guaranteed for testing at maximum recommended supply current. At the FS-Device side 18 V shall be available in this case.

b) Minimum current in order to guarantee a high degree of interoperability.

c) The recommended maximum current for a wire gauge of 0,34 mm^2 and standard M12 connector is 3,5 A. Maximum current depends on the type of connector, the wire gauge, maximum temperature, and simultaneity factor of the ports (check user manual of a Master).

J.2 Rules

As a general rule for non-safety Devices it is recommended not to consume more than the minimum current a Master shall support (see Table 6 in [1]) in order to achieve easiest handling ("plug & play") of IO-Link Master/Device systems without inquiries, checking, and calculations.

Whenever the Device requires more than the minimum current the capabilities of the respective Master port and of the cabling shall be checked.

FS-Devices should follow this recommendation also. However, 5.9 and Table 7 show mitigation means for FS-Devices and FS-Masters to certain extend.

In general, the requirements of Devices/FS-Devices shall be checked whether they meet the available capabilities of the Master/FS-Master. The simultaneity factor for the Master/FS-Master ports shall be observed.

Power2 on port class B shall meet the following requirements

- electrical isolation of Power2 from Power1;
- degree of isolation according to IEC 60664 (clearance and creepage distances);
- electrical safety (SELV) according to IEC 61010-2-201:2017;
- direct current with P24 (+) and M24 (-);
- EMC tests shall be performed with maximum ripple and load switching
- Device shall continue communicating correctly even in case of failing Power2

Figure J.2 shows a possible layout of a cable for port Class B operation.

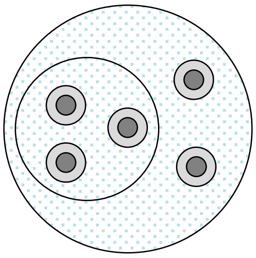


Figure J.2 – Possible layout of cable with Power1 and Power2

In case of functional safety, the following standards shall be observed whenever applicable:

- ISO 13849-2:2012
- IEC 60204-1
- VDE 0298, Part 4:2013 (Current ratings for flexible cables)
- VDE 0891-1:1990 (Use of cables and insulated wires for telecommunication systems and information processing systems; general directions)

Annex K
(normative)

Test of FS-Master and FS-Device

This part will be provided at a later date.

Bibliography

[1] IO-Link Community, *IO-Link Interface and System*, V1.1.2, July 2013, Order No. 10.002

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

[3] IEC 61784-3 Ed 3.0: *Industrial communication networks – Profiles – Part 3: Functional safety fieldbuses – General rules and profile definitions*

[4] IEC 61010-2-201:2017*, Safety requirements for electrical equipment for measurement, control and laboratory use – Part 2-201: Particular requirements for control equipment*

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

[6] Bruce P. Douglass, *Real Time UML – Advances in the UML for Real-Time Systems*, 3rd Edition, Addison-Wesley, ISBN 0-321-16076-2

[7] Chris Rupp, Stefan Queins, Barbara Zengler, *UML 2 glasklar – Praxiswissen für die UML-Modellierung*. Hanser-Verlag, 2007, ISBN 978-3-446-41118-0

[8] IEC/TR 62390, *Common Automation Device – Profile Guideline*

[9] IO-Link Community, *IO Device Description (IODD)*, V1.1, July 2011, Order No. 10.012

[10] IO-Link Community, *IO-Link Test*, V1.1.2, July 2014, Order No. 10.032

[11] IO-Link Community, *IO-Link Safety (Single Platform) – Requirements, Use Cases, and Concept Baseline,* V1.0, November 2014, Order No. 10.062

[12] Position Paper CB24I, *Classification of Binary 24V Interfaces – Functional Safety aspects covered by dynamic testing*, Edition 2.0.1: https://www.zvei.org/verband/fachverbaende/fachverband-automation/schaltgeraete-schaltanlagen-industriesteuerungen/

[13] IO-Link Community, *IO-Link Profile BLOB & FW-Update*, V1.0, June 2016, Order No. 10.082

[14] Klaus Grimmer, *AIDA_IP-67-Safety_Positionspapier*, June 27th, 2013

[15] FDT Joint Interest Group, *FDT 2.0 – Specification,* V1.0, Order No. 0001-0008-000

[16] FDT Joint Interest Group, *FDT for IO-Link – Annex to FDT Specification – Based on FDT Specification Version 1.2.1,* V1.0, Order No. 0002-0013-000

[17] IEC 62453 series: *Field device tool (FDT) interface specification*

[18] CRC signature calculator for a seed value of "0": https://www.ghsi.de/CRC/index.php?Polynom=10100111010101011&Message=1%0D%0A (Date: 29-Jan-2017)

[19] Philip Koopman, *Cyclic Redundancy Code (CRC) Polynomial Selection For Embedded Networks*, The International Conference on Dependable Systems and Networks, DSN-2004.

[20] Philip Koopman, *Best CRC Polynomials*, https://users.ece.cmu.edu/~koopman/crc/ (Date: 29-Jan-2017)

[21] IO-Link Community, *IO-Link Design Guideline*, V1.0, November 2016, Order No. 10.912

____

IO-Link Community
Haid-und-Neu-Str. 7
76131 Karlsruhe
Germany
Phone: +49 (0) 721 / 96 58 590
Fax: +49 (0) 721 / 96 58 589
e-mail: info@io-link.com
http://www.io-link.com/