

IO-Link Profile BLOBs & FW-Update

Specification

**Version 1.2
September 2024**

Order No: 10.082

File name: **IOL-Profile_Firmware-Update_V12_10082_Sep24.docx**

This specification has been prepared by the IO-Link Firmware Update profile group.

Any comments, proposals, requests on this document are appreciated. Please use www.io-link-projects.com for your entries and provide name and email address.

Login: *IOL-SM-Profile*

Password: *Report*

Important notes:

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

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

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

NOTICE:


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

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

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

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

USE OF TRADEMARKS:

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

PNO is the owner of several registered trademarks, such as PROFIBUS®, PROFINET®, omlox®, IO-Link®, MTP® and others. More detailed terms for the use can be found on the web page www.profibus.com. Please select buttons "Downloads / Presentations & logos". In some cases, PNO is the licensee of registered trademarks owned by third parties and which may be relevant in regard with products compliant to this document.

PNO shall always be the sole entity that may authorize developers, suppliers and sellers of hardware and software to use certification marks, trademarks or other special designations to indicate compliance with a PNO specification. Products developed using a PNO specification may claim compliance or conformance with a PNO specification only if the hardware and/or software satisfactorily meets the certification requirements set by PNO. Products that do not meet these requirements may claim only that the product was based on a PNO specification and must not claim compliance or conformance with a PNO specification.

COPYRIGHT

Copyright © 2024 PROFIBUS Nutzerorganisation e.V.

Any unauthorized use of this publication may violate Copyright Law, Trademark Law and other legal regulations. This document contains information which is protected by Copyright. No part of this work covered by Copyright herein may be reproduced or used in any form or by any means -graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems--without permission of the publisher.

Publisher:

IO-Link Community

c/o PROFIBUS Nutzerorganisation e.V.

Ohiostrasse 8

76149 Karlsruhe

Germany

Phone: +49 721 / 98 61 97 0

Fax: +49 721 / 98 61 97 11

E-mail: info@io-link.com

Web site: www.io-link.com

LICENSE AGREEMENT

1. License

1.1 Subject of this license agreement is this document issued by the Licensor, in electronic form. If applicable, also software may be provided.

1.2 The user of this document (Licensee) acquires the license solely from PROFIBUS Nutzerorganisation e.V., having its principal place of business in Karlsruhe, Germany (hereinafter referred to as "Licensor").

1.3 This document is not an industrial standard acknowledged by any standardization body or otherwise and may be further enhanced.

2. Rights and Duties of Licensee

2.1 Licensor hereby grants to Licensee the right to use this document exclusively for developing and supporting products compliant with this document. Licensee may copy this document for this purpose and for data backup purposes.

2.2 Licensee shall not be entitled to modify, decompile, reverse engineer or extract any individual parts of this document, unless this is permitted by mandatory Copyright Law. Furthermore, Licensee shall not be entitled to remove any alphanumeric identifiers, trademarks or copyright notices from this document and, insofar as Licensee is entitled to make copies of this document, Licensee shall copy them without alteration.

2.3 Licensee is not entitled to copy and redistribute this document to any third party, except for "Have Made" purposes. All copies must be obtained on an individual basis, directly from the website www.de.profibus.com or www.profibus.com or upon request from the Licensor.

3. Liability of Licensor

3.1 Licensor shall have no obligation to enhance the document and shall assume no liability in case the document or future versions thereof shall not be approved as an industrial standard.

3.2 Licensor's liability for defects as to quality or title of this document, especially in relation to the correctness or absence of defects or the absence of claims or third-party rights or in relation to completeness, usability and/or fitness for purpose are excluded, except for cases involving gross negligence, willful misconduct or fraudulent concealment of a defect.

3.3 Any further liability is excluded unless required by law, e.g. in cases of personal injury or death, willful misconduct, gross negligence, or in case of breach of fundamental contractual obligations. The damages in case of breach of fundamental contractual obligations is limited to the contract-typical, foreseeable damage if there is no willful misconduct or gross negligence.

4. Place of Jurisdiction and Applicable Law

4.1 The sole place of jurisdiction shall be the principal place of business of Licensor.

4.2 All relations arising out of the contract shall be governed by the substantive law of Germany, to the exclusion of the United Nations Convention on Contracts for the International Sale of Goods (CISG).

Conventions:

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

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

Revision log:

Date	Version	Editor	Description	CRs covered
20-Jun-2016	WD1.0.0	WS	Final version after CR resolution session	HO (HWIDKey)
18-Mar-2019	WD1.1.0	WS	First draft considering Change Requests from the IO-Link project database	CR-IDs 1 to 41
05-Apr-2019	WD1.1.0	WS	Corrections by WR, Clarification CRC calculation, incorporation of test cases	
15-Apr-2019	WD1.1.0	WS	CRC again, shorter headings, code snippets	
26-Apr-2019	WD1.1.0	WS	Finalization: Typos; ready for Community review	
17-Sep-2019	WD1.1.0	WS	Final version after community review (first part)	
18-Sep-2019	WD1.1.0	WS	Final version after community review (second part)	
20-Sep-2019	WD1.1.0	WS	Final version after phone conference WR+HO+WS	
29-Sep-2019	WD1.1.0	WS	Final version after editorial comments from HO and WR	
08-Oct-2019	WD1.1.0	WS/WR	Final version after editorial comments from Balluff	
13-Aug-2024	WD1.1.1	OW	First draft after CR resolution session.	CR-IDs 75, 79, 80, 81, 82, 83, 102, 105
14-Aug-2024	WD1.1.1	OW	Second draft after working group review comments.	
Aug 2024	WD1.1.1	OW	Renaming of IODD IDs to be consistent with IODD snippets.	CR IDs 58, 59, 60, 61, 62, 63, 65, 66, 69, 70, 71, 72, 73, 74
Sep 2024	WD1.2	OW	Final version after editorial comments from TMG	CR 108

CONTENTS

0	Introduction	13
0.1	General.....	13
0.2	Patent declaration.....	13
1	Scope.....	14
2	Normative references	14
3	Terms, definitions, symbols, abbreviated terms and conventions	14
3.1	Terms and definitions.....	14
3.2	Symbols and abbreviated terms	17
3.3	Conventions.....	17
3.3.1	Behavioral descriptions.....	17
3.3.2	Memory and transmission octet order	18
4	Overview of the profile.....	18
5	Profile characteristic.....	20
6	Binary Large Objects (BLOBs).....	20
6.1	Purpose and system positioning.....	20
6.2	Components involved.....	21
6.2.1	PLC / Host.....	21
6.2.2	Tools	21
6.2.3	Master	22
6.2.4	Device	22
6.3	Data objects within BLOBs.....	22
6.3.1	Types of objects	22
6.3.2	Securing measures	22
6.4	ISDU as transport vehicle	22
6.4.1	General	22
6.4.2	Diagnosis – EventCodes.....	23
6.4.3	Acknowledgments.....	23
6.5	BLOB parameters and transfer.....	23
6.5.1	Profile related Index space	23
6.5.2	BLOB_ID	23
6.5.3	BLOB_CH.....	24
6.6	Protocol of BLOB transmission.....	28
6.6.1	Device BLOB state machine	28
6.6.2	Host BLOB state machine.....	30
6.7	Access conflicts	32
6.7.1	Overview	32
6.7.2	Concurrent access of tools	32
6.7.3	Access blocking.....	32
7	Firmware-Updates	33
7.1	Purpose and system positioning.....	33
7.2	Components involved.....	33
7.2.1	Master (PCDT) or PC tools	33
7.2.2	Master	33
7.2.3	Device	33
7.2.4	FW-Update file	33

7.3	Use cases	34
7.3.1	Update Firmware	34
7.3.2	Upgrade	35
7.3.3	Downgrade	35
7.3.4	Upload firmware	36
7.4	File formats	36
7.4.1	General	36
7.4.2	Creation of file	36
7.4.3	File naming convention	36
7.4.4	Meta data file description	37
7.5	Bootload management	40
7.5.1	Main activities	40
7.5.2	Device identification	40
7.5.3	Acquisition of the FW-Update file	40
7.5.4	Verification of FW-Update file compatibility	41
7.5.5	Password entry and check (optional)	41
7.5.6	FW-Update via bootloader	42
7.5.7	Finalization and activation	42
7.6	Definitions and constraints	43
7.6.1	Initial Device operation (OPERATE/PREOPERATE)	43
7.6.2	Bootloader	43
7.6.3	IODD for Bootload mode	43
7.6.4	M-sequence types	43
7.6.5	DeviceID versus Boot_DeviceID	43
7.6.6	VendorID	43
7.6.7	FW-Update specific parameters	43
7.6.8	FW-Update specific SystemCommands	44
7.6.9	Unlocking sequence	44
7.6.10	Required BLOB_IDs	45
7.7	FW-Update protocol	45
7.7.1	Protocol layers	45
7.7.2	Device FW-Update state machine	45
7.7.3	Reception of binaries and flashing activity	48
7.7.4	Master behavior	48
7.7.5	Tool FW-Update state machine	48
7.7.6	Sequence charts	51
7.8	Validation/responsibility	52
7.9	Common look and feel	52
7.9.1	Graphical user interface (GUI) and functions	52
7.9.2	Multi-language terms	53
7.9.3	Error displays of FW-Update tools	53
7.9.4	Indications	54
7.10	Recommended strategies	54
7.10.1	Design aspects	54
7.10.2	Distribution to multiple destinations within the Device	57
7.10.3	Compatibility levels	57
Annex A	(normative) IODD extensions	58
A.1	Overview	58
A.2	Binary Large Objects (BLOBs)	58

A.3	FW-Update	59
A.4	IODD Checker V1.1.12 or higher.....	60
Annex B	(normative) Calculation of CRC signatures	62
B.1	Overview of CRC-32 signatures	62
B.2	Implementation considerations.....	62
B.2.1	Overview	62
B.2.2	Bit shift algorithm (32 bit).....	62
B.2.3	Octet shift and Look-up tables (32 bit)	63
B.2.4	Seed value	64
Annex C	(informative) Compression, authentication, and encryption	65
C.1	Compression.....	65
C.1.1	General	65
C.1.2	LZRW (Lempel-Ziv-Ross Williams)	65
C.1.3	Deflate (Lempel-Ziv 77 + Huffman Coding)	65
C.1.4	LZO (Lempel-Ziv-Oberhumer).....	65
C.2	Authentication via signatures	66
C.3	Encryption of FW-Update files.....	67
Annex D	(informative) Performance estimations.....	68
D.1	Assumptions	68
D.2	Estimated time for a BLOB transfer.....	68
D.3	Required number of ISDU transfers	68
D.4	Number of M-sequences per ISDU	68
D.5	Recommended Master cycle times	69
D.6	Conclusions	69
Annex E	(normative) Profile specific test cases and tools	71
E.1	Concept, conventions, and preconditions	71
E.1.1	Concept.....	71
E.1.2	Conventions	71
E.1.3	Test exceptions	71
E.2	Device implementations of BLOB transfer	72
E.2.1	Reserved BLOB_IDs for Device test	72
E.2.2	Complete BLOB read (positive test).....	73
E.2.3	Complete BLOB write (positive test)	75
E.2.4	Write BLOB with invalid CRC.....	77
E.2.5	Write access to read-only parameter	78
E.2.6	Illegal BLOB_CH command	79
E.2.7	Invalid BLOB_ID	80
E.2.8	Read access on BLOB_CH in state "Idle_0"	81
E.2.9	Invalid BLOB_Start during BLOB read	82
E.2.10	Invalid BLOB_Start during BLOB write	84
E.2.11	BLOB Abort during BLOB read	86
E.2.12	BLOB_Abort during BLOB write	87
E.2.13	Inadmissible write to BLOB_CH during BLOB read	89
E.2.14	Inadmissible read to BLOB_CH during BLOB write	91
E.2.15	Write BLOB_Segment at WaitOnCRC and WaitOn_BLOB_complete	93
E.2.16	BFlowCtrl overflow during BLOB write	94
E.2.17	BFlowCtrl nok during BLOB write.....	95
E.2.18	Profile specific IODD content.....	97

E.2.19	Reserved indices	99
E.3	Host implementation of BLOB transfer	100
E.3.1	Reserved BLOB_IDs for host application test purposes	100
E.3.2	Complete BLOB write (positive Test)	101
E.3.3	Complete BLOB read (positive Test)	102
E.3.4	Unsupported BLOB_ID read	103
E.3.5	Unsupported BLOB_ID write	104
E.3.6	Check active BLOB transfer	105
E.3.7	Error handling BLOB Read	106
E.3.8	FLOW control error handling	107
E.3.9	Error handling BLOB Write	108
E.3.10	Error handling CRC read	109
E.3.11	Error handling CRC write	110
E.3.12	Write segments of equal length at ISDUs of maximum size	111
E.3.13	Write segments of unequal length at ISDUs of maximum size	112
E.3.14	Write segments of equal length at ISDUs of limited size	113
E.3.15	Write with segments of unequal length at ISDUs of limited size	114
E.3.16	BLOB size is too big for Device	115
E.4	Device implementation of Firmware Update	116
E.4.1	Test Modules	116
E.4.2	Complete firmware update with/without password	118
E.4.3	Unlock with a wrong password	120
E.4.4	Inadmissible BM_UNLOCK_F	121
E.4.5	Inadmissible BM_UNLOCK_T	122
E.4.6	Inadmissible BM_ACTIVATE	123
E.4.7	Inadmissible BM_UNLOCK_S and restart unlock sequence	124
E.4.8	Wrong firmware-image with & without password	126
E.4.9	Random power fail or reset with & without password	127
E.4.10	Error cases during BLOB transfer	128
E.4.11	Firmware Download specific IODD content	129
E.5	Host implementation of Firmware Update	132
E.5.1	Required test firmware files	132
E.5.2	Write w/o password and Device in TFW mode	133
E.5.3	FW-Update w/o password Device in bootloader mode	134
E.5.4	FW-Update with password	135
E.5.5	Incorrect VendorID	136
E.5.6	Incorrect VendorID and Device in bootloader mode	137
E.5.7	No matching HW_ID_KEY in the IOLFW file	138
E.5.8	No matching HW_ID_KEY in the IOLFW file and Device in bootloader mode	139
E.6	Test Tools	140
E.6.1	Device Tester	140
E.6.2	Firmware Update and BLOB test Device (FWBD)	141
E.6.3	BLOB Host Application Tester System	141
E.6.4	Firmware Update Application Test System	142
E.6.5	Random Data	143
Annex F (informative)	Information on conformance testing of profile Devices	146
Bibliography	147

Figure 1 – Example of a nested state	17
Figure 2 – Superstate with regions (And-state)	18
Figure 3 – Memory and transmission octet order	18
Figure 4 – IO-Link operations including BLOB transfer and FW-Update	19
Figure 5 – Device architecture including BLOB and FW-Update	20
Figure 6 – BLOB transmission system	21
Figure 7 – Example structure of an ISDU for BLOB transfer (BLOB_CH).....	22
Figure 8 – Structure of BLOB_ID.....	23
Figure 9 – Header of BLOB_CH	24
Figure 10 – Structure of "BLOB_Info_Read" block	25
Figure 11 – Structure of "BLOB_Info_Write" block	25
Figure 12 – Structure of "BLOB_Segment"	26
Figure 13 – Structure of "BLOB_Last"	26
Figure 14 – Structure of "BLOB_CRC"	26
Figure 15 – Structure of command "BLOB_Abort"	27
Figure 16 – Structure of command "BLOB_Start"	27
Figure 17 – Structure of command "BLOB_Finish"	27
Figure 18 – Device BLOB state machine	28
Figure 19 – Host BLOB state machine	30
Figure 20 – FW-Update transmission system	33
Figure 21 – Use cases of the FW-Update procedure	34
Figure 22 – FW-Update file naming convention	36
Figure 23 – Example of a FW-Update file name	36
Figure 24 – Meta data XML structure	37
Figure 25 – Identification activity.....	40
Figure 26 – FW-Update file acquisition and check.....	41
Figure 27 – Verification of compatibility	41
Figure 28 – Password check	42
Figure 29 – Finalization and activation	42
Figure 30 – Device FW-Update state machine	46
Figure 31 – BLOB and flashing activity	48
Figure 32 – Tool FW-Update state machine	49
Figure 33 – Sequence chart of the FW-Update procedure	51
Figure 34 – Exemplary illustration of the GUI functions	52
Figure 35 – Layer structures of Master and Device	55
Figure B.1 – Bit shift algorithm in "C" language.....	62
Figure B.2 – CRC-32 signature calculation using a lookup table	63
Figure D.1 – BLOB transfer timings for COM3.....	70
Figure D.2 – BLOB transfer timings for COM2.....	70
Figure E.1 – BLOB and Firmware Update Device Tester system	140
Figure E.2 – BLOB Host Application Tester System	142
Figure E.3 – Firmware Update Application Tester System	143
Figure E.4 – C code sample of the "iol_rand()" function	143

Table 1 – Index assignment of the BLOB parameters.....	23
Table 2 – Coding of BLOB_ID	23
Table 3 – Coding of the header of BLOB_CH	24
Table 4 – State transition tables of the Device BLOB state machine	29
Table 5 – State transition tables of the Host BLOB state machine	31
Table 6 – Use Case reference table	34
Table 7 – Use cases of possible modifications	35
Table 8 – Items of the file naming convention	36
Table 9 – Definition of the root elements	38
Table 10 – Definition of elements in DocumentInfo	38
Table 11 – Definition of elements in MetaInfo	39
Table 12 – Device parameters reserved for FW-Update	43
Table 13 – Coding of BootmodeStatus	44
Table 14 – Coding of FW-Update SystemCommands	44
Table 15 – BLOB_IDs for FW-Update	45
Table 16 – State transition tables of the Device bootloader state machine	46
Table 17 – State transition tables of the Tool FW-Update state machine	50
Table 18 – Human interface terms	53
Table 19 – Error displays of FW-Update tools	54
Table 20 – Affected items of the Direct Parameter page 1	55
Table 21 – Affected MasterCommands	55
Table 22 – Affected ISDUs	56
Table 23 – Affected SystemCommands	56
Table 24 – Affected services	57
Table B.1 – CRC-32 generator polynomial	62
Table B.2 – Definition of variables used in Figure B.1	63
Table B.3 – Definition of variables used in Figure B.2	63
Table B.4 – Lookup table for CRC-32 signature calculation (hexadecimal)	63
Table D.1 – Number of M-sequences per ISDU	69
Table D.2 – Recommended cycle times (t_{CYC})	69
Table E.1 – Predefined test BLOB_IDs	71
Table E.2 – Exempted test cases for Devices	71
Table E.3 – Reserved BLOB_IDs for the Device test	72
Table E.4 – Complete BLOB read	73
Table E.5 – Complete BLOB write	75
Table E.6 – Write BLOB with invalid CRC	77
Table E.7 – Write access to read-only parameter	78
Table E.8 – Illegal BLOB_CH command	79
Table E.9 – Invalid BLOB_ID	80
Table E.10 – Read access on BLOB_CH in state "Idle_0"	81
Table E.11 – Invalid BLOB_Start during BLOB read	82
Table E.12 – Invalid BLOB_Start during BLOB write	84

Table E.13 – BLOB Abort during BLOB read	86
Table E.14 – BLOB_Abort during BLOB write.....	87
Table E.15 – Inadmissible write to BLOB_CH during BLOB read.....	89
Table E.16 – Inadmissible read to BLOB_CH during BLOB write.....	91
Table E.17 – Write BLOB_Segment at WaitOnCRC and WaitOn_BLOB_complete	93
Table E.18 – BFlowCtrl overflow during BLOB write	94
Table E.19 – BFlowCtrl nok during BLOB write	95
Table E.20 – Profile specific IODD content	97
Table E.21 – Reserved indices	99
Table E.22 – Reserved BLOB_IDs for host application test purposes.....	100
Table E.23 – Test sequences for host application tests.....	100
Table E.24 – Complete BLOB write (positive Test).....	101
Table E.25 – Complete BLOB read (positive Test)	102
Table E.26 – Unsupported BLOB_ID read	103
Table E.27 – Unsupported BLOB_ID write	104
Table E.28 – Check active BLOB transfer	105
Table E.29 – Error handling BLOB Read.....	106
Table E.30 – FLOW control error handling	107
Table E.31 – Error handling BLOB Write.....	108
Table E.32 – Error handling CRC read.....	109
Table E.33 – Error handling CRC write	110
Table E.34 – Write segments of equal length at ISDUs of maximum size	111
Table E.35 – Write segments of unequal length at ISDUs of maximum size	112
Table E.36 – Write segments of equal length at ISDUs of limited size.....	113
Table E.37 – Write with segments of unequal length at ISDUs of limited size.....	114
Table E.38 – BLOB size is too big for Device	115
Table E.39 – Complete Unlock Sequence	116
Table E.40 – Wait for BootmodeStatus = 0x01	116
Table E.41 – Wait for BootmodeStatus = 0x00	117
Table E.42 – Wait for BM_ACTIVATE = successful	117
Table E.43 – Complete firmware update with/without password	118
Table E.44 – Unlock with a wrong password	120
Table E.45 – Inadmissible BM_UNLOCK_F.....	121
Table E.46 – Inadmissible BM_UNLOCK_T.....	122
Table E.47 – Inadmissible BM_ACTIVATE	123
Table E.48 – Inadmissible BM_UNLOCK_S and restart unlock sequence.....	124
Table E.49 – Wrong firmware-image with & without password.....	126
Table E.50 – Random power fail or reset with & without password	127
Table E.51 – Error cases during BLOB transfer.....	128
Table E.52 – Firmware Download specific IODD content.....	129
Table E.53 – Required test firmware files.....	132
Table E.54 – Write w/o password and Device in TFW mode.....	133
Table E.55 – FW-Update w/o password Device in bootloader mode	134

Table E.56 – FW-Update with password.....	135
Table E.57 – Incorrect VendorID	136
Table E.58 – Incorrect VendorID and Device in bootloader mode.....	137
Table E.59 – No matching HW_ID_KEY in the IOLFW file.....	138
Table E.60 – No matching HW_ID_KEY in the IOLFW file & Device in bootloader mode	139
Table E.61 – System requirements for the Device Tester	140
Table E.62 – FWBD result states	141
Table E.63 – Pre-calculated random numbers.....	144

0 Introduction

0.1 General

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

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

This document specifies the common additional protocol means for the transfer of Binary Large Objects (BLOB) and in particular the means for firmware-updates of Devices. Its previous version V1.1 is no more valid.

This document follows the IEC 62390 [3] to a certain extent. Terms of general use are defined in IEC 61131-1 or in [4]. Specific SDCI terms are defined in this part.

0.2 Patent declaration

The IO-Link Community draws attention to the fact that it is claimed that compliance with this document may involve the use of patents concerning FW-Update in case of functional safety devices as follows, where the [xx] notation indicates the holder of the patent right:

EP 1532494 B1	[PI]	Safety control system for fail-safe control of safety-critical processes and method for running a new operating program therein
---------------	------	---

IO-Link Community takes no position concerning the evidence, validity and scope of this patent right.

The holders of the patent rights have assured the IO-Link Community that they are willing to negotiate licences either free of charge or under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statements of the holders of these patent rights are registered with the IO-Link Community.

Information may be obtained from:

[PI]	Pilz GmbH, Felix-Wankel-Straße 2, 73760 Ostfildern, Deutschland/Germany
------	---

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.

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

PROGRAMMABLE CONTROLLERS —

Profile for the transfer of BLOBs and firmware-updates within systems according to IEC 61131-9

1 Scope

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

This document specifies the common means for the transfer of Binary Large Objects (BLOBs) between tools, programmable logic controllers (PLC), Masters and Devices as well as associated identification rules for BLOBs, securing measures for uploads, index definitions, definitions for transfer segmentation (> ISDU size), and access conflict resolutions. It is supplemented by recommendations for the encryption and compression of BLOBs.

This document specifies also the common extra protocol means for Device firmware-updates between tools, Master and Devices and the associated use cases; the necessary equipment set-up; commands and messages; file formats; identification, authentication, and validation issues; DeviceID and version rules; parameter versions and data storage rules. It is supplemented by recommendations on encryption and compression of firmware-update files, authentication via signatures, and distribution to multiple destinations within a Device.

This document contains specific IODD extensions according to this profile and instructions for an extension of the IODD-Checker.

This document contains specific test cases for the conformity testing of tools, Masters, and Devices according to this profile.

2 Normative references

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

IEC 61131-2, *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)*

3 Terms, definitions, symbols, abbreviated terms and conventions

3.1 Terms and definitions

For the purposes of this document, the following terms and definitions in addition to those given in IEC 61131-9 apply. For the convenience of readability, major terms and definitions of IEC 61131-9 are repeated.

3.1.1**bidirectional**

transfer of data from a tool via a Master to the Device and vice versa

3.1.2**binary large object****BLOB**

amount of coherent data to be transferred to or from the Device

3.1.3**Bootloader**

Device application responsible for *unlocking* existing firmware, handling of FW-Update binaries and permanent storage, e.g. flashing

Note 1 to entry: The technology specific Bootloader application and the IO-Link communication can be cut down to the minimum functionality required for the purpose of bootloading in case of an unsuccessful update

3.1.4**D-BLOB_Trans_Layer**

communication protocol in a Device application for the transmission of data larger than the ISDU length

NOTE 1 to entry: Counterpart is either a P-BLOB_Trans_Layer or T-BLOB_Trans_Layer

3.1.5**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.

3.1.6**download**

transfer of data from a tool via a master to the Device

3.1.7**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.

3.1.8**firmware**

entire nonvolatile software of a Device consisting of the technology specific application including the *Bootloader* and the communication stack

Note 1 to entry: The technology specific application comprises for example measuring or control algorithms, hardware drivers, and local user interfaces

3.1.9**FW-Update application**

computer software *tool* for the purpose of updating a Device's *firmware*

3.1.10**HW_ID_Key**

identifier within a Device and within a FW-Update file to ensure that both match

3.1.11**header**

data structure containing fields for commands and flow control in a protocol

3.1.12**host**

PC or PLC, hosting software tools as counterpart of Device applications

3.1.13**ISDU**

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

3.1.14**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.

NOTE 2 to entry: This profile has no impact on the design of Master.

3.1.15**On-request Data (OD)**

acyclically transmitted data upon request of the Master application consisting of parameters or Event data

3.1.16**P-BLOB_Trans_Layer**

communication protocol in PLC user programs for the transmission of data larger than the ISDU length

Note 1 to entry: Usually, the protocol is embedded in a function block (FB) written in the IEC 61131-3 programming language structured text (ST)

Note 2 to entry: Counterpart is located as D-BLOB_Trans_Layer within a Device

3.1.17**port**

communication medium interface of the Master to one Device

3.1.18**Process Data (PD)**

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

3.1.19**ProfileIdentifier**

list of supported profiles and function classes

3.1.20**segment**

data structure consisting of a *header* and user data

3.1.21**T-BLOB_Trans_Layer**

communication protocol in computer tools such as supervisory software for the transmission of data larger than the ISDU length

Note 1 to entry: Counterpart is located as D-BLOB_Trans_Layer within a Device

3.1.22**Technology firmware**

permanently stored individual software in a Device providing control, monitoring, and data manipulation to support a particular technology

3.1.23**Tool**

software means within controllers or personnel computers for the processing of *BLOBs* or firmware updates

3.1.24**Unlocking**

specified sequence of SystemCommands to enable a FW-Update by the *Bootloader*

Note 1 to entry: This feature prevents the Device from accidental access and firmware changes

3.1.25

upload

transfer of data from the Device to a tool via a Master

Note 1 to entry: The upload feature is not required for the firmware update

3.2 Symbols and abbreviated terms

BLOB	binary large object	
BM	boot mode	
FB	function block	IEC 61131-3
FW	firmware	
GUI	graphical user interface	
IOLFW	IO-Link firmware (file extension)	
ISDU	indexed service data unit	IEC 61131-9
LSO	Least significant octet	
MSO	most significant octet	
OD	on-request data	IEC 61131-9
PC	personal computer	
PCDT	parameterization, configuration, and diagnosis tool	IEC 61131-9
PLC	programmable logic controller	
SDCI	single-drop digital communication interface	IEC 61131-9

3.3 Conventions

3.3.1 Behavioral descriptions

For the behavioral descriptions, the notations of UML 2 [3] are used, mainly state, activity, and sequence diagrams. The layout of the associated state-transition tables is following IEC 62390 [3].

Triggers are for example external requests ("calls") or internal changes such as timeouts; [guard] are Boolean conditions for exits of states; numbered transitions describe actions in addition to the triggers within separate state-transition tables.

In this document, the concept of "nested states" with superstates and substates is used as shown in the example of Figure 1.

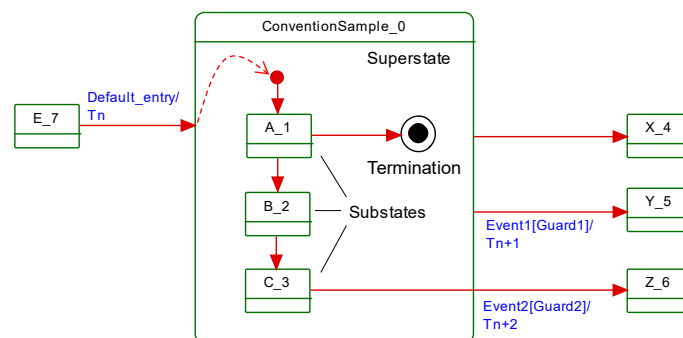


Figure 1 – Example of a nested state

UML 2 allows hierarchies of states with superstates and substates. The highest superstate represents the entire state machine. This concept allows for simplified modelling since the content of superstates can be moved to a separate drawing. An eyeglasses icon usually represents

this content. Compared to "flat" state machines, a particular set of rules shall be observed for "nested states":

- a) A transition to the edge of a superstate (e.g. Default_entry) implies transition to the initial substate (e.g. A_1).
- b) Transition to a termination state inside a superstate implies a transition without event and guard to a state outside (e.g. X_4). The superstate will become inactive.
- c) A transition from any of the substates (e.g. A_1, B_2, or C_3) to a state outside (Y_5) can take place whenever event1 occurs and guard1 is true. This is helpful in case of common errors within the substates. The superstate will become inactive.
- d) A transition from a particular substate (e.g. C_3) to a state outside (Z_6) can take place whenever event2 occurs and guard2 is true. The superstate will become inactive.

In this document, the concept of "nested states" with regions is used also as shown in the example of Figure 2.

The two "lanes" (regions) can run independently or concurrently. It is possible to exit from a substate in one of the regions, for example from substate B_2, whenever Event1 occurs and guard1 is true. The superstate will become inactive (both regions).

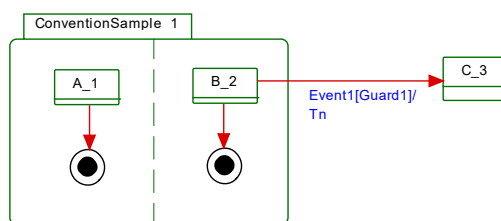


Figure 2 – Superstate with regions (And-state)

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

3.3.2 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 (Figure 3).

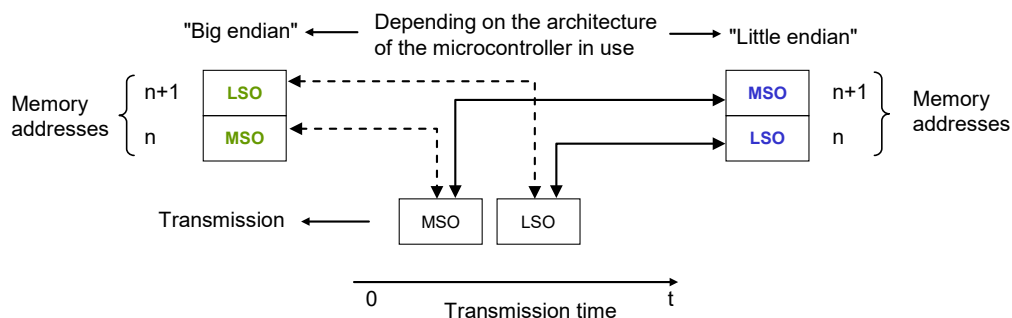


Figure 3 – Memory and transmission octet order

4 Overview of the profile

This profile consists of two parts, "BLOB" and "Firmware-Update". The first part is about the transfer of so-called *binary large objects* such as pictures taken by an optical sensor or large amount of structured data collected by a Device. Due to the limited size of ISDUs (≤ 232 octets) for the transfer of consistent records in IO-Link (see 6.4.1), it is necessary to define a segmented and controlled transfer mechanism on top of the existing ISDU mechanism.

The second part is about *firmware update* (short: FW-Update) of Devices in the field. It deals with unlocking of a Device's firmware; the necessary commands and messages; FW-Update file

formats; identification, authentication, and validation issues; DeviceID and version rules; parameter versions and data storage rules.

Figure 4 shows how the additional functions "BLOB transfer" and "Firmware-Update" conform to the basic IO-Link operations (SIO, STARTUP, PREOPERATE, and OPERATE). The profile assumes a tested IO-Link communication layer according to [1], confirmed by a manufacturer declaration.

Usually, a configured Device in the field moves to the OPERATE state after power-on (cyclic Process Data exchange). From there, it is possible via particular Parameters (BLOB_ID and BLOB_CH) to initiate BLOB transfers from and to the Device in an acyclic manner. The entire transmission of a BLOB is secured by a 32 bit CRC signature. The profile specifies state machines for the Device, for function blocks (FB according IEC 61131-3 programming languages) in PLCs or for computer software tools (Figure 6).

For FW-Update, a "Firmware valid" check is mandatory prior to a Device start-up until it reaches the OPERATE state.

NOTE Usually, Masters and Devices automatically run through the states in Figure 4 until the OPERATE state is taken. In principle, a FW-Update is also possible from the PREOPERATE state.

From there, additional SystemCommands allow for unlocking of the existing technology firmware within a Device. This new "unlocking" feature as part of the technology firmware is a precondition for the FW-Update of a Device. After unlocking, the Device causes a communication interrupt and after 4 Master cycles, a Bootloader takes over responsibility. This one supports the download of the new firmware using the BLOB transfer mechanism.

At the end of the transfer, the new firmware is activated and after another communication interrupt and after 4 Master cycles, the new firmware can take over responsibility.

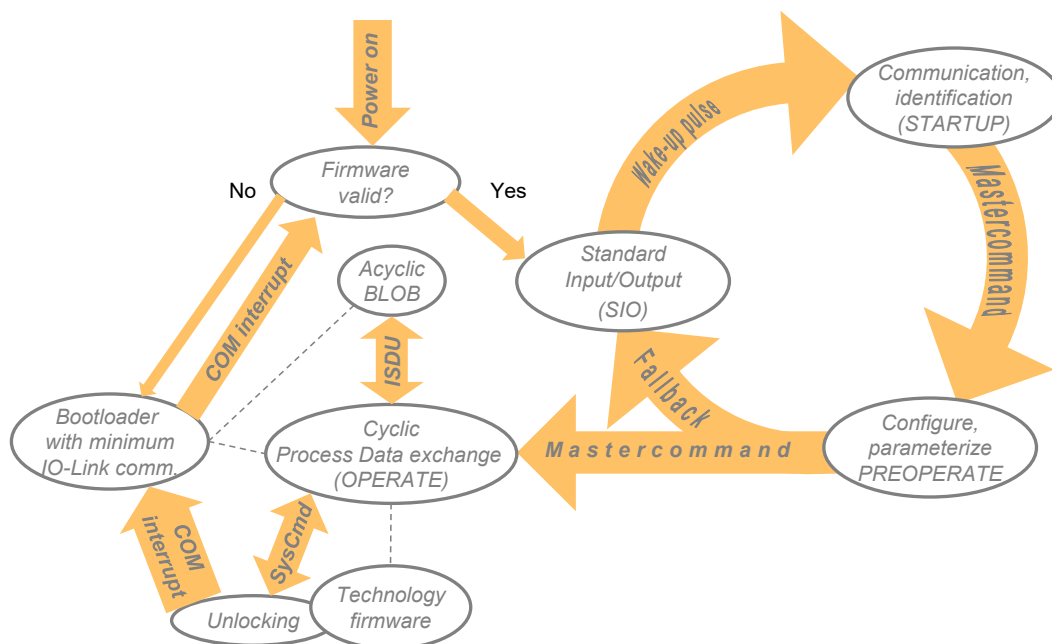


Figure 4 – IO-Link operations including BLOB transfer and FW-Update

However, in case the firmware valid test failed, the Bootloader can be re-enabled to support another update procedure.

Figure 5 shows how the additional features "BLOB transfer", "Bootloader" for FW-Update, and the "Unlocking" of the firmware fit into the Device architecture defined in clause 10 of [1].

The content of this profile is structured into the following main clauses. After this overview, clause 5 deals with Binary Large Objects (BLOBs) such as images, totalized measurements, etc. exceeding the size of one ISDU and to be transmitted in segmented manner. Clause 6 deals with firmware updates (FW-Update) of Devices using the BLOB mechanism.

Annex A describes the necessary IODD extensions for both. Annex B specifies the CRC signature calculations and Annex C optional features such as compression and encryption. Annex D provides deep insight in performance issues. Annex E lists the necessary test cases for conformance testing and Annex F defines quality measures.

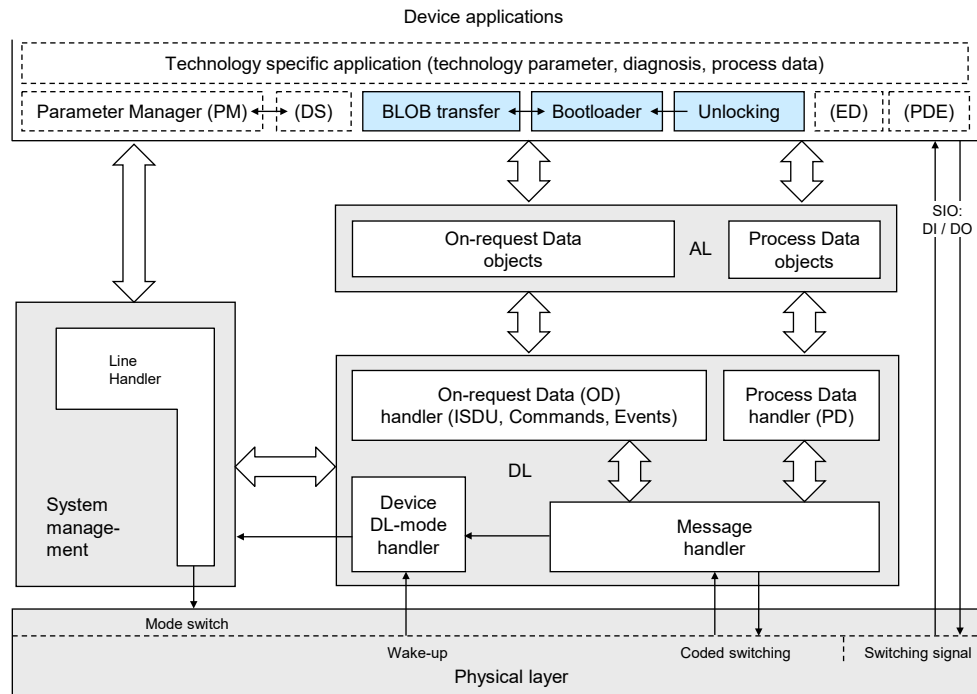


Figure 5 – Device architecture including BLOB and FW-Update

5 Profile characteristic

It is mandatory for the BLOB & FW-Update profile to observe the definitions within the IO-Link Common Profile (see [8]) except for the bootloader. This profile provides a parameter "Profile Characteristic" (see [1], Table B.8) indicating the ProfileIdentifiers (PFIDs) a particular Device supports. In this case the following PFIDs apply:

- BLOB: PFID = 0x0030
- FW-Update: PFID = 0x0031

In case of BLOB applications within non-profile Devices, the BLOB feature can be used as a FunctionClass (FunctionClassID = 36608 (0x8F00)) with its own property (see [8]).

6 Binary Large Objects (BLOBs)

6.1 Purpose and system positioning

Figure 6 illustrates the transmission of BLOBs within an automation system. "BLOB_Trans_Layers" provide the necessary protocol features between a Device and a host controller such as a PLC or a PC-based tool running for example manufacturing supervisory software.

Usually, the D-BLOB_Trans_Layer is implemented with the help of the programming language for the Device's firmware, whereas the P-BLOB_Trans_Layer is implemented in a function block (FB) with the help of an IEC 61131-3 programming language, for example "Structured Text" (ST). In the ideal case, the PLC manufacturers are supporting a standardized FB according to this document in their FB library of the engineering tool.

The T-BLOB_Trans_Layer is implemented using appropriate programming languages of that system environment, the software tool is launched and running in.

The acyclic On-request Data mechanism "ISDU" is engaged in the transmission of BLOBs (see 6.4).

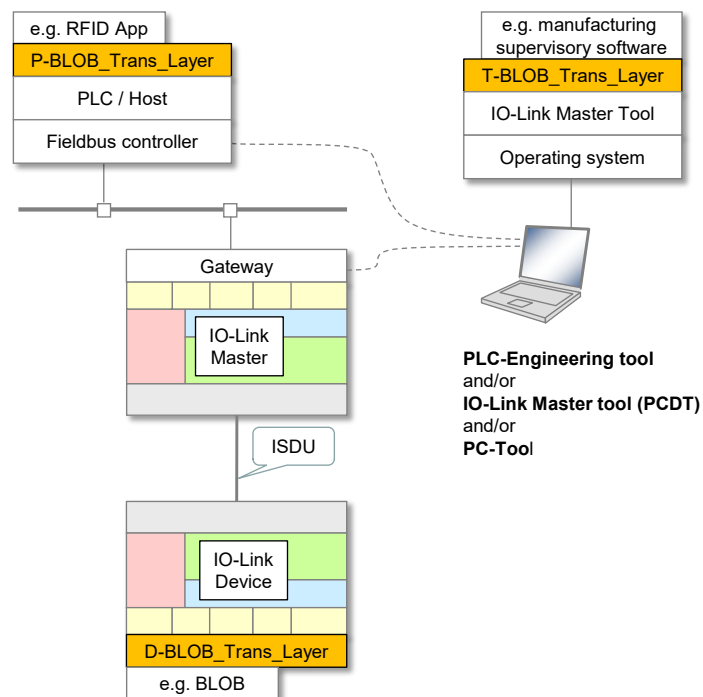


Figure 6 – BLOB transmission system

6.2 Components involved

6.2.1 PLC / Host

BLOB transfer is based on the standard IO-link protocol using ISDU communication between Master and Device. It can be implemented in a PLC user program as a P-BLOB_Trans_Layer, usually in form of a function block (FB). PLC vendors often provide common function blocks for read/write records adapted to the ISDU mechanism. This facilitates access from the PLC across the fieldbus via the Master to the Device and vice versa. Figure 6 illustrates the mechanism. The BLOB function block can be used for example to read or write RFID tags via an RFID user program within the PLC.

6.2.2 Tools

6.2.2.1 PLC-Engineering

This tool supports configuration of the IO-Link Master for fieldbus operation (network access, I/O data exchange, port configuration, etc.). It also allows development of user programs including function blocks (e.g. BLOB FB) based on IEC 61131-3 programming languages.

6.2.2.2 PC-Tools

Computer coming with communication On-request Data access to Masters (AL layer) can be used for the implementation of the BLOB transfer mechanism (T-BLOB_Trans_Layer). This enables the user to directly acquire information from a particular associated technology application within a Device or to transfer BLOBs into a Device (for example through manufacturing supervisory systems).

Such BLOBs could be FW-Update data objects. Corresponding PC-Tool software can provide user dialogs in addition to the BLOB transfer mechanism.

6.2.2.3 Master tools (PCDT)

Most of the Masters are coming with a Master tool (PCDT) that provides already On-Request Data access. Thus, it is easy to extend this tool by a T-BLOB_Trans_Layer and an additional BLOB application such as FW-Update.

328 **6.2.3 Master**

329 All Masters according IO-link V1.1 (or a later version) can handle the BLOB transfer mechanism.
330 No modification is required.

331 **6.2.4 Device**

332 The BLOB transfer mechanism supports all Devices requiring a data channel for data objects
333 larger than the ISDU size, e.g. cameras, 3D-scanner, and measurement sensors (totalizer, data
334 recorder). The data objects can consist of parameters, process data, diagnosis information,
335 firmware updates, etc.

336 **6.3 Data objects within BLOBs**

337 **6.3.1 Types of objects**

338 BLOBs can contain various data objects. Examples are images (JPG, PNG, or other formats),
339 text (ASCII, XML, CSV, or other formats), binaries (HEX, BIN, or other formats), etc.

340 This document specifies only the data object "FW-Update". It is a binary data object with a
341 manufacturer-specific content (see 7.2.4).

342 **6.3.2 Securing measures**

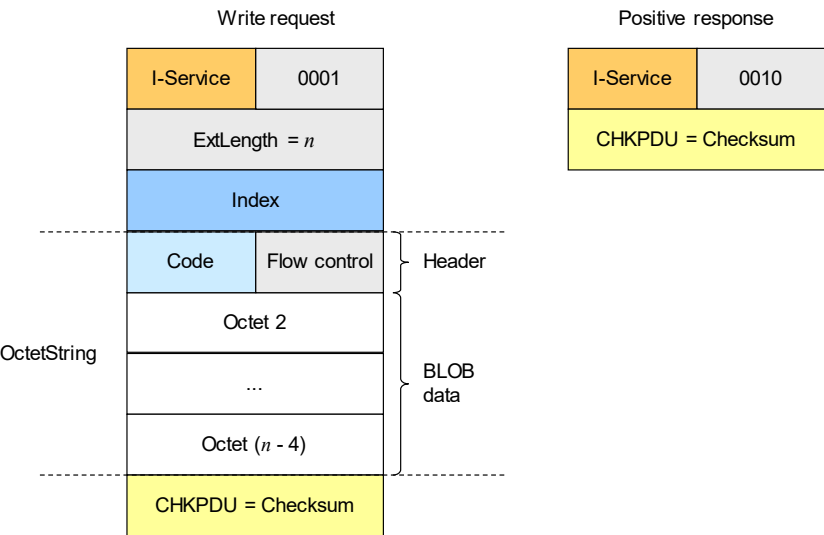
343 A 32 bit CRC signature is used to ensure data integrity of the transmitted BLOB content (see
344 Annex B).

345 Additional proprietary measures for compression, authentication, and encryption can be incor-
346 porated in the content also (see Annex C).

347 **6.4 ISDU as transport vehicle**

348 **6.4.1 General**

349 BLOBs are transferred by means of the ISDU mechanism specified in [1]. No modifications are
350 required. The size of a BLOB can be up to 231 octets to fit into one ISDU. Larger BLOBs shall
351 be segmented and transferred ISDU by ISDU. Figure 7 shows example ISDUs usually used for
352 BLOB transfer (see [1], Annex A.5.7).



353
354 **Figure 7 – Example structure of an ISDU for BLOB transfer (BLOB_CH)**

355 Annex D provides information on the BLOB transfer performance with respect to transmission
356 rates and BLOB segment sizes.

357 Since BLOB transfers can last for some time due to their extended length, SDCI communication
358 can always be interrupted after a valid ISDU transmission of a write request. Any Tool shall be
359 able to manage those communication interrupts.

6.4.2 Diagnosis – EventCodes

There are no profile specific EventCodes required.

6.4.3 Acknowledgments

The CRC signature serves as acknowledgment for the transfer of the BLOB data. The ISDU mechanism provides already sufficient acknowledgments for the transfer of segments (see [1], A.5).

6.5 BLOB parameters and transfer

6.5.1 Profile related Index space

The parameters required for the BLOB transmission are specified in Table 1 using Indices reserved for profiles (see Table B.8 in [1]). Support of these parameters is mandatory for Devices providing BLOB transfer ("conditional").

Table 1 – Index assignment of the BLOB parameters

Index (dec)	Object name	Access	Length	Data type	M/O/C	Remark
...						
0x0031 (49)	BLOB_ID	R	2 octets	IntegerT	C	See 6.5.2 and Table B.8 in [1]
0x0032 (50)	BLOB_CH	R/W	variable	OctetString	C	See 6.5.3 and Table B.8 in [1]
...						
Key M = mandatory; O = optional; C = conditional						

Details of parameter "BLOB_ID" are specified in 6.5.2 and details of parameter "BLOB_CH" in 6.5.3. ISDU type of the "BLOB_CH" shall not be a standard OctetStringT that is fixed regarding its length. The "BLOB_CH" type shall be an octet array referred to as OctetString with a dynamic range depending on the transmitted content.

BLOB parameters are transferred using ISDU Read or Write requests.

6.5.2 BLOB_ID

The parameter BLOB_ID is used to indicate the current BLOB whose transmission is in progress. Its data type is IntegerT (16) and read only. The Device provides its available BLOB_IDs via its IODD (see A.2) and the associated parameter value within this Index.

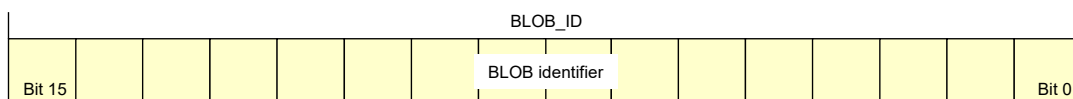


Figure 8 – Structure of BLOB_ID

Table 2 shows the coding of BLOB_ID, which implicitly indicates the read or write transmission direction.

Table 2 – Coding of BLOB_ID

Value (dec)	Definition
-32768	Not permitted
-32767 to -8192	Reserved (read)
-8191 to -4096	Manufacturer specific (read)
-4095 to -1	Profile specific (read)
0	Idle transmission of a BLOB

Value (dec)	Definition
1 to 4095	Profile specific (write)
4096 to 8191	Manufacturer specific (write)
8192 to 32767	Reserved

6.5.3 BLOB_CH

6.5.3.1 Header and body

The parameter BLOB_CH defines the transmission channel for a particular BLOB through a particular BLOB_ID within the command BLOB_Start. It has a variable structure and always starts with an 8-bit header and a variable length of BLOB (body) data.

Figure 9 shows the structure of the header of BLOB_CH.

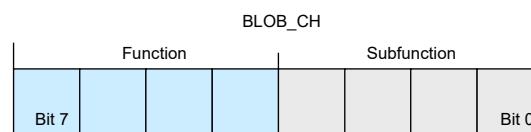


Figure 9 – Header of BLOB_CH

Table 3 shows the coding of the header of BLOB_CH. A Read request of this parameter has no associated data to the header. Therefore, the response of the Device to a Read request depends on the respective state of the BLOB state machine. A Read request also triggers internal state changes and thus cannot be performed twice.

Table 3 – Coding of the header of BLOB_CH

Transmission items	Read/Write	Function	Subfunction	Definition/Parameter
–	–	0x0	0x0 to 0xF	Reserved
BLOB_Info_Read	R	0x1 (Read Info block)	0x0	Parameter: See Figure 10 (Information for the Read channel)
BLOB_Info_Write	R		0x1	Parameter: See Figure 11 (Information for the Write channel)
BLOB_Segment	R/W	0x2	0x0 to 0xF (flow control)	Counting of segments modulo 16. It starts at 0 and rolls over after 15 to 0. Parameter: Segment. NOTE
BLOB_Last	R/W	0x3	0x0	Parameter: Last segment of the BLOB.
BLOB_CRC	R/W	0x4	0x0	Parameter: CRC signature across the BLOB.
–	–	0x5 to 0xE	0x0 to 0xF	Reserved
BLOB_Abort	W	0xF (commands)	0x0	Command to abort the active transmission channel. No parameter.
BLOB_Start	W		0x1	Command to select the BLOB_ID and to establish the transmission channel. Parameter: BLOB_ID
BLOB_Finish	W		0x2	Command to finish the active transmission channel. No parameter.
-		0xF	0x3 to 0xF	Reserved for commands
NOTE Mechanism similar to [1], Table 50 – FlowCTRL definitions, COUNT				

6.5.3.2 BLOB_Info_Read

Prior to reading the parameter "BLOB_Info_Read" block, the required BLOB shall be assigned via "BLOB_Start" (see 6.5.3.9). Without an assigned BLOB, a negative Read response will be returned via an ISDU error: 0x8020 – *Service temporarily unavailable* (see [1], Table C.1).

A Read request to the BLOB_CH Index will return the BLOB_Info_Read, which contains the header plus BLOB length in octets, coded in UIntegerT(32).

The "BLOB_Info_Read" block contains the BLOB length in octets, coded in UIntegerT (32).

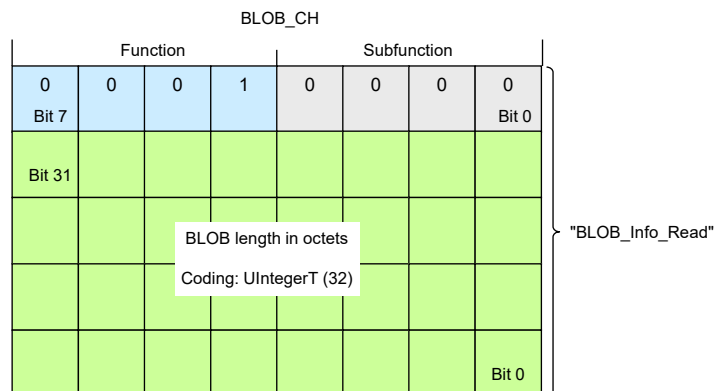


Figure 10 – Structure of "BLOB_Info_Read" block

6.5.3.3 BLOB_Info_Write

Prior to reading the parameter "BLOB_Info_Write" block, the required BLOB (BLOB_ID) shall be assigned via "BLOB_Start" (see 6.5.3.9). Without an assigned BLOB, a negative read response will be returned via an ISDU error 0x8020 – *Service temporarily unavailable*.

A Read request to the BLOB_CH Index with the corresponding header (Function and Subfunction, see Table 3) will return the "write" properties of the active BLOB_ID within an information block as shown in Figure 11.

The "BLOB_Info_Write" block contains the maximum BLOB size in octets, coded in UIntegerT (32) and the maximum ISDU size of a particular Device in octets, coded in UIntegerT (8).

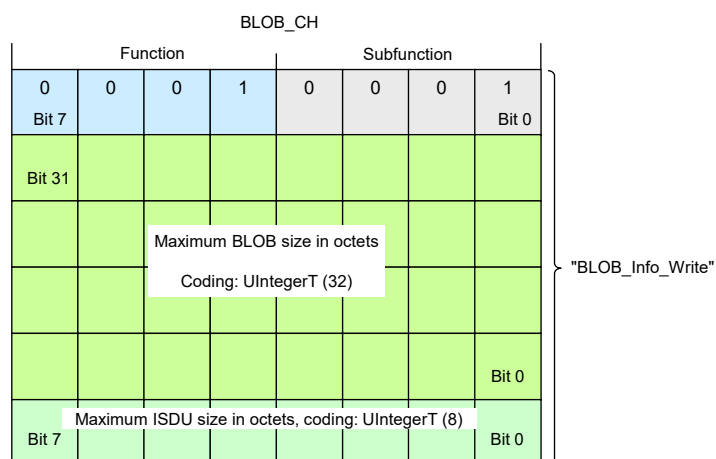


Figure 11 – Structure of "BLOB_Info_Write" block

6.5.3.4 Write BLOB_Segment

Whenever a BLOB is larger than the size of an ISDU, it will be transferred in segments, in each case filling the entire ISDU data size. The item "BLOB_Segment" indicates a particular part of the BLOB via a flow control number ("BFlowCtrl") of the segment.

A Write request to the BLOB_CH Index with the corresponding header (Function and Subfunction, see Table 3) and its body is shown in Figure 12. The data shall be transmitted as-is.

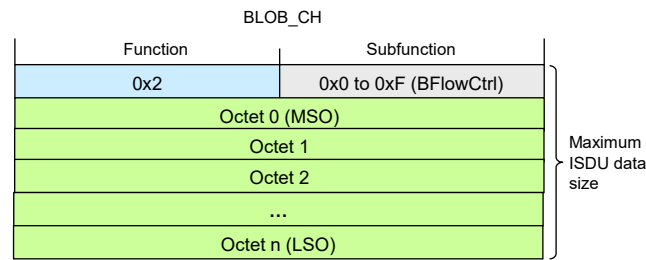


Figure 12 – Structure of "BLOB_Segment"

6.5.3.5 Read BLOB_Segment

A Read request to the BLOB_CH Index with the corresponding header (Function and Subfunction, see Table 3) will return a body as shown in Figure 12. Data shall be transmitted as a big endian sequence, i.e. the most significant octet (MSO) shall be transmitted first, followed by less significant octets in descending order, with the least significant octet (LSB) being sent last according to [1].

6.5.3.6 BLOB_Last

The item "BLOB_Last" indicates the last segment to be transferred. This segment contains the remainder of the BLOB. Padding octets shall not be used. The last ISDU shall have the length of the remaining octets of the BLOB including the BLOB_CH header.

Write and Read requests correspond to 6.5.3.4 and 6.5.3.5.

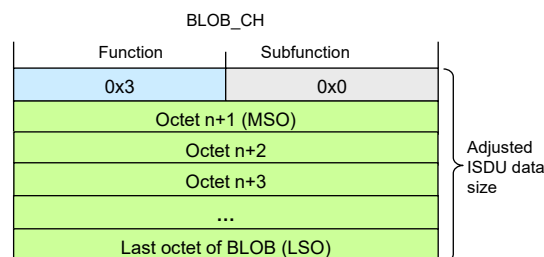


Figure 13 – Structure of "BLOB_Last"

6.5.3.7 BLOB_CRC

The item "BLOB_CRC" transmits the 32 bit CRC signature used to secure the entire BLOB data.

A Write request to the BLOB_CH Index with the corresponding header (Function and Subfunction, see Table 3) and its body is shown in Figure 14 .

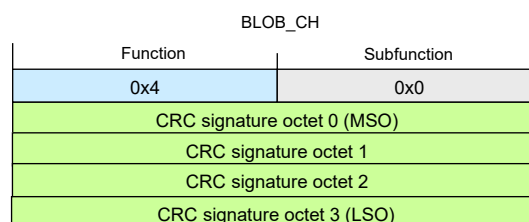


Figure 14 – Structure of "BLOB_CRC"

Writing a CRC signature will lead to a positive acknowledgment in case of correct transmission or to an ISDU error 0x8040 – *Invalid parameter set* (see [1], Table C.1).

Annex B provides information on how to calculate the BLOB_CRC signature.

A Read request to the BLOB_CH Index with the corresponding header (Function and Subfunction, see Table 3) will return a body as shown in Figure 14.

6.5.3.8 BLOB_Abort

The item "BLOB_Abort" represents a command to abort an ongoing BLOB transmission. The Write request is shown in Figure 15.

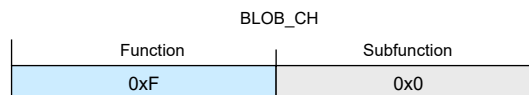


Figure 15 – Structure of command "BLOB_Abort"

The Write request will lead to a positive acknowledgment. The Write request will lead to a negative acknowledgment, if no BLOB transfer is ongoing (see Idle_0 and T3 in Figure 18).

6.5.3.9 BLOB_Start

The item "BLOB_Start" represents a command to launch the transmission of that BLOB whose BLOB_ID is contained in the parameter. The Write request is shown in Figure 16.

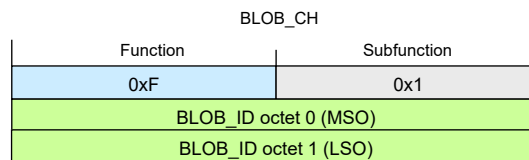


Figure 16 – Structure of command "BLOB_Start"

The Write request will lead to

- a positive acknowledgment in case of an accepted command, or
- an ISDU error 0x8030 – *Parameter value out of range* in case the BLOB_ID is not supported, or
- an ISDU error 0x8022 – *Service temporarily not available – Device control* in case the transfer for this BLOB_CH is already active.

6.5.3.10 BLOB_Finish

The item "BLOB_Finish" represents a command to finalize a successful BLOB transmission. The Write request is shown in Figure 17.

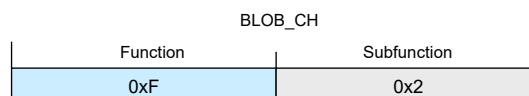


Figure 17 – Structure of command "BLOB_Finish"

The Write request will lead to a positive acknowledgment in case of an accepted command or to an ISDU error 0x8030 – *Parameter value out of range* in case the state machine is not in state "SupplyCRC_3" or "WaitOn_BLOB_complete_6" (see Figure 18).

6.5.3.11 Concurrent ISDU transfers

If any concurrent ISDU transfer besides the BLOB transfer leads to a conflict, ISDU error 0x8020 – *Service temporarily unavailable* shall be returned.

6.6 Protocol of BLOB transmission

6.6.1 Device BLOB state machine

Figure 18 shows the state machine of the "D-BLOB_Trans_Layer" (see 6.1). It is driven by Read or Write requests from the Host state machine (see Figure 19). However, within each state a Read_BLOB_ID will always be responded without quitting the state.

According to the conventions in 3.3.1, the nested states 1 to 6 in superstate 7 are enabled to react on errors (T18) within all of these states or on an incoming service "BLOB_Abort" (T17).

Any incorrect Read or Write request within superstate 7 will lead to the state IDLE_0 in order to synchronize with the Host state machine in Figure 19.

The Device shall prevent communication from being disrupted during the FW-Update operation by setting robust parameters such as a sufficient Min_Cycle_Time.

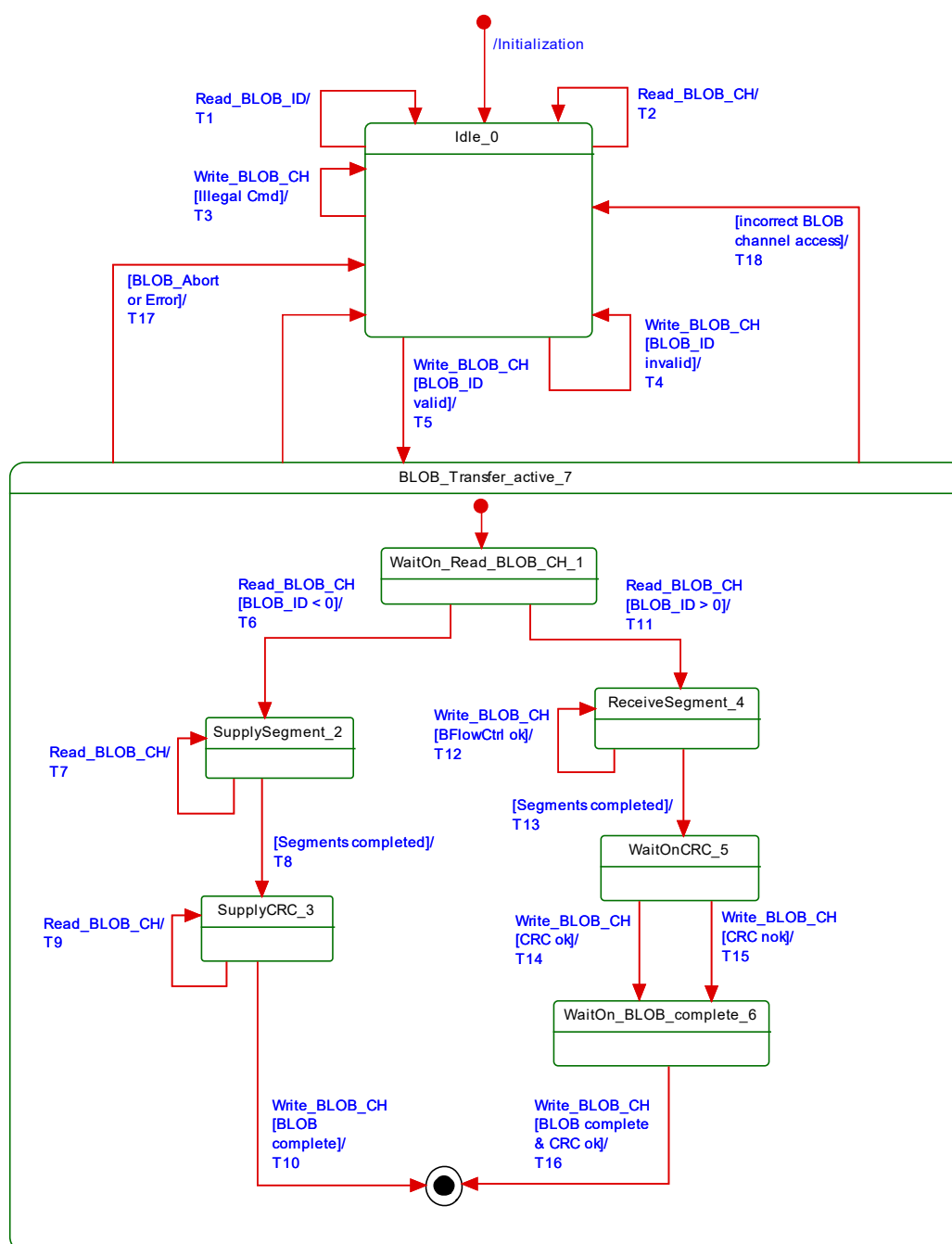


Figure 18 – Device BLOB state machine

Table 4 shows the state transition tables of the Device BLOB state machine.

Table 4 – State transition tables of the Device BLOB state machine

STATE NAME		STATE DESCRIPTION	
Idle_0		No BLOB transmission active.	
WaitOn_Read_BLOB_CH_1		Prepare BLOB info for requested BLOB_ID and direction.	
SupplySegment_2		Prepare segments in ascending order upon every read request, generate flow control	
SupplyCRC_3		Prepare CRC signature across transmitted BLOB content. Wait on BLOB finalization via BLOB_Finish command.	
ReceiveSegment_4		Receive segments in ascending order upon every write request, check flow control.	
WaitOnCRC_5		Receive target CRC signature across transmitted BLOB content. Compare with internally calculated CRC.	
WaitOn_BLOB_complete_6		Wait on BLOB finalization via BLOB_Finish command.	
BLOB_transfer_active_7		This superstate allows all states inside to react on - command "BLOB_Abort", - reading BLOB_ID to provide actual BLOB_ID, - command BLOB_Start to provide ISDU error 0x8022 – <i>Service temporarily unavailable – Device control</i> .	
TRANSITION	SOURCE STATE	TARGET STATE	ACTION
T1	0	0	Response: BLOB_ID = 0.
T2	0	0	Response: ISDU error 0x8020 – <i>Service temporarily unavailable</i>
T3	0	0	Response: ISDU error 0x8030 – <i>Parameter value out of range</i>
T4	0	0	Response: ISDU error 0x8030 – <i>Parameter value out of range</i>
T5	0	1	Response: ISDU acknowledgment
T6	1	2	Response BLOB_CH: "BLOB_Info_Read"
T7	2	2	Response BLOB_CH: "BLOB_Segment" content
T8	2	3	–
T9	3	3	Response BLOB_CH: "BLOB_CRC"
T10	3	0	Set parameter BLOB-ID to IDLE
T11	1	4	Response BLOB_CH: "BLOB_Info_Write"
T12	4	4	Store BLOB segment, response: ISDU acknowledgment
T13	4	5	Calculate CRC signature across received BLOB content
T14	5	6	Response ISDU acknowledge "No Error"
T15	5	6	Response ISDU error 0x8040 – <i>Invalid parameter set</i>
T16	6	0	Set parameter BLOB-ID to IDLE
T17	1,2,3,4,5,6	0	Set parameter BLOB-ID to IDLE; garbage collection
T18	1,2,3,4,5,6	0	Set parameter BLOB-ID to IDLE; garbage collection. Return ISDU error 0x8030 – <i>Parameter value out of range</i> .
INTERNAL ITEMS		TYPE	DEFINITION
BLOB_ID		Index	See 6.5.2
BLOB_CH		Index	See 6.5.3
BLOB_Abort		Service	Item of Index BLOB_CH: See 6.5.3.8
BLOB_Finish		Service	Item of Index BLOB_CH: See 6.5.3.10
CRC		Variable	CRC signature across BLOB data
BLOB_Info_Read		Variable	Item of Index BLOB_CH: "Information", see 6.5.3.2

INTERNAL ITEMS	TYPE	DEFINITION
BLOB_Info_Write	Variable	Item of Index BLOB_CH: "Information", see 6.5.3.3
Illegal Cmd	Guard	Any access to BLOB-CH with an invalid header or data content Only Write function command BLOB_Start is valid
Incorrect BLOB channel access	Guard	Any invalid read or write access on BLOB-CH when this direction or content is not allowed
Error	Guard	Any negative ISDU response which is not handled within the super-state
BFlowCtrl	Variable	Flow control of BLOB transmission according to Table 3

6.6.2 Host BLOB state machine

Figure 19 shows the state diagram of the "P-BLOB_Trans_Layer" or "T-BLOB_Trans_Layer" (see 6.1). Substates 5 to 9 are nested in superstate 3 that allows reacting on errors (T15) within all of these substates. Substates 10 to 14 are nested in superstate 4 that allows reacting on errors (T25) within all of these substates (see conventions in 3.3.1).

NOTE Req_W_BLOB_ID is a "CallTrigger" (in UML) and means: Host requests the Master to send an ISDU with "Write_BLOB_ID"

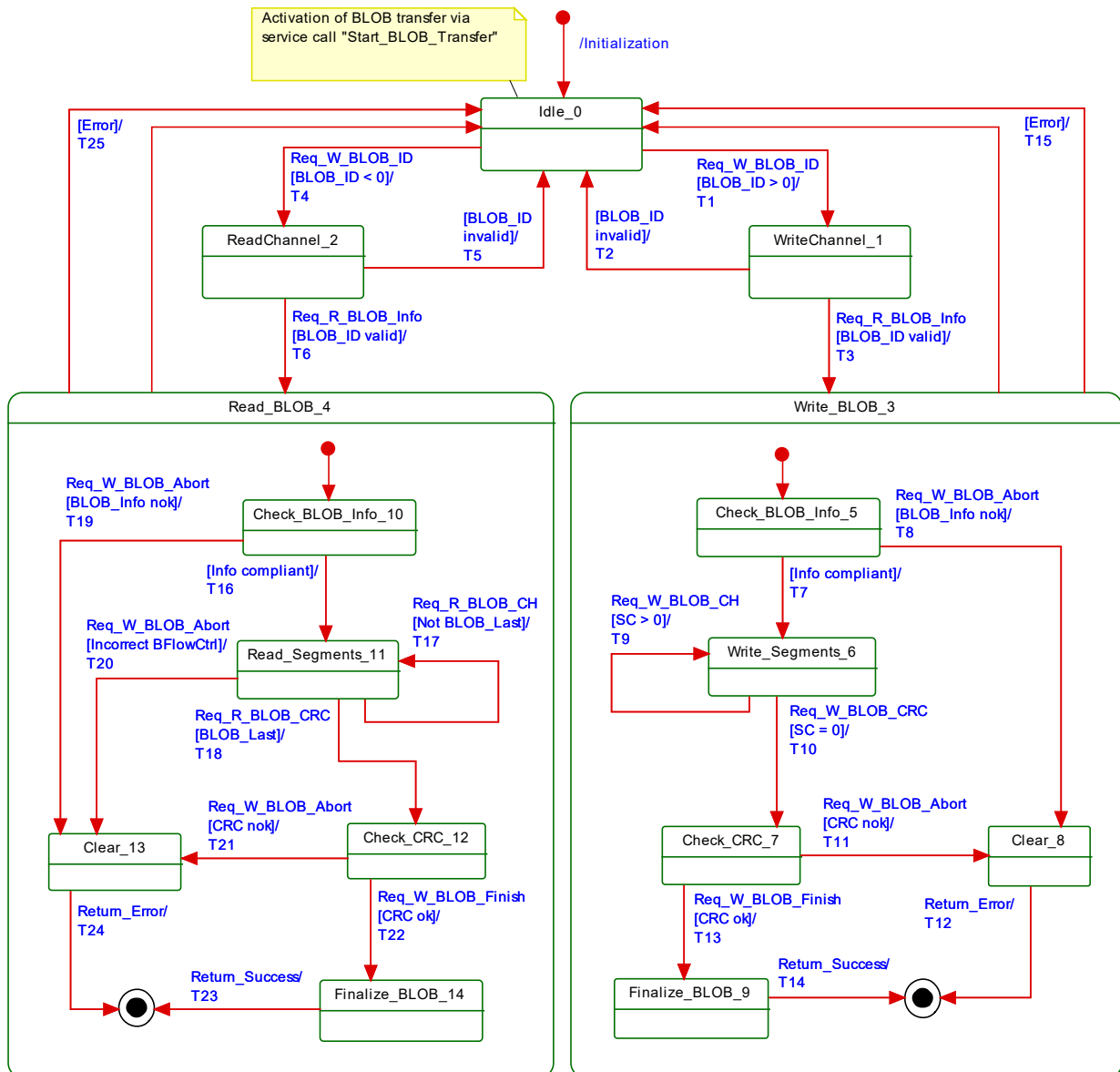


Figure 19 – Host BLOB state machine

512 Table 5 shows the state transition tables of the Host BLOB state machine.

513 **Table 5 – State transition tables of the Host BLOB state machine**

STATE NAME		STATE DESCRIPTION	
Idle_0		No BLOB transmission active. Wait on activation through service call "Start_BLOB_Transfer" with arguments BLOB_ID and a pointer to the binaries of the BLOB.	
WriteChannel_1		Await Device ISDU response: "BLOB_ID" OK?	
ReadChannel_2		Await Device ISDU response: "BLOB_ID" OK?	
Write_BLOB_3		Complete BLOB transfer to the Device. This superstate allows all states inside to react on any error and to quit.	
Read_BLOB_4		Complete BLOB transfer from the Device. This superstate allows all states inside to react on any error and to quit.	
Check_BLOB_Info_5		Wait on Device response: "BLOB_Info" OK? Calculate number of segments depending on BLOB size, max ISDU size and set SegmentCounter value.	
Write_Segments_6		Demand Master to write BLOB segment by segment (ISDU by ISDU); generate/update BFlowCtrl. Wait on Device response: ISDU OK? Decrement SegmentCounter at each successful segment transfer. In case of ComLost the last ISDU request shall be retried at least 2 times.	
Check_CRC_7		Wait on Device response: CRC OK? In case of ComLost the last ISDU request shall be retried at least 2 times.	
Clear_8		Garbage collection. Prepare error message.	
Finalize_BLOB_9		Wait on Device response.	
Check_BLOB_Info_10		Wait on Device response: "BLOB_Info" OK?	
Read_Segments_11		Demand Master to read BLOB segment by segment (ISDU by ISDU); check BFlowCtrl. Wait on Device response.	
Check_CRC_12		Compare internally calculated CRC with received CRC	
Clear_13		Garbage collection. Prepare error message.	
Finalize_BLOB_14		Wait on Device response.	
TRANSITION	SOURCE STATE	TARGET STATE	ACTION
Initialization	-	0	–
T1	0	1	–
T2	1	0	–
T3	1	4	–
T4	0	2	–
T5	2	0	–
T6	2	3	–
T7	5	6	–
T8	5	8	–
T9	6	6	–
T10	6	7	–
T11	7	8	–
T12	8	0	–
T13	7	9	–
T14	9	0	–
T15	5,6,7,8,9	0	Garbage collection
T16	10	11	–
T17	11	11	–

514

TRANSITION	SOURCE STATE	TARGET STATE	ACTION
T18	11	12	–
T19	10	13	–
T20	11	13	–
T21	12	13	–
T22	12	14	–
T23	14	0	–
T24	13	0	–
T25	10,11,12, 13,14	0	Garbage collection
INTERNAL ITEMS		TYPE	DEFINITION
BLOB_ID		Index	See 6.5.2
BLOB_CH		Index	See 6.5.3
BLOB_Abort		Service	Item of Index BLOB_CH: See 6.5.3.8
SegmentCounter (SC)		Variable	Preset with number of required segments and decrement to "0".
BLOB_CRC		Variable	Item of Index BLOB_CH: CRC signature across BLOB data
BLOB_Info_Read		Variable	See 6.5.3.2
BLOB_Info_Write		Variable	See 6.5.3.3
BLOB_Last		Variable	Last BLOB segment transmitted
Incorrect flow		Boolean	BFlowCtrl violated
ComLost		Variable	Communication failed
RetryCounter		Variable	Preset = 2
Retry		Guard	Retry = true, when ComLost and RetryCounter <>0

6.7 Access conflicts

6.7.1 Overview

The host application shall be designed in such a manner that access conflicts are avoided. The Device cannot distinguish between two different BLOB transfers.

6.7.2 Concurrent access of tools

Any Host shall delay/reject a transmission, when

- "Read BLOB_ID" does not occur in state "Idle_0",
- a "BLOB_Start" results in Error 0x8022 – *Service temporarily not available – Device control* or 0x8082 – *Application not ready*.

6.7.3 Access blocking

An ongoing BLOB transfer shall only be interrupted by the PLC or Host upon deliberate user intervention. This can occur whenever a previously interrupted BLOB transfer is restarted, or a second Host starts an independent BLOB access.

The user's decision depends heavily on the BLOB currently under transmission and the circumstances.

7 Firmware-Updates

7.1 Purpose and system positioning

Figure 20 illustrates the transmission of firmware updates within an automation system with IO-Link. The "FW-Update" application and the "Bootloader" using "BLOB_Trans_Layers" provide the necessary protocol features between a Device and a computer software tool.

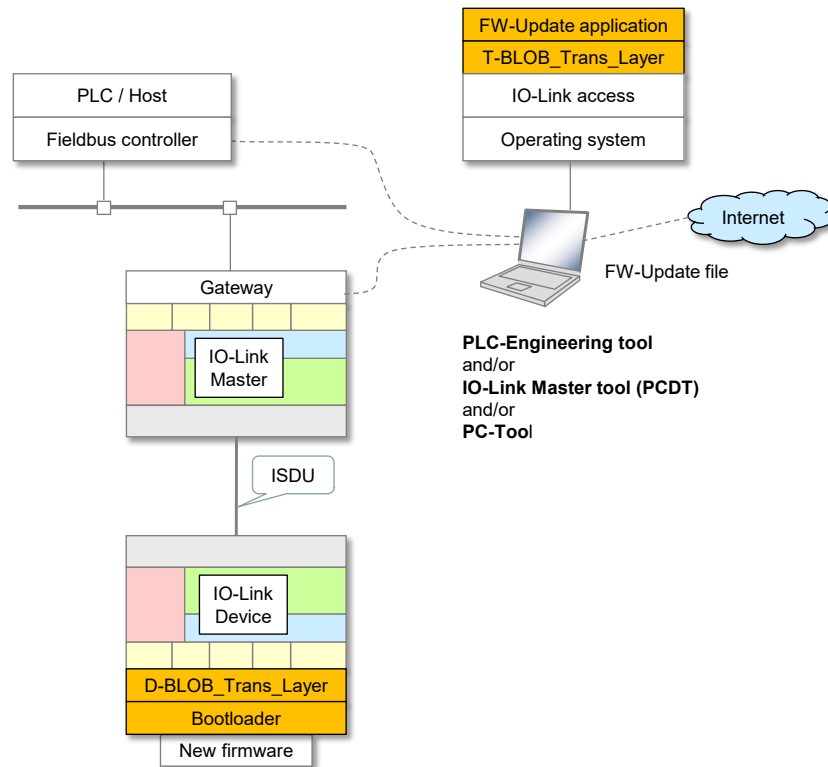


Figure 20 – FW-Update transmission system

7.2 Components involved

7.2.1 Master (PCDT) or PC tools

Either Master (PCDT) or individual PC tools carrying the BLOB transfer mechanism (see 6.2.2.2 or 6.2.2.3) can provide a FW-Update application software with particular user dialogs such as Device access and identification, FW-Update file acquisition, verification of compatibility, authorization, download, and finalization.

7.2.2 Master

All Masters according IO-link V1.1 or later can handle the FW-Update/BLOB transfer mechanism. No modification is required.

7.2.3 Device

As a precondition, the Device shall support the FW-Update profile. The technology firmware of such a modified Device is extended by a firmware check state and a bootloader. The bootloader is responsible for the BLOB transfer, the verification, the optional decryption and the flashing of the new technology firmware. In case of a FW-Update error (invalid technology firmware) the active bootloader remains enabled and a retry is possible.

7.2.4 FW-Update file

The Device manufacturer is responsible for the provision of the FW-Update file. It contains metadata (e. g. for identification and verification) and the binary data of a valid firmware. The binary data is manufacturer specific and can be encrypted or packed.

7.3 Use cases

7.3.1 Update Firmware

Figure 21 shows the use cases for the FW-Update procedure.

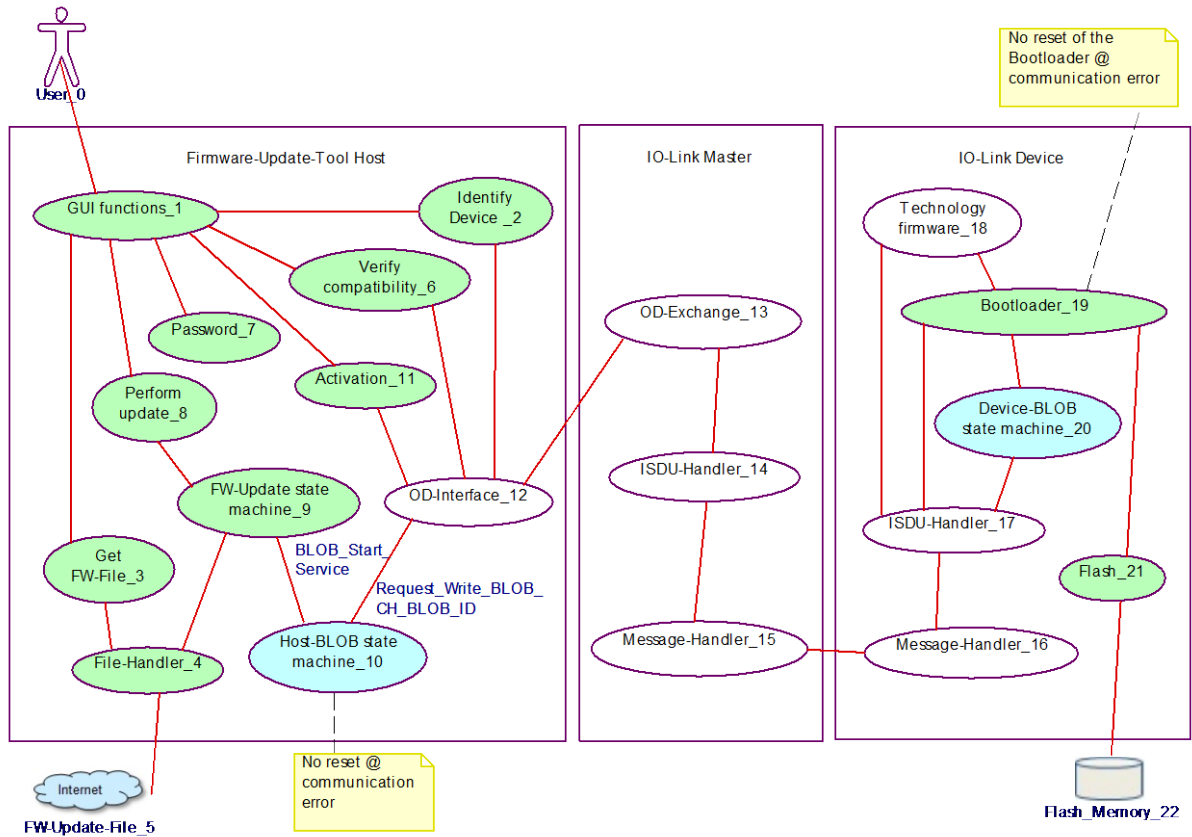


Figure 21 – Use cases of the FW-Update procedure

Table 6 shows a listing of the items in Figure 21 and references to clauses within this document or to other IO-Link specifications (bibliography).

Table 6 – Use Case reference table

No.	Item	Type	Reference	Remarks
0	User	Role description	–	Responsibility of the software tool manufacturer
1	GUI functions	Services	Clause 7.9.1	
2	Identify Device	Activity	Clause 7.5.2	
3	Get FW-File	Activity	Clause 7.5.3	
4	File-Handler	Activity	Clause 7.5.3	
5	FW-Update-File	Meta data	Clause 7.4.4	
6	Verify compatibility	Activity	Clause 7.5.4	
7	Password	Activity	Clause 7.5.5	
8	Perform update	Activity	Clause 7.5.6	
9	FW-Update state machine	State machine	7.7.2 and 7.7.4	
10	Host-BLOB state machine	State machine	Clause 6.6.2	
11	Activation	Activity	Clause 7.5.7	
12	OD-Interface	Comm-Layer	[1]	Proprietary

No.	Item	Type	Reference	Remarks
13	OD-Exchange	Gateway application	[1]	IO-Link standard
14	ISDU-Handler	Master DL	[1]	IO-Link standard
15	Message-Handler	Master DL	[1]	IO-Link standard
16	Message-Handler	Device DL	[1]	IO-Link standard
17	ISDU-Handler	Device DL	[1]	IO-Link standard
18	Technology firmware	Device application	–	Proprietary
19	Bootloader	Activity	Clause 7.5	
20	Device-BLOB state machine	State machine	Clause 6.6.1	
21	Flash	Activity	Clause 7.7.2	
14	Flash memory	Requirements	–	Proprietary

7.3.2 Upgrade

In this case, the firmware within a Device will be replaced by a newer version providing enhanced functionality or bug fixes. The vendor of the FW-Update file shall ensure that the new firmware is compatible and complies with all required standards, for example [1]. The upgrade can comprise the entire firmware or only parts of it.

Table 7 lists use cases of possible changes or upgrades and the corresponding implications to be considered by the designer/manufacturer.

Table 7 – Use cases of possible modifications

Modification	Update in the field/at customer site	Impact
Technology FW (bug fix)	Yes	–
Technology FW (functions)	Yes	- New DeviceID and IODD - Reset DS
IO-Link stack (bug fix)	Yes	–
Bootloader/FW-Update	No	–
Additional IO-Link parameter	Yes	- New DeviceID and IODD - Reset DS
Parameter structure change	Not recommended	- New DeviceID and IODD - Reset DS
Additional profile/ new profile version	Not recommended	- New DeviceID and IODD - Reset DS
DeviceID	Yes	- New IODD - Reset IL (Inspection Level)
VendorID	No	–
SerialNumber	No	–
HardwareRevision	No	–
ProductID, ProductName	Not recommended	- New IODD
Password option	No	–

7.3.3 Downgrade

In one case, the new firmware within the Device may lead to an incompatibility with external automation components such as a Master or function blocks (FB) within a PLC. In another case, a user requires a previous proven-in-use or qualified firmware version within his machine application even in spare part Devices with a newer firmware.

In both cases, a rollback to the previous version is possible and the manufacturer/vendor can provide the previous version as a second FW-Update file. The downgrade shall restore all

firmware parts modified by the upgrade. User parameters are available through the Data Storage mechanism.

Some microcontrollers allow flashing of a second firmware version and simple switching between both. It is the responsibility of the manufacturer to provide the switching mechanism.

7.3.4 Upload firmware

Upload of a flashed firmware is not supported.

7.4 File formats

7.4.1 General

The FW-Update file contains information specific to the FW-Update mechanism of Devices via IO-Link communication. The file internally represents a zip archive. The archive is a package that shall consist of a metadata file, binary data file with the firmware BLOB and optionally additional resource files. The metadata file describes the FW-Update data and the internal file structure of the package. All these files shall be included in a zip archive without embedded folders. The archive extension shall be ".iolfw".

7.4.2 Creation of file

The FW-Update file can be created by the use of third-party tools. Furthermore, it is possible to generate the FW-Update file by manually adding the metadata file, the binary data file (BLOB), and optionally additional resource files to a zip archive and renaming it according to the file naming convention (see 7.4.3). The structure and contents of the meta data file is described in 7.4.4.

7.4.3 File naming convention

The file name shall be chosen according to the naming convention shown in Figure 22:

`<Vendor name>-<Firmware descriptor>-<Date of creation>-IOLFW<Schema version>.iolfw`

Figure 22 – FW-Update file naming convention

The terms used in Figure 22 are specified in Table 8.

Table 8 – Items of the file naming convention

Item	Format	Definition
Vendor name	UTF8	Name of FW-Update file vendor, usually the Device vendor
Firmware descriptor	UTF8	Vendor specific descriptor of the file
Date of creation	YYYYMMDD	This date shall correspond to the "releaseDate" attribute in the DocumentInfo element of the metadata file
Schema version	[0-9].[0-9]	This version of the schema for the firmware description file shall comply with the XML standard, for example "1.0".

Figure 23 shows an example of an FW-Updatefile name.

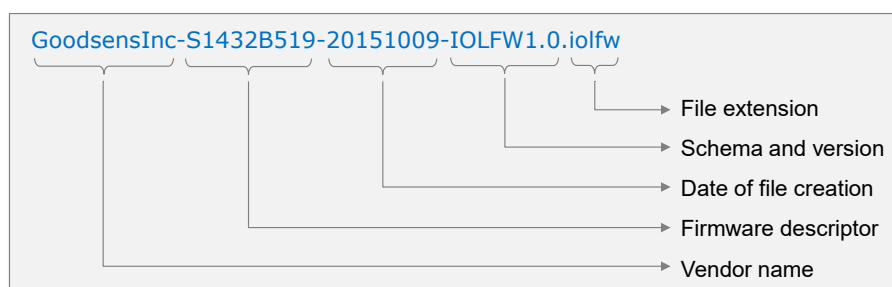


Figure 23 – Example of a FW-Update file name

7.4.4 Meta data file description

7.4.4.1 General

The metadata file format is based on extended markup language (xml). The file name shall be the same as for the entire archive except the extension that shall be ".xml". All XML files shall use "UTF-8" coding. The structure of the file is formally defined in the xml schema file (IOLFW1.0.xsd). The schema can be used to validate, edit and process the file in firmware update/viewer tools.

7.4.4.2 XML structure

Figure 24 shows the Meta data structure of FW-Update files.

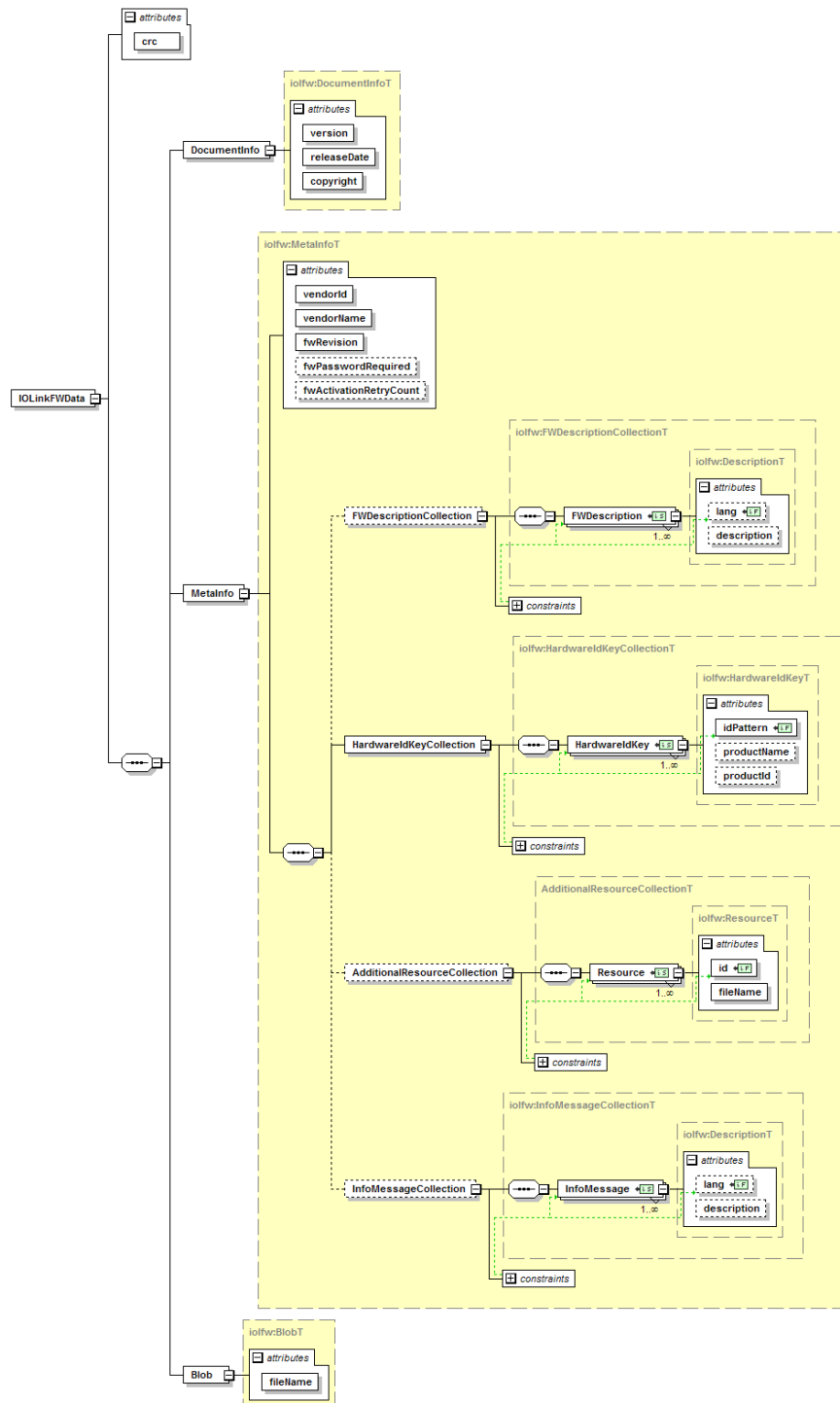


Figure 24 – Meta data XML structure

The elements in Figure 24 are defined in Table 9, Table 10, and Table 11. In firmware description XML files, the prefix of namespace <http://www.io-link.com/iolfw/2016> shall be "iolfw". Each attribute needs the namespace prefix "iolfw".

7.4.4.3 Root elements

The root element of the FW-Update file is IOLinkFWData. It has one mandatory CRC attribute, which is a signature across the xml metadata file, the firmware binary file and all resource files listed in the xml element "IOLinkFWData → MetaInfo → AdditionalResourceCollection" (lower part of the yellow marked area in Figure 24). The root element shall contain the elements specified in Table 9.

Table 9 – Definition of the root elements

Item/attribute	M/O	Data type	Description
DocumentInfo	M	DocumentInfoT	This element contains common information about the file itself (version, release date, copyright).
MetaInfo	M	MetaInfoT	This element contains information about the Device firmware (Hardware IDs, revision, vendor details, description and additional resources).
Blob	M	BlobT	This element contains the binary data to be transferred to the Device via the FW-Update protocol. It has one mandatory attribute <i>fileName</i> . The <i>fileName</i> is the name (including an extension) of the file inside the firmware archive containing the binary data.
crc (signature)	M		<p>CRC signature across the xml metadata file, the firmware binary file and all resource files listed in the xml element "AdditionalResourceCollection".</p> <p>The CRC-32 algorithm shall be the same as with stamping of IODD files (see [2]). Before the actual calculation, the attribute "crc" is set to an empty string.</p> <p>CRC calculation shall be continuous across all data related to the firmware update file. The sequence is: Meta data file, binary Blob file and all files which are listed under attribute "AdditionalResourceCollection". The order is the same as for the "AdditionalResourceCollection" and all files shall be used as binary data for calculation.</p> <p>The generated CRC is then inserted into the attribute "crc".</p>
Key M = mandatory; O = optional			

7.4.4.4 Document information

The terms used in item "DocumentInfo" in Figure 24 are specified in Table 10.

Table 10 – Definition of elements in DocumentInfo

Item	M/O	Data type	Description
version	M	string: V\d+(\.\d+){1,7}	This attribute contains the version of the FW-Update file (for example V1.02). Its actual format is vendor specific.
releaseDate	M	date: \d{4}-\d{2}-\d{2}	The date information in the FW-Update file name shall correspond to the "releaseDate" attribute in the DocumentInfo element.
copyright	M	string	Vendor-specific copyright text.
Key M = mandatory; O = optional			

7.4.4.5 Firmware meta information

The terms used in item "MetaInfo" in Figure 24 are specified in Table 11.

643

Table 11 – Definition of elements in MetaInfo

Item	M/O	Data type	Description
vendorId	M	unsignedShort	This attribute is the firmware vendor ID obtained from IO-Link Consortium. It shall be the same as the value encoded in the Device direct page parameters in address 0x07 and 0x08 (VendorID).
vendorName	M	string	This attribute is mandatory. It is recommended that it complies with the corresponding Device parameter.
fwRevision	M	string	This attribute describes the version of the firmware data. Its format is vendor specific. It should not be used to compare firmware versions by the FW-Update-Software. It is recommended that this attribute complies with the corresponding Device parameter.
fwPassword Required	O	boolean	Set this attribute to TRUE in order to request the manufacturer password during the FW-Update process. Default = "false".
fwActivation RetryCount	C	uint16	Default value is "3" if this variable is not present. Variable shall be present, if the value is larger. Host tool shall either use "3" or the indicated value when showing the retry counting while waiting for the Device (see INTERNAL ITEMS in Table 17).
FWDescription Collection	O		This element is optional. It consists of one or more FirmwareDescription elements.
FWDescription	M		This element contains any vendor specific information about the FW-Update file (release notes, for which Device types it is, etc.). The FW-Update-Software can show the information after the FW-Update file is available. It has two mandatory attributes <i>xml:lang</i> and <i>description</i> . The <i>xml:lang</i> attribute describes the language of the description according to ISO 639-1:2002 (standard two-letter format such as "en", "de" etc.) and shall be unique across all <i>FirmwareDescription</i> inside <i>FirmwareDescriptionCollection</i> . The <i>description</i> attribute contains the text of the description.
HardwareIdKey Collection	M		This collection contains HardwareIdKeys. The ID links the FW-Update data to the Devices for which it can be used. The Devices shall have the corresponding ISDU parameter with the same ID. FW-Update-Software should read it from the Device and check whether it matches with one of the IDs in the file. The ID in the Device and one of the IDs in the file shall be the same to permit the update of the Device with the data in the file. The IDs in the file can end with a wildcard symbol "*" (idPattern attribute of each HardwareIdKey element). That means the ID in the Device shall match the string in front of "*". The characters after can differ.
HardwareIdKey	M	See idPattern	This is one of the elements in <i>HardwareIdKeyCollection</i> . It describes one of the hardware variants for which this firmware can be used for update. The element has one mandatory attribute <i>idPattern</i> . The <i>idPattern</i> attribute contains the text of this hardwareIdKey according to the pattern [A-Za-z][A-Za-z0-9_-]*[A-Za-z0-9*]. This element can also have optionally <i>productName</i> and <i>productId</i> string attributes. It is recommended that the attributes comply with the corresponding Device parameter.
AdditionalResource Collection	O		This element is optional. It consists of one or more resource elements. In case of supplemented IODDs all related files shall be packed within a zipped file.
Resource	M	[a-zA-Z\d_-.]+	This element describes a file with additional resources (e.g. pictures, lookup tables etc.). It has two attributes <i>id</i> and <i>fileName</i> . The <i>id</i> attribute is a unique identifier of the resource without any meaning. The <i>fileName</i> attribute is the name of the resource file inside the firmware archive restricted by the pattern "([a-zA-Z\d_-.]+)".
InfoMessage Collection	O		It consists of one or more InfoMessage elements.
InfoMessage	M		This element contains any vendor specific user instructions which will appear after successful completion of the FW-Update process (hints for dealing with data storage, sensor recalibration instructions, etc.). It has two mandatory attributes "xml:lang" and "description". The "xml:lang" attribute describes the language of

Item	M/O	Data type	Description
			the description according to ISO 639-1:2002 (standard two-letter format such as "en", "de" etc.) and shall be unique across all "InfoMessage" inside "Info Message Collection". The "description" attribute contains the text of the InfoMessage.
Key M = mandatory; O = optional			

7.5 Bootload management

7.5.1 Main activities

From a user's point of view, the entire firmware update (FW-Update) procedure consists of a sequence of six main activities as already indicated in Figure 21:

- Identification of the connected Device
- Acquisition of the FW-Update file
- Compatibility verification between Device and file
- Password entry and check (optional)
- Execution of firmware update
- Finalization and activation

Each activity is part of the FW-Updatesoftware tool and specified separately in clauses 7.5.2 to 7.5.7. The activities can be implemented for implicit automatic execution without user interaction except in case of faults or explicit execution as illustrated in Figure 34.

7.5.2 Device identification

The activity diagram of Device identification is shown in Figure 25. It refers to item 16 in the use case diagram in Figure 21.

The parameters HardwareRevision and FirmwareRevision are mandatory for this profile (see Table 12).

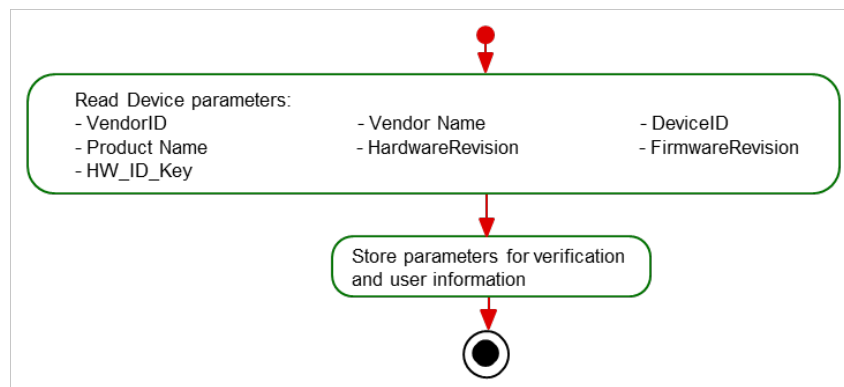


Figure 25 – Identification activity

The mandatory Device parameters VendorID, Vendor Name, DeviceID, Product Name, and HW_ID_Key are retrieved from the connected Device and stored locally for further activities.

7.5.3 Acquisition of the FW-Update file

Manufacturers or vendors shall provide FW-Update files in a standardized manner specified in this document (see 7.4). Figure 26 shows the activity diagram for the acquisition and the integrity check of the FW-Update file.

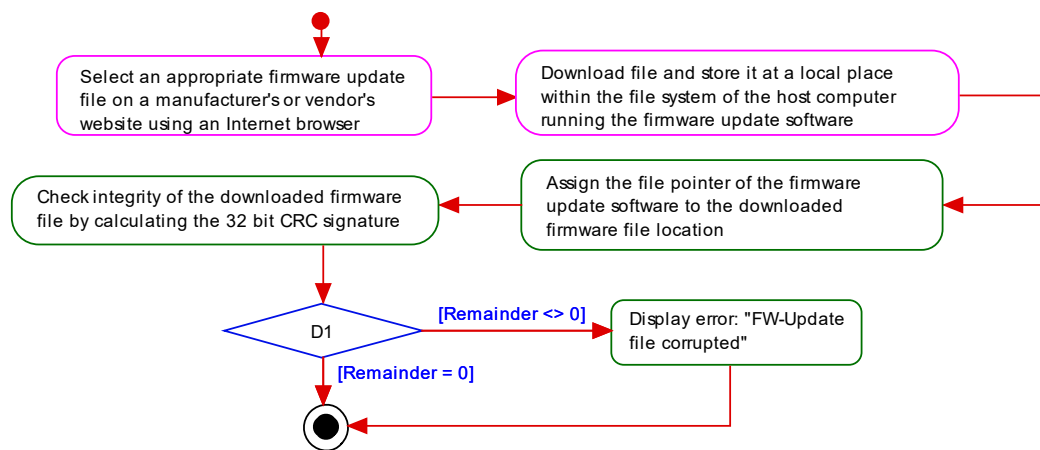


Figure 26 – FW-Update file acquisition and check

Activities in purple borders can be outside the FW-Update software tool.

7.5.4 Verification of FW-Update file compatibility

The Device parameters VendorID and HW_ID_Key are used for the compatibility check of the connected Device and the FW-Update file. The HW_ID_Key parameter of the Device shall match one of the HW_ID_Keys within the HW_ID_Key_Collection in the Meta information of the FW-Update file (see Table 10). Figure 27 shows the corresponding activity diagram.

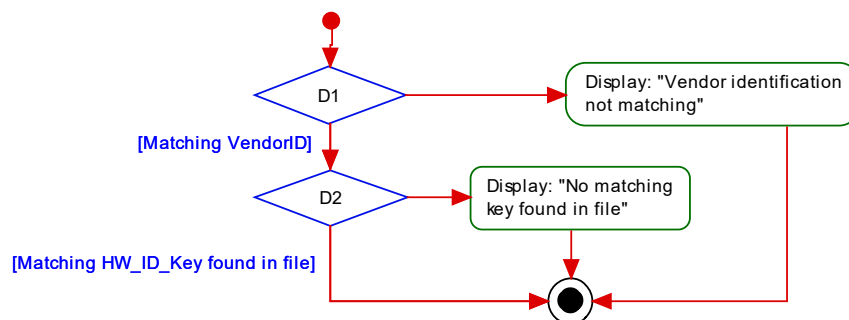


Figure 27 – Verification of compatibility

7.5.5 Password entry and check (optional)

The Device provides an optional parameter "FW-Password" (see 7.6.7). The FW-Update software expects a user entry, whenever the attribute "fwPasswordRequired" is TRUE within the FW-Update file (see Table 11). In case of multiple Devices of same type to be updated, it is possible to store a successful password value and to skip re-entries (see 7.9.1).

Figure 28 shows the corresponding activity diagram.

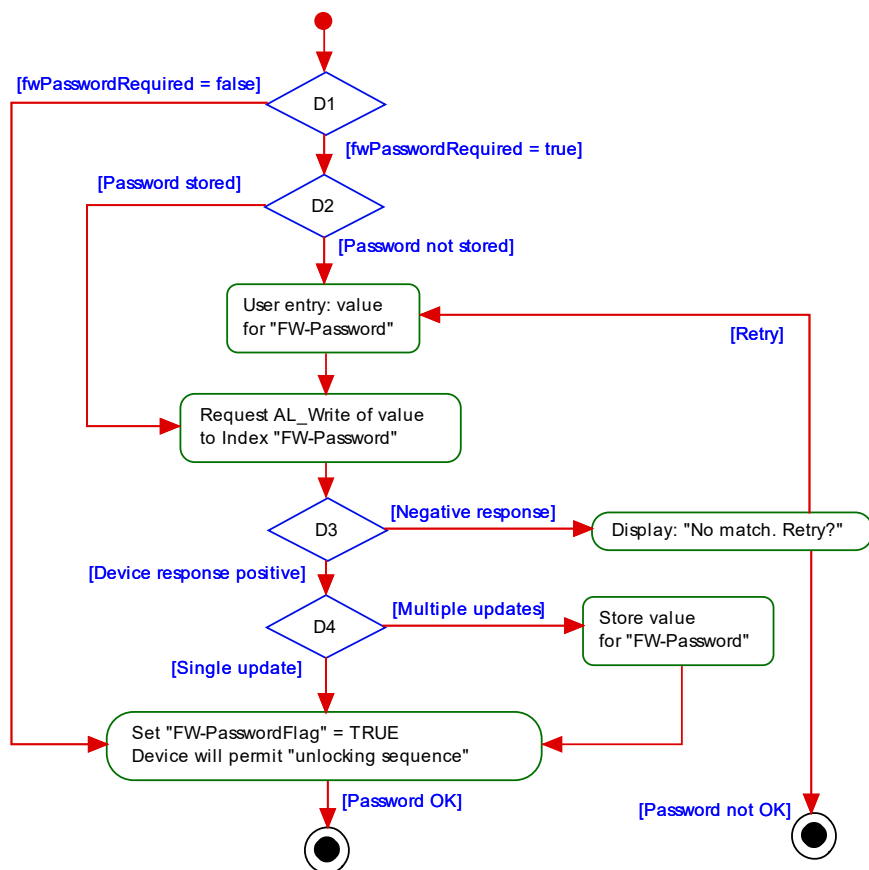


Figure 28 – Password check

If password entry has been OK, the flag "FW-PasswordFlag" shall be set (see 7.7.2).

7.5.6 FW-Update via bootloader

This activity corresponds to a protocol with states and transitions and is thus specified in a separate clause 7.7.

7.5.7 Finalization and activation

If an error occurs during FW-Update the user shall be informed via a fault indication. Retries shall always begin with verification (see 7.5.4). Automatic retries shall be limited. Figure 29 shows the activity.

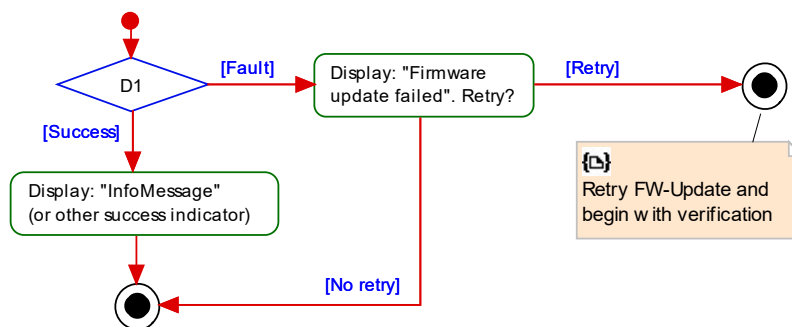


Figure 29 – Finalization and activation

7.6 Definitions and constraints

7.6.1 Initial Device operation (OPERATE/PREOPERATE)

Normally, a Device reaches the OPERATE state automatically without any tool intervention. Thus, the preferred state for FW-Update is the OPERATE state, where it is possible to chose the cycle time and optimize the update performance (see Annex D). If a Master port is configured to inspection level "TYPE_COMP" or "IDENTICAL", where the DeviceID shall match the configured value, the Device will stop in PREOPERATE.

FW-Update is also possible in this state. In this case the predefined minimum recovery times from Table A.8 in [1] shall be used.

7.6.2 Bootloader

As soon as the existing technology firmware of the Device is unlocked (see 7.6.9), the Device activates the Bootloader including the BLOB transfer mechanism. This means that the bootload mode is active after a communication reset of the Device.

Once the Bootloader is active, the Device shall react on the wake-up request of the Master and comply with the message handling of [1].

The Bootloader can be functionally downsized as specified in 7.10.1.

7.6.3 IODD for Bootload mode

A second IODD for FW-Update is not required. A Device comes already with an IODD containing the necessary information such as the parameter description for HW_ID_Key (see Annex A).

7.6.4 M-sequence types

In bootload mode it is mandatory to support M-sequence type "TYPE_0". This M-sequence type supports the transfer of one octet On-request Data (OD) per message, which gives it 75% overhead. This is applicable for the phases STARTUP, PREOPERATE and OPERATE.

For higher download speeds it is recommended to implement one of the following M-sequence types:

- TYPE_1_V code 6, 8 octets OD, no PD (28% overhead)
- TYPE_1_V code 7, 32 octets OD, no PD (9% overhead)

A manufacturer/vendor can decide to implement M-sequence types with PD (Process Data).

7.6.5 DeviceID versus Boot_DeviceID

In bootload mode the manufacturer shall change the DeviceID to a unique value within the manufacturer specific used range of values to indicate the Device is in bootload mode.

7.6.6 VendorID

The VendorID shall not be changed in bootload mode.

7.6.7 FW-Update specific parameters

The Device shall provide the additional parameters listed in Table 12 for the FW-Update profile ("conditional"). The parameters use Indices reserved for Device profiles and supplement Table B.8 in [1].

Table 12 – Device parameters reserved for FW-Update

Index (dec)	Object name	Access	Length	Data type	M/O/C	Definitions
...						
0x43BD (17341)	FW-Pass-word	W	variable	StringT	O/C	64 octets (UTF-8) is maximum length.
0x43BE (17342)	HW_ID_Key	R	variable	StringT	C	64 octets (UTF-8) is maximum length. Pattern is [A-Za-z][A-Za-z0-9_-]*[A-Za-z0-9*].

Index (dec)	Object name	Access	Length	Data type	M/O/C	Definitions
0x43BF (17343)	Bootmode Status	R	1 octet	UIntegerT	C	
...						
Key M = mandatory; O = optional; C = conditional						

7.6.7.1 FW-Password

This parameter shall be "write only". Device manufacturers shall set the attribute "fwPasswordRequired" = TRUE within the FW-Update file (see Table 11). The Device expects an AL_Write of the correct password value to the "FW-Password" Index prior to the unlocking of the firmware/bootloader (see 7.5.5). In case of an incorrect password, the Device shall return the ErrorCode 0x8030 – *Parameter value out of range*.

7.6.7.2 HW_ID_Key

This profile-conditional read only parameter shall be used for the identification of valid FW-Update files. The HW_ID_Key will be checked against the meta information "HardwareId-KeyCollection", wherein one of the listed HW_ID_Keys should match in order to start an update process (see Table 11).

Format example: SDAT-MHS-160

7.6.7.3 BootmodeStatus

This profile-conditional read only parameter shall be used as a flag to indicate whether the Bootloader is active or inactive. Table 13 shows the coding.

Table 13 – Coding of BootmodeStatus

Code	Definition
0x00	Bootloader inactive
0x01	Bootloader active
0x02 to 0xFF	Reserved

7.6.8 FW-Update specific SystemCommands

The required FW-Update SystemCommands are listed in Table 14. This table supplements Table B.9 in [1]. A Device equipped with the Bootloader mechanism shall support commands 0x50 to 0x52 in normal operation and 0x53 in Bootloader mode (see Table 14 and 7.7.4).

Table 14 – Coding of FW-Update SystemCommands

Command (hex)	Command (dec)	Command name	M/O/C	Definition
...				
0x50	80	BM_UNLOCK_S	C	Start unlocking sequence
0x51	81	BM_UNLOCK_F	C	Unlocking command 1
0x52	82	BM_UNLOCK_T	C	Unlocking command 2
0x53	83	BM_ACTIVATE	C	Stop communication and activate new firmware
...				

7.6.9 Unlocking sequence

Unlocking of the existing firmware is performed through a particular sequence of SystemCommands sent to the Device. This sequence is designed such that accidental flashing of the Device is very unlikely to occur. All the following steps are mandatory except password.

- 1) It is a precondition for the Device to be in communication (see 7.6.1).
- 2) The FW-Update software tool retrieved the HW_ID_Key from the Device (see 7.5.2 and 7.6.7.2) and checks it against the list in the FW-Update file (see 7.5.4).
- 3) Password protection is optional and shall not be disabled by a firmware update. It protects the Device from unauthorized unlocking.
- 4) In case of a match, the FW-Update software tool sends the sequence of SystemCommands shown in Figure 30 for unlocking the firmware and switching the Device to the Bootloader (see Figure 5).
The BM_UNLOCK_S SystemCommand is always accepted. Receiving this at any time does not generate an error and (re)starts the unlock sequence.
There is no time-out between the SystemCommands. ISDU communication and other access are permitted while the unlocking process is ongoing.
- 5) In case a communication startup is detected (see state "Startup_2" in Figure 38 in [1]) before Bootloader mode is entered, the unlocking sequence shall be restarted.
- 6) After successful reception of the sequence, the Device shall respond with a positive acknowledgment. Upon reception of a faulty sequence, the Device shall respond with a negative acknowledgment: ISDU error 0x8036 – *Function temporarily unavailable*.

7.6.10 Required BLOB_IDs

Table 15 shows the BLOB_IDs in use for firmware updates (see 6.5.2).

Table 15 – BLOB_IDs for FW-Update

Value (dec)	Definition
0	Idle transmission of a BLOB
1	FW-Update (write)

7.7 FW-Update protocol

7.7.1 Protocol layers

Figure 20 illustrates the transmission of firmware updates within an automation system with IO-Link. The "FWU_Trans_Layers" contain the necessary state machines using the BLOB transmission engines specified in 6.6.1 and 6.6.2 and provide the necessary protocol features between a Device and a PC-based tool.

7.7.2 Device FW-Update state machine

Figure 30 shows the state diagram of the "Bootloader_Layer" (see Figure 20). States 5 to 12 are nested in state 4 allowing these states to react on errors (T13 and T14) within all of these states (see 3.3.1).

After power-up, the Device performs a firmware check (state "0") before entering the regions (see 3.3.1) of state "Device_Operating_1".

The Device is able to receive the AL_Write service with the value of the "FW-Password" in state "DataExchange_2" (see 7.5.5). Once the written value matches the hard-coded value behind the Index of "FW-Password", the Device sets the "FW-PasswordFlag" = TRUE and is ready for the unlocking sequence.

It is important for the Device to switch into Bootmode (state 14) before any changes are applied to the internal flash memory affecting the technology application of the Device.

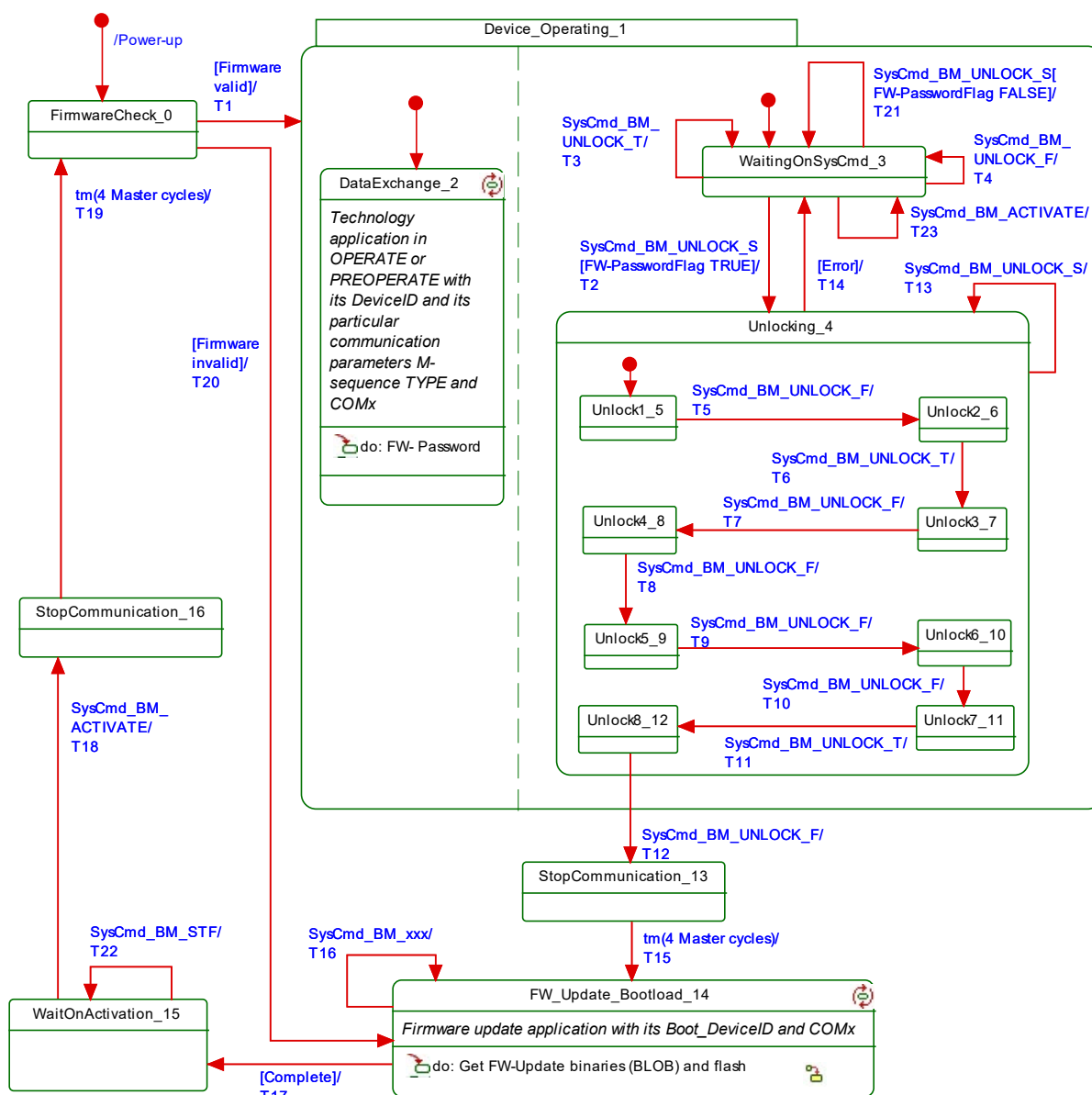


Figure 30 – Device FW-Update state machine

Not more than 2 subsequent M-sequences shall fail during flash processing to keep communication ongoing.

Any communication restart such as a fallback command shall cause a reset of the BLOB-transfer or Firmware Update state machine respectively. This shall not take place in case of an unintended communication interrupt.

The activity in state 14 "FW-Update_Bootload" is specified in 7.7.3. Table 16 shows the state transition tables of the Device bootloader state machine.

Table 16 – State transition tables of the Device bootloader state machine

STATE NAME	STATE DESCRIPTION
FirmwareCheck_0	After power-on of the Device, the validity of the Device's firmware is checked. The check is manufacturer specific. Valid firmware leads to a regular start of the Device according to [1]. Invalid firmware initiates the Bootloader and a FW-Update is started.
Device_Operating_1	Device is in OPERATE state (or PREOPERATE). See 7.6.1.
DataExchange_2	The technology application is exchanging data while using its appropriate M-sequence TYPES and COMx transmission rate.

STATE NAME	STATE DESCRIPTION
WaitingOnSysCmd_3	A separate additional firmware extension is started and waits on a particular System-Command "BM_Unlock_S", the first command of the unlocking sequence. Any other SystemCommand is ignored.
Unlocking_4	This superstate monitors the reception of the correct unlocking sequence. Any System-Command "BM_Unlock_S" will restart the entire sequence check. Any mismatch of received and expected SystemCommands will abandon the superstate and return to the previous state.
Unlock1_5	Expected SystemCommand is "BM_Unlock_F"
Unlock2_6	Expected SystemCommand is "BM_Unlock_T"
Unlock3_7	Expected SystemCommand is "BM_Unlock_F"
Unlock4_8	Expected SystemCommand is "BM_Unlock_F"
Unlock5_9	Expected SystemCommand is "BM_Unlock_F"
Unlock6_10	Expected SystemCommand is "BM_Unlock_F"
Unlock7_11	Expected SystemCommand is "BM_Unlock_T"
Unlock8_12	Expected SystemCommand is "BM_Unlock_F"
StopCommunication_13	The firmware extension causes the Device to stop current communication and thus forces the Master to restart communication via Wake-up.
FW_Update_Bootload_14	The Device re-establishes communication with new communication parameters (e.g. transmission rate). The Device BLOB state machine is activated. In this state the bootloader receives segment by segment of the FW-Update binary using the BLOB transmission state machine (see 6.6.1). Any SystemCommand will be ignored. Received segments are passed over to the flashing mechanism (see activity diagram in 7.7.3).
WaitOnActivation_15	After correct or incorrect flashing, the Device waits on a SystemCommand "BM_ACTIVATE".
StopCommunication_16	The Bootloader causes the Device to stop current communication and thus forces the Master to restart communication via Wake-up. After 4 Master cycles, the Bootloader switches to the new technology firmware.

814

TRANSITION	SOURCE STATE	TARGET STATE	ACTION
T1	0	1	Device start until OPERATE (or PREOPERATE), see 7.6.1. Set "FW-PasswordFlag" = FALSE (optional).
T2	3	4	–
T3	3	3	Device shall return ErrorCode 0x8036 – <i>Function temporarily unavailable</i> when host sends SysCmd_BM_UNLOCK_F in state "WaitingOnSysCmd_3"
T4	3	3	Device shall return ErrorCode 0x8036 – <i>Function temporarily unavailable</i> when host sends SysCmd_BM_UNLOCK_T in state "WaitingOnSysCmd_3"
T5	5	6	Acknowledgment
T6	6	7	Acknowledgment
T7	7	8	Acknowledgment
T8	8	9	Acknowledgment
T9	9	10	Acknowledgment
T10	10	11	Acknowledgment
T11	11	12	Acknowledgment
T12	12	13	Acknowledgment
T13	4	4	Received SystemCommand BM_Unlock_S. New unlocking sequence.
T14	5,6,7,8,9,10,11,12	3	SystemCommand "SysCmd_BM_xxx" out of sequence. Return ErrorCode 0x8036 – <i>Function temporarily unavailable</i>
T15	13	14	The firmware extension establishes a (manufacturer specific) Boot_DeviceID and optimized transmission parameters (e.g. COM3) for the download of the FW-Update binaries.
T16	14	14	Device shall return ErrorCode 0x8036 – <i>Function temporarily unavailable</i> when host sends SysCmd_BM_xxx in state "FW_Update_Bootload_14"

TRANSITION	SOURCE STATE	TARGET STATE	ACTION
T17	14	15	Acknowledgment. Return "Update completed"
T18	15	16	Acknowledgment. Return "Firmware check"
T19	16	0	Acknowledgment
T20	0	14	–
T21	3	3	Device shall return ErrorCode 0x8036 – <i>Function temporarily unavailable</i> , if host sends SysCmd_BM_UNLOCK_S and FW-PasswordFlag is FALSE.
T22	15	15	Device shall return ErrorCode 0x8036 – <i>Function temporarily unavailable</i> , if host sends SysCmd_BM_UNLOCK_S/T/F in state WaitOnAcitivation_15.
T23	3	3	Device shall return ErrorCode 0x8036 – <i>Function temporarily unavailable</i> , if host sends SysCmd_BM_ACTIVATE in state WaitingOnSysCmd_3.
INTERNAL ITEMS		TYPE	DEFINITION
Acknowledgment		Variable	
FW-PasswordFlag		Bool	See 7.5.5

7.7.3 Reception of binaries and flashing activity

The activity diagram for state 14 (FW_Update_Bootload) in Figure 31 shows the following actions:

- Reception of FW-Update file binaries (BLOB transfer),
- Flashing, and
- Firmware check.



Figure 31 – BLOB and flashing activity

The activity uses the BLOB state machines in 6.6 to receive the FW-Update file binaries segment by segment until "BLOB_Last" arrives. In case the Device does not have enough memory space, each segment will be flashed after reception. Otherwise, the Device receives the entire binaries prior to flashing.

NOTE For performance reasons, the flashing of segments should not slow down the transmission of segments.

7.7.4 Master behavior

In state 13 (StopCommunication) of the Device's FW-Update state machine, the firmware extension of the Device causes a communication interrupt and switches to Bootload mode. As a consequence, the Master will restart the communication but with different parameters:

- In STARTUP phase the Master reads the Direct Parameter page 1 parameters including the VendorID and the DeviceID which is now the Boot_DeviceID.
- The Master readjusts the corresponding port to an appropriate M-sequence and COM transmission rate in order to achieve optimized performance (see Annex D).

In state 16 (StopCommunication) of the Device's FW-Update state machine, the firmware extension of the Device causes another communication interrupt and switches to the updated firmware. As a consequence, the Master will restart the communication but with original parameters: DeviceID, M-sequence, and COM transmission rate.

7.7.5 Tool FW-Update state machine

Figure 32 shows the state diagram of the "T_FWU_Trans_Layer" (see Figure 20).

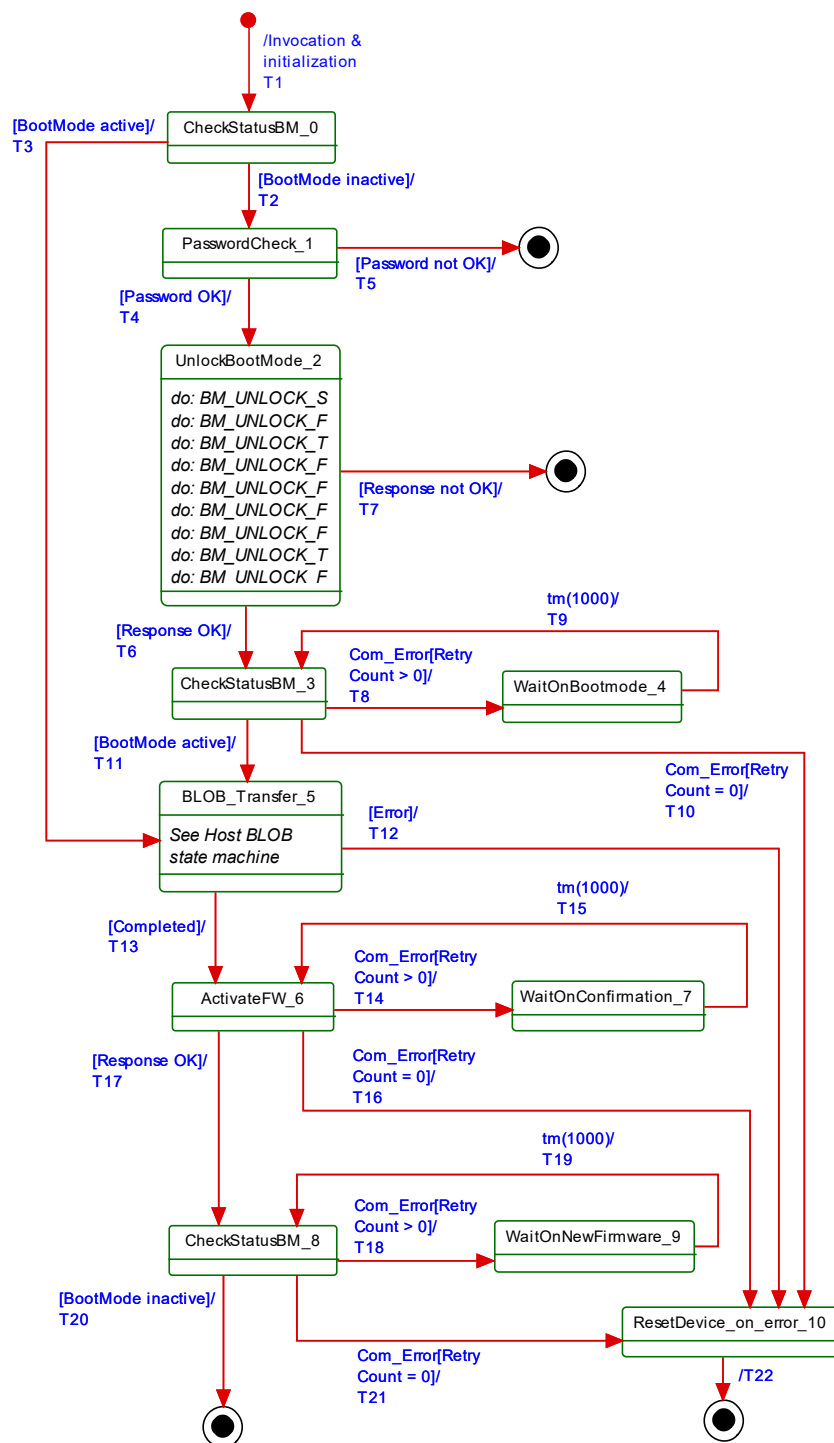


Figure 32 – Tool FW-Update state machine

The FW-Update software tool checks first whether the firmware of the Device is active and then enters the password check (option). After sending the unlocking sequence it starts the BLOB transmission (FW-Update binary). The following rules shall be observed:

- Any interruption of communication with subsequent restart of the communication shall not abort the BLOB transmission sequence.
- The host FW-Update tool shall not abort the BLOB when receiving a timed-out ISDU.
- The FW-Update can be aborted upon user request.

Table 17 shows the state transition tables of the Device FW-Update state machine.

854 **Table 17 – State transition tables of the Tool FW-Update state machine**

STATE NAME		STATE DESCRIPTION	
CheckStatusBM_0		Read Bootloader status via parameter "BootmodeStatus" (see 7.6.7.3)	
PasswordCheck_1		Check whether entered password matches password stored in Device (see 7.5.5)	
UnlockBootmode_2		Send unlocking sequence of SystemCommands to the Device (see 7.6.9)	
CheckStatusBM_3		Read Bootloader status via parameter "BootmodeStatus". If response is not OK, retry if count is not "0"	
WaitOnBootmode_4		Wait 1 s (1000 ms) before return to state 3	
BLOB_Transfer_5		Transfer body of the FW-Update file to the Device using the host BLOB state machine (see 6.6.2)	
ActivateFW_6		New firmware activated by means of SystemCommand "BM_ACTIVATE". If response is not OK, retry if count is not "0"	
WaitOnConfirmation_7		Wait 1 s (1000 ms) before return to state 6	
CheckStatusBM_8		Read Bootloader status via parameter "BootmodeStatus". If response is not OK, retry if count is not "0"	
WaitOnNewFirmware_9		Wait 1 s (1000 ms) before return to state 8	
ResetDevice_on_error_10		A SystemCommand 0x80 causes a move of the Device to Bootloader or technology FW	
TRANSITION	SOURCE STATE	TARGET STATE	ACTION
T1	init	0	–
T2	0	1	–
T3	0	5	–
T4	1	2	–
T5	1	exit	–
T6	2	3	RetryCount = Preset value (see INTERNAL ITEMS)
T7	2	exit	–
T8	3	4	RetryCount = RetryCount -1
T9	4	3	–
T10	3	10	–
T11	3	5	–
T12	5	10	–
T13	5	6	RetryCount = Preset value (see INTERNAL ITEMS)
T14	6	7	RetryCount = RetryCount -1
T15	7	6	–
T16	6	10	–
T17	6	8	RetryCount = Preset value (see INTERNAL ITEMS)
T18	8	9	RetryCount = RetryCount -1
T19	9	8	–
T20	8	exit	–
T21	8	10	–
T22	10	exit	–
INTERNAL ITEMS		TYPE	DEFINITION
RetryCount		Variable	Retry counter, starting from "Preset = 3" and decrementing to "0", if variable fwActivationRetryCount in MetaInfo (Table 11) is not present. Otherwise, use "Preset = indicated value", which shall be ≥ "3".
Bootloader active/inactive		Flag	Parameter "BootmodeStatus" (see 7.6.7.3).

855

856

INTERNAL ITEMS	TYPE	DEFINITION
Error	Variable	Possible errors: see BLOB state machines.
Com_Error	Variable	Master specific

7.7.6 Sequence charts

Figure 33 shows the sequence chart of the entire FW-Update procedure. For the sake of better readability, the involvement of the Master is not shown during and after the BLOB transfer.

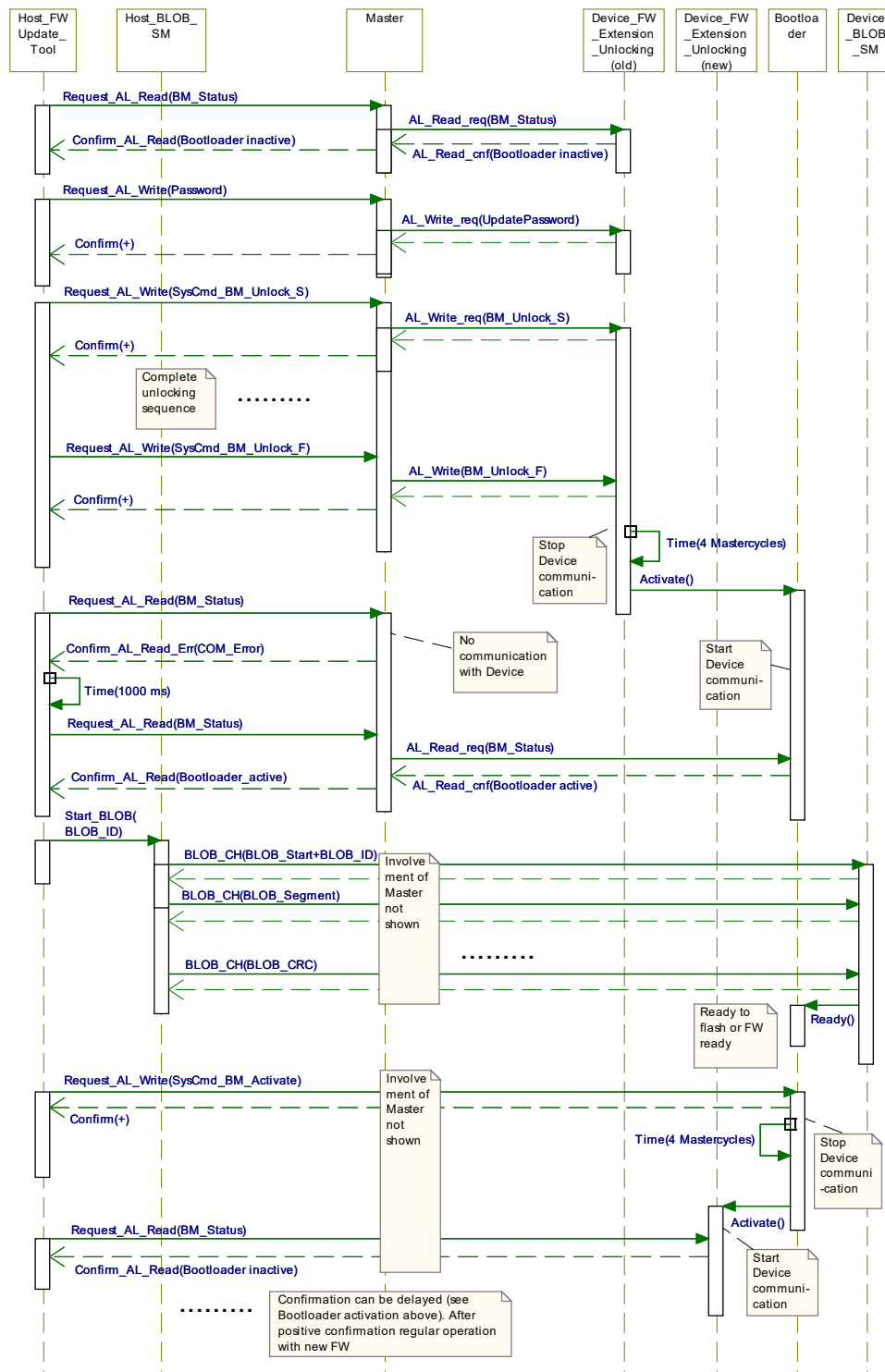


Figure 33 – Sequence chart of the FW-Update procedure

7.8 Validation/responsibility

The FW-Update tool manufacturer is responsible for the proper function of the tool and its conformity with this specification via manufacturer declaration based on the tests in [6] and in Annex E.

The Device manufacturer is responsible for the proper function of a Device after a FW-Update and the conformity with the IO-Link specifications via manufacturer declaration based on the tests in [6] and in Annex E. The tests include the unlocking and the Bootloader mechanisms.

The manufacturer provides information about appropriate corrective actions in case an update failed, e.g. recovery, customer service, etc.

7.9 Common look and feel

7.9.1 Graphical user interface (GUI) and functions

Figure 34 illustrates exemplary the GUI functions shown in 7.3. The tab "Firmware" shall only be enabled or available if a Device supports FW-Update. The layout is conceptual and not mandatory. The aspect of multiple updates of several Devices of the same type is not shown.

It is possible to split the functions into a PC part for FW-Update file acquisition and pre-verification. A dedicated tool can then perform the second part comprising verification and FW-Update start. It is up to the manufacturer of the tool to provide appropriate indication (e.g. LED), for example the verification result and the update progress.

Topology	Identification	Parameter	Monitoring	Diagnosis	Firmware	Device Library																
Toplevel ... - Master - Port 1: Device aa - Port 2: Device b - ... - Port n: Device xxx	1. Identification: Device: <table border="1"> <tr><td>Vendor name</td><td></td></tr> <tr><td>Device ID</td><td></td></tr> <tr><td>FW-Revision</td><td></td></tr> <tr><td>...</td><td></td></tr> </table>	Vendor name		Device ID		FW-Revision		...		FW-Update file: <table border="1"> <tr><td></td><td>✓</td></tr> <tr><td></td><td>✓</td></tr> <tr><td></td><td>✓</td></tr> <tr><td></td><td>✓</td></tr> </table>		✓		✓		✓		✓				Vendor 1 - Device a V1.03 - Device b V1.23 - Device c V2.00 - ... Vendor 2 - Device aa V0.99 - Device bb V1.1.2 - Vendor n - Device xxx V2.3.04 - Device yyy V1.3 - Device zzz V123
Vendor name																						
Device ID																						
FW-Revision																						
...																						
	✓																					
	✓																					
	✓																					
	✓																					
	2. FW-Update file locator: <input type="text" value="..UltraSensLtd-UD3675B2-20150808-IOLFW1.0.iolfw"/> <input type="button" value="Select"/>				✓																	
	3. Password (optional): <input type="text" value="Enter (display) password"/> <input type="button" value="Submit"/>				✓																	
	4. Verify compatibility: <input type="text" value="Result"/> <input type="button" value="Verify"/>				✓																	
	5. Update: <input type="text" value="Progress indication"/> <input type="button" value="Start"/>				✓																	
	6. Status: <input type="text" value="Report"/> <input type="button" value="Read"/>				✓																	
				<input type="button" value="Exit"/>																		

Figure 34 – Exemplary illustration of the GUI functions

The connected Device is identified in step 1 and relevant parameters are displayed (see fields below "Device:"). The FW-Update file is selected in step 2. This example assumes files that are already acquired via Internet, data media, or other means.

If the attribute "fwPasswordRequired" is set TRUE, the entry of a password is necessary in step 3. In case of multiple updates the password is stored for the sake of ease of use (see 7.5.5 for details). Selection of another FW-Update file overwrites password entries.

A crosscheck between the Device parameters and the items in "HardwareIdKey" (Table 11) is performed in step 4 and leads to the display of the items in step 1 (see fields below "FW-Update file:").

The update is performed in step 5 via the "Start" button and progress is indicated. The button is only activated if the verification in step 4 has been successful. The progress indicator shall be displayed as a bar or "%" for the transferred share of the total of octets.

A detailed report can be retrieved via the "Read" button in step 6. This report can comprise information about the successful update or error information. It can also comprise an individual "InfoMessage" provided by the manufacturer (see Table 11).

The FW-Update application can be finalized via the Exit button. This button is disabled during an ongoing firmware update process.

7.9.2 Multi-language terms

Table 18 shows the terms and the definitions that should be used whenever a graphical user interface is implemented in a FW-Update tool.

Text for other languages can be retrieved from the IO-Link website (www.io-link.com).

Table 18 – Human interface terms

Term	Definition
Identification	Device parameters involved are specified in 7.5.2
Device	[1]
FW-Update file	See 7.4
VendorID	[1], Table B.1
Vendor Name	[1], Table B.8
DeviceID	[1], Table B.1
Product Name	[1], Table B.8
HardwareRevision	[1], Table B.8
FirmwareRevision	[1], Table B.8
HW_ID_Key	See 7.6.7.2
FW-Revision	[1], Table B.8
Locator	Directory path to the FW-Update file
Select	Browse to FW-Update file
Password	See 7.5.5
Submit	Transmit entry
Verify	See 7.5.4
Compatibility	See 7.5.4
Result	Detailed verification information: success or mismatches
Progress indicaton	See 7.9.4
Start	Initiate Bootloader and update process
Status	Detailed update information: success or abort
Report	User information: "InfoMessage", see Table 11
Read (print)	Detailed success report with relevant parameters or error/abort report
Exit	Quit FW-Update. Disabled during an ongoing firmware update process.

7.9.3 Error displays of FW-Update tools

Table 19 shows a list of possible error displays of FW-Update tools.

908

Table 19 – Error displays of FW-Update tools

Operational phase	Error/status display	Corrective action (Help)
Startup/connection	Device does not support firmware update (Profile)	Read and check parameter "Profile-Characteristic"
	Firmware update version not supported	Read and check parameter "Profile-Characteristic"
File select	Firmware update file is corrupted and cannot be opened	Contact customer/sales service
	Tool does not support firmware update file version	Contact customer/sales service
	Selected file is not a firmware update file	Look for a file with extension .iolfw
Password	Password required	Enter password
	Password incorrect	Correct entry
Verification	Connected Device not supported by file	Look for appropriate file or contact customer/sales service
	Status: Connected Device supported, check manually	Check parameter by parameter
BLOB and ISDU	Invalid BLOB_ID	BLOB_ID shall be "1" or "0" (see Table 15)
	Incorrect BLOB_ID	Check IODD (see Annex A)
	ISDU transaction failed	Retry, then contact customer/sales service
Update/status	Status: Update may take several minutes	Abort or wait
	Status: Update completed	Read/print report ("InfoMessage"), see Table 11
	Update failed	Retry, then contact customer/sales service
	Update denied due to an active process	Retry later
	Time-out	Retry, then contact customer/sales service
	CRC error	Retry, then contact customer/sales service
	Activation of new firmware failed	Repeat and/or contact customer/sales service

909

910 7.9.4 Indications

911 The following rules apply for indications:

- 912 a) Progress indicator shall show % of full length of the data object to be transmitted.
- 913 b) Successful completion shall be indicated by a green symbol (preferably check mark).
- 914 c) Faults shall be indicated on displays by a red cross ⊗.
- 915 d) A Device's indicator should indicate an unsuccessful FW-Update if possible. The indication
- 916 shall be described in the user manual.

917 7.10 Recommended strategies**918 7.10.1 Design aspects**

919 The Device shall behave in Bootload mode as Device communicating with any Master that con-

920 forms with [1]. However, in Bootload mode functional downsizing of the Device to a certain

921 degree with respect to [1] is possible.

922 The layer structure of a Device is shown in Figure 5 of this profile and in Figure 35 below ([1]).

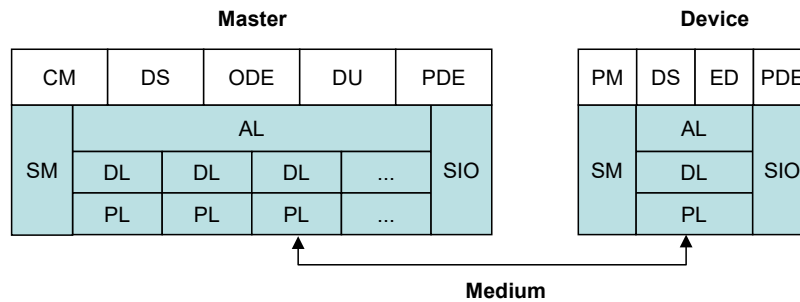


Figure 35 – Layer structures of Master and Device

For conformance, the Physical Layer (PL), the Data Link Layer (DL) and the Application Layer (AL) shall be implemented in Bootload mode (BM). The System Management (SM) and the SIO-Mode shall be implemented as far as required for the wake-up to switch between operational modes and to switch back into the regular technology firmware of the Device (TFW).

Regarding Device applications and features, only the Parameter Manager (PM) shall be present in Bootload mode. The applications

- Data Storage (DS),
- Event Dispatcher (ED), and
- Process Data Exchange (PDE)

are not required and should not to be implemented in Bootload mode.

The features

- Device access locks,
- Dynamic parameters, and
- Block parameterization

are not required and can be omitted in Bootload mode.

The following tables provide specifications of items which affect Devices supporting the BLOB & FW-Update profile.

Table 20 shows the affected items of the Direct Parameter page 1.

Table 20 – Affected items of the Direct Parameter page 1

Item	Subindex	Ref	BM	TFW	Comment
VendorID 1 + 2	0x07, 0x08	[1], Table B.1	M	M	Identical in BM and TFW
DeviceID 1 + 2 + 3	0x09, 0x0A, 0x0B	[1], Table B.1	M	M	Not identical in BM and TFW
Key for BM and TFW M = mandatory, O = optional, C = conditional, – = not permitted					

Table 21 shows the affected MasterCommands.

Table 21 – Affected MasterCommands

Item	Command	Ref	BM	TFW	Comment
MasterIdent	0x95	[1], Table B.2	M	M	Standard behavior
DeviceIdent	0x96	[1], Table B.2	M	M	Standard behavior

Item	Command	Ref	BM	TFW	Comment
DeviceStartup	0x97	[1], Table B.2	M	M	Standard behavior
ProcessDataOutput Operate	0x98	[1], Table B.2	–	M	Ignored in BM
DeviceOperate	0x99	[1], Table B.2	M	M	Standard behavior
DevicePreoperate	0x9A	[1], Table B.2B1.2	M	M	Standard behavior
Key for BM and TFW M = mandatory, O = optional, C = conditional, – = not permitted					

Table 22 shows the affected ISDUs.

Table 22 – Affected ISDUs

Item	Index	Ref	BM	TFW	Comment
HW_ID_Key	0x43BE	6.6.7	M	C	–
BLOB_ID	0x0031	5.5.1	M	O	–
BLOB_CH	0x0032	5.5.1	M	O	–
BootmodeStatus	0x43BF	6.6.7	M	C	–
Update-Password	0x43BD	6.6.7	–	C	–
Profile Characteristic	0x000D	[1], Table B.8	M	O	See clause 5
Hardware Revision	0x0016	[1], Table B.8	M	M	Can be different according to company policy
FirmwareRevision	0x0017	[1], Table B.8	M	M	Refers in BM to the revision of the Bootloader
ProductName	0x0012	[1], Table B.8	M	M	–
ProductId	0x0013	[1], Table B.8	O	O	–
Key for BM and TFW M = mandatory, O = optional, C = conditional, – = not permitted					

Table 23 shows the affected SystemCommands.

Table 23 – Affected SystemCommands

Item	Index	Ref	BM	TFW	Comment
Device reset	0x80	[1] Table B.9	M	O	BM: reset blob transfer
BM_UNLOCK_S	0x50	6.6.8	–	M	–
BM_UNLOCK_F	0x51	6.6.8	–	M	–
BM_UNLOCK_T	0x52	6.6.8	–	M	–
BM_ACTIVATE	0x53	6.6.8	M	–	–
Key for BM and TFW M = mandatory, O = optional, C = conditional, – = not permitted					

Table 24 shows the affected services.

Table 24 – Affected services

Item	Index	Ref	BM	TFW	Comment
FW-Password check		6.7.2	–	C	Mandatory for FW-Update profile
FW validity check		6.7.2	M	C	–
BLOB_Transfer		5.5	M	O	–
Firmware flashing		–	M	O	–
Key for BM and TFW M = mandatory, O = optional, C = conditional, – = not permitted					

7.10.2 Distribution to multiple destinations within the Device

It is within manufacturer's responsibility to manage multiple destinations within the Device via one single BLOB (FW-Update file).

7.10.3 Compatibility levels

Manufacturer has a possibility to achieve compatibility via the parameter HW_ID_Key and the meta information within the FW-Update file (see 7.4.4.5). Further compatibility levels are within manufacturer's responsibility if needed.

Annex A (normative) IODD extensions

A.1 Overview

The IODD syntax is defined within the IODD specification and is enforced by the companion XML schema file. Thus, any change to the IODD syntax requires a new version of the IODD specification and schema.

Creating a new IODD version is a complex, time consuming task. To make things worse, each new version causes labor for the tool manufacturers and may increase complexity for users. As a consequence, new versions of the IODD specification and schema are only published in case of major updates.

Therefore, the IODD definitions for BLOB transfer and FW-Update are done in two steps. An intermediate step allows this profile to go ahead and work with IODDs based on the existing IODD V1.1 syntax. The longterm step is recommended for the proper integration into a future version of the IODD specification and schema. That is why IO-Link profiles are always based on this unchanged standard.

The exact contents of the IODD for a device with profiles from this specification as well as the instructions to the IODD Checker are specified in the IODD snippets (see also A.4). Appendices A.2 and A.3 provide an example of the description of the profiles in the IODD. However, the snippets that are also used by the checker to check IODDs with the profiles specified here are normative.

Note: If errors in the representation of the profiles in the IODD become known after the specification has been released and require immediate changes, the snippets can be updated without having to adapt the specification.

A.2 Binary Large Objects (BLOBs)

BLOBs are transferred via a transfer channel described by a pair of indices: One used for the identification of the BLOB and one used for the actual segmented data transfer. A Device could have more than one BLOB transfer channels for simultaneous transfer of more than one BLOB at a time. Within this profile, only one channel is defined, using the indices BLOB_ID 0x0031 (49) and BLOB_CH 0x0032 (50). These indices are reserved for profiles.

The Device adheres to this profile and supports BLOB transfer (but not necessarily FW-Update), whenever the attribute Features/@profileCharacteristic is present and contains the value 0x0030.

```
<Features ... profileCharacteristic="...48..." />
```

The following variable shall be inserted into the VariableCollection section:

```
<Variable index="49" accessRights="ro" id="V_BT_BLOBID">
  <Datatype xsi:type="IntegerT" bitLength="16">
    (see next paragraphs for detailed explanations)
  </Datatype>
  <Name textId="TN_V_BT_BLOBID"/>
</Variable>
```

The following shall be inserted into the ExternalTextCollection/PrimaryLanguage section:

```
<Text id="TN_V_BT_BLOBID" value="ID of the BLOB that is currently transferred"/>
```

NOTE 1: Text for other languages can be retrieved from the IO-Link website (www.io-link.com).

The variable "V_BT_BLOBID" shall not be referenced by any Menu.

The index for BLOB_CH is not described in the IODD. The IODD syntax cannot describe complex communication protocols using a single index as transport channel. Tools shall assume the presence of BLOB_CH, whenever "V_BT_BLOBID" is present.

Which BLOB IDs are supported by the Device is expressed by enumerating the allowed values of V_BT_BLOBID as SingleValues within the DataType element. Negative values are used for BLOBs readable from the Device, and positive values are used for writing BLOBs to the Device. Zero shall always be defined (BLOB idle).

Example:

```
<SingleValue value="-4096">
  <Name textId="TN_SV_BT_BLOBID_Manufacturer_Read"/>
</SingleValue>
<SingleValue value="0">
  <Name textId="TN_SV_BT_BLOBID_idle"/>
</SingleValue>
<SingleValue value="4096">
  <Name textId="TN_SV_BT_BLOBID_Manufacturer_Write"/>
</SingleValue>
```

The texts referenced by Name/@textId shall describe the usage of the supported BLOBs, e.g.

```
<Text id="TN_SV_BT_BLOBID_Manufacturer_Read" value="Calibration values (read)"/>
<Text id="TN_SV_BT_BLOBID_idle" value="Idle, no BLOB transfer active"/>
<Text id="TN_SV_BT_BLOBID_Manufacturer_Write" value="Calibration values (write)"/>
```

NOTE 2: The IODD snippets can also contain other national languages. If this is the case, then the texts must be implemented in the IODD exactly as defined in the snippets. If the desired languages are not included in the snippets, then the texts shall be translated accordingly by the manufacturer or the default language English applies.

NOTE 3: See the example IODD "IO-Link-5121-BLOBTransferProfileDevice-xxxxxxx-IODD1.1.xml" (www.io-link.com).

A.3 FW-Update

The IODD of the Device shall only reflect the behavior of the Device in normal operation, not within the Bootload mode. An IODD describing the behavior in Bootload mode is not required.

The Device adheres to this profile and supports FW-Update, whenever the attribute Features/@profileCharacteristic is present and contains the value 0x0031.

```
<Features ... profileCharacteristic="...49..." />
```

Furthermore, the Device shall support the SystemCommand values required for activating the Bootload mode. This can be expressed by inserting SingleValues to the V_SystemCommand variable:

```
<StdVariableRef id="V_SystemCommand">
  <SingleValue value="80">
    <Name textId="TN_SV_FU_SystemCommand_BM_UNLOCK_S"/>
  </SingleValue>
  <SingleValue value="81">
    <Name textId="TN_SV_FU_SystemCommand_BM_UNLOCK_F"/>
  </SingleValue>
  <SingleValue value="82">
    <Name textId="TN_SV_FU_SystemCommand_BM_UNLOCK_T"/>
  </SingleValue>
</StdVariableRef>
```

The SystemCommand "BM_Activate" is not available in normal operation and thus shall not be described here.

Activating the boot load mode via a sequence of writes of these values to the variable V_SystemCommand is reserved for built-in functionality of tools. The values shall not be writeable individually by the user. Therefore, IODDs shall not contain Buttons for V_SystemCommand which use these values as buttonValue.

The texts referenced by Name/@textId shall be:

```

1063     <Text id="TN_SV_FU_SystemCommand_BM_UNLOCK_S" value="Start unlocking sequence"/>
1064     <Text id="TN_SV_FU_SystemCommand_BM_UNLOCK_F" value="Unlocking command 1"/>
1065     <Text id="TN_SV_FU_SystemCommand_BM_UNLOCK_T" value="Unlocking command 2"/>

```

1066 NOTE 6: Text for other languages can be retrieved from the IO-Link website (www.io-link.com).

1067 The following profile specific variable shall be described whenever the firmware password fea-
 1068 ture is supported:

```

1069     <Variable index="17341" accessRights="wo" id="V_FU_FW_Password">
1070         <Datatype xsi:type="StringT" encoding="UTF-8" fixedLength="16"/>
1071         <Name textId="TN_V_FU_FW_Password"/>
1072     </Variable>

```

1073 The value of "fixedLength" is manufacturer specific.

1074 The following two profile specific variables are mandatory:

```

1075     <Variable index="17342" accessRights="ro" id="V_FU_HW_ID_Key">
1076         <Datatype xsi:type="StringT" encoding="UTF-8" fixedLength="16"/>
1077         <Name textId="TN_V_FU_HW_ID_Key"/>
1078         <Description textId="TD_V_FU_HW_ID_Key"/>
1079     </Variable>

```

1080 The value of "fixedLength" is manufacturer specific.

```

1081     <Variable index="17343" accessRights="ro" id="V_FU_BootmodeStatus" dynamic="true">
1082         <Datatype xsi:type="UIntegerT" bitLength="8">
1083             <SingleValue value="0">
1084                 <Name textId="TN_SV_FU_BootmodeStatus_inactive"/>
1085             </SingleValue>
1086             <SingleValue value="1">
1087                 <Name textId="TN_SV_FU_BootmodeStatus_active"/>
1088             </SingleValue>
1089         </Datatype>
1090         <Name textId="TN_V_FU_BootmodeStatus"/>
1091     </Variable>

```

1092 The HW_ID_KEY shall be referenced in the user interface identification menu. A default value
 1093 is optional.

```

1094     <VariableRef variableId="V_FU_HW_ID_Key"/>

```

1095 The texts referenced by Name/@textId shall be:

```

1096     <Text id="TN_V_FU_FW_Password" value="Firmware Password"/>
1097     <Text id="TN_V_FU_HW_ID_Key" value="Hardware Identification Key"/>
1098     <Text id="TD_V_FU_HW_ID_Key" value="Identifies the hardware for compatibility check with the
1099 firmware to be installed."/>
1100     <Text id="TN_V_FU_BootmodeStatus" value="Bootmode Status"/>
1101     <Text id="TN_SV_FU_BootmodeStatus_inactive" value="Bootloader is inactive"/>
1102     <Text id="TN_SV_FU_BootmodeStatus_active" value="Bootloader is active"/>

```

1103 NOTE 7: The IODD snippets can also contain other national languages. If this is the case, then the texts must be
 1104 implemented in the IODD exactly as defined in the snippets. If the desired languages are not included in the snippets,
 1105 then the texts shall be translated accordingly by the manufacturer or the default language English applies.

1106 NOTE 8: See the example IODD "IO-Link-5120-FirmwareUpdateProfileDevice-xxxxxxx-IODD1.1.xml" ([www.io-
 1107 link.com](http://www.io-link.com)).

1108 A.4 IODD Checker V1.1.12 or higher

1109 Since the IODD specification V1.1.4 and the package 2024, the implementation of all profiles
 1110 in the IODD is described by so-called IODD snippets.

1111 The IODD snippets contain all elements that must be included in the IODD for the respective
 1112 profile. Since a profile can have multiple characteristics and optional function classes, additional
 1113 information is required to describe this. Information is also required for the checker to check.

1114 This type of check is supported from IODD Checker V1.1.12 or higher.

1115 The rules for the snippets can be found in the IODD specification V1.1.4 [2].

Annex B (normative) Calculation of CRC signatures

B.1 Overview of CRC-32 signatures

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

If a generator polynomial $g(x) = p(x) \cdot (1 + x)$ 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 [23]).

Figure 20 shows the topology to be considered for the investigations on efficiency of the CRC signature (layered transmission protocols). It should be unlikely that the CRC generator polynomial used for BLOB transfer matches the CRC generator polynomial used in the underlying transmission systems, for example IO-Link, fieldbus, or PC connection.

Table B.1 shows the characteristics of the chosen CRC-32 generator polynomial.

Table B.1 – CRC-32 generator polynomial

Polynomial		Data length (bits)	Hamming distance	Proper- ness	Referenz	Remark
"Normal"	"Reversed"					
0x741B8CD7	0xEB31D82E	< 16360	≥ 6	n/a	[23]	~ 2 kB
		< 114663	≥ 4			~ 14 kB
NOTE Representations: "Normal": high order bit omitted, most-significant bit leftmost; "Reversed": high order bit omitted, but most-significant bit rightmost. "Reversed" allows for algorithm without branch (see Figure B.1.						

B.2 Implementation considerations

B.2.1 Overview

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

B.2.2 Bit shift algorithm (32 bit)

For the 32-bit CRC signature, the reversed form 0xEB31D82E is used as the generator polynomial. The number of data bits may be odd or even. Figure B.1 shows the bit shift algorithm in "C" programming language.

```
uint32_t crc32_bitwise(char *data, size_t length, uint32_t previousCrc32 = 1)
{
    uint32_t crc = ~previousCrc32;
    int j;
    const uint8_t* current = (const uint8_t*) data;
    while (length-- > 0)
    {
        crc ^= *current++;
        for (j = 0; j < 8; j++)
        {
            crc = (crc >> 1) ^ (-(int32_t)(crc & 1) & 0xEB31D82E);
        }
    }
    return ~crc;
}
```

Figure B.1 – Bit shift algorithm in "C" language

The variables used in Figure B.1 are specified in Table B.2.

Table B.2 – Definition of variables used in Figure B.1

Variable	Definition
data	octet buffer containing the data to be protected
length	number of octets in data (starting with index 0) used for crc signature processing
previousCrc32	CRC signature value before new data are applied; seed value = 1
0xEB31D82E	polynomial in reversed presentation (most-significant bit rightmost)
crc	updated CRC signature value

B.2.3 Octet shift and Look-up tables (32 bit)

The corresponding function "crc32_octetwise" in "C" language is shown in Figure B.2. This function can be 20 times faster. However, the lookup table requires memory space.

```
uint32_t crc32_octetwise(uint8_t const * current, uint32_t length,
uint32_t previousCrc32 = 1, const uint32_t crc32Lookup[256])
{
    uint32_t crc = ~previousCrc32;
    while (length-- > 0)
        crc = (crc >> 8) ^ crc32Lookup[(crc & 0xFF) ^ *current++];
    return ~crc;
}
```

Figure B.2 – CRC-32 signature calculation using a lookup table

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

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

Variable	Definition
current	octet buffer containing the data to be protected
length	number of octets in data (starting with index 0) used for crc processing
previousCrc32	CRC signature value before new data are applied
crc32Lookup	table lookup function with argument
crc	updated CRC signature value

The function in Figure B.2 uses the lookup table in Table B.4.

Table B.4 – Lookup table for CRC-32 signature calculation (hexadecimal)

CRC-32 lookup table (0 to 255)							
00000000	9695C4CA	FB4839C9	6DDDFD03	20F3C3CF	B6660705	DBBBFA06	4D2E3ECC
41E7879E	D7724354	BAAFBE57	2C3A7A9D	61144451	F781809B	9A5C7D98	0CC9B952
83CF0F3C	155ACBF6	788736F5	EE12F23F	A33CCCF3	35A90839	5874F53A	CEE131F0
C22888A2	54BD4C68	3960B16B	AFF575A1	E2DB4B6D	744E8FA7	199372A4	8F06B66E
D1FDAE25	47686AEF	2AB597EC	BC205326	F10E6DEA	679BA920	0A465423	9CD390E9
901A29BB	068FED71	6B521072	FDC7D4B8	B0E9EA74	267C2EBE	4BA1D3BD	DD341777
5232A119	C4A765D3	A97A98D0	3FEF5C1A	72C162D6	E454A61C	89895B1F	1F1C9FD5
13D52687	8540E24D	E89D1F4E	7E08DB84	3326E548	A5B32182	C86EDC81	5EFB184B
7598EC17	E30D28DD	8ED0D5DE	18451114	556B2FD8	C3FEEB12	AE231611	38B6D2DB
347F6B89	A2EAAF43	CF375240	59A2968A	148CA846	82196C8C	EFC4918F	79515545
F657E32B	60C227E1	0D1FDAE2	9B8A1E28	D6A420E4	4031E42E	2DEC192D	BB79DDE7
B7B064B5	2125A07F	4CF85D7C	DA6D99B6	9743A77A	01D663B0	6C0B9EB3	FA9E5A79

CRC-32 lookup table (0 to 255)							
A4654232	32F086F8	5F2D7BFB	C9B8BF31	849681FD	12034537	7FDEB834	E94B7CFE
E582C5AC	73170166	1ECAFC65	885F38AF	C5710663	53E4C2A9	3E393FAA	A8ACFB60
27AA4D0E	B13F89C4	DCE274C7	4A77B00D	07598EC1	91CC4A0B	FC11B708	6A8473C2
664DCA90	F0D80E5A	9D05F359	0B903793	46BE095F	D02BCD95	BDF63096	2B63F45C
EB31D82E	7DA41CE4	1079E1E7	86EC252D	CBC21BE1	5D57DF2B	308A2228	A61FE6E2
AAD65FB0	3C439B7A	519E6679	C70BA2B3	8A259C7F	1CB058B5	716DA5B6	E7F8617C
68FED712	FE6B13D8	93B6EEDB	05232A11	480D14DD	DE98D017	B3452D14	25D0E9DE
2919508C	BF8C9446	D2516945	44C4AD8F	09EA9343	9F7F5789	F2A2AA8A	64376E40
3ACC760B	AC59B2C1	C1844FC2	57118B08	1A3FB5C4	8CAA710E	E1778C0D	77E248C7
7B2BF195	EDBE355F	8063C85C	16F60C96	5BD8325A	CD4DF690	A0900B93	3605CF59
B9037937	2F96BDFD	424B40FE	D4DE8434	99F0BAF8	0F657E32	62B88331	F42D47FB
F8E4FEA9	6E713A63	03ACC760	953903AA	D8173D66	4E82F9AC	235F04AF	B5CAC065
9EA93439	083CF0F3	65E10DF0	F374C93A	BE5AF7F6	28CF333C	4512CE3F	D3870AF5
DF4EB3A7	49DB776D	24068A6E	B2934EA4	FFBD7068	6928B4A2	04F549A1	92608D6B
1D663B05	8BF3FFCF	E62E02CC	70BBC606	3D95F8CA	AB003C00	C6DDC103	504805C9
5C81BC9B	CA147851	A7C98552	315C4198	7C727F54	EAE7BB9E	873A469D	11AF8257
4F549A1C	D9C15ED6	B41CA3D5	2289671F	6FA759D3	F9329D19	94EF601A	027AA4D0
0EB31D82	9826D948	F5FB244B	636EE081	2E40DE4D	B8D51A87	D508E784	439D234E
CC9B9520	5A0E51EA	37D3ACE9	A1466823	EC6856EF	7AFD9225	17206F26	81B5ABEC
8D7C12BE	1BE9D674	76342B77	E0A1EFBD	AD8FD171	3B1A15BB	56C7E8B8	C0522C72
NOTE This table contains 32 bit values in hexadecimal representation for each value (0 to 255) of the argument a in the function crc32Lookup [a]. The table should be used in ascending order from top left (0) to bottom right (255).							

1165

1166 **B.2.4 Seed value**1167 The seed value shall be "1" (see *previousCrc32* = 1 in Figure B.1 and Figure B.2).

Annex C (informative)

Compression, authentication, and encryption

C.1 Compression

C.1.1 General

The advantage of compressing data objects is the smaller size leading to a faster BLOB download via the relatively slow IO-Link transmission.

There is no need to receive the complete compressed firmware image before starting to decompress and flash it; several algorithms are available which only need read access to the already decompressed part of the firmware image, and only very little additional memory. Most of the complexity of data compression is within the compressor, which typically resides on a PC with ample resources, while the decompressor is fairly simple. Several algorithms are so simple that decompression is fast enough to keep pace with the transmission without additional delays.

Depending on the available memory (Flash and RAM) and computing power, some algorithms are more suitable than others (see [9]).

The principle of data compression (applies for both lossy and lossless compression)

The compressor has a model of the data. This model is used to predict the next few octets to be compressed. The incoming data is compared with the prediction, and only the difference (the "surprise") needs to be stored. The better the model, the better the prediction, and the better the data can be compressed. The model can be predefined /static, if the kind of data to be compressed is known in advance. But most of the time, the model is derived from the already processed data, and dynamically updated. So the compressor does an extrapolation of the already received data to predict the next data. The difference to the prediction is then coded in a compact form. For example with the "Deflate" algorithm (C.1.3), the model is "repeated sequences of byte-strings" (Lempel-Ziv) and Huffman coding.

The decompressor also has the model, which it updates the same way the compressor does, from the already decompressed data. Step-by-step, it generates a prediction and then decodes the compressed stream and applies the differences to the prediction to regenerate the octets of the original uncompressed stream.

The following clauses C.1.2 to C.1.4 contain some suggestions for compression algorithms, sorted in increasing requirements for memory and computing power.

C.1.2 LZRW (Lempel-Ziv-Ross Williams)

This family of algorithms "LZRW" was invented by Ross Neil Williams (see [10], [11] and Dr. Ross's compression crypt in [12] with source code). For example, the decompressor for LZRW1 is 22 lines of "C" code, the while loop within is 12 lines of code. Apart from the source and destination pointers, only a few local variables are required.

C.1.3 Deflate (Lempel-Ziv 77 + Huffman Coding)

The "Deflate" algorithm is explained in [13] and specified in RFC 1951 [14]. It was invented by Phil Katz for PKZIP (see [15]). Source code is available for the zlib compression library [16]. If only one stream has to be compressed, the ZLIB format can be used (see RFC 1950 in [17]). For combining multiple streams into one compressed stream, the gzip format can be used (see RFC 1952 in [18]).

"Deflate" has a better compression than "LZRW". However, it is more complex and slower.

C.1.4 LZO (Lempel-Ziv-Oberhumer)

The "LZO" algorithm was invented by Markus F.X.J. Oberhumer and is explained in [19]. Source code is available in [20].

"LZO" is much faster than "Deflate", but more complex and requires a buffer of 8 kilooctets (or 64 kilooctets) for the decompressor.

C.2 Authentication via signatures

Adding a CRC signature to the binary code is a very effective measure against *unintended* modification (random corruption) of the firmware image. It allows detection of firmware alterations ever since the CRC signature was last calculated. However, since the CRC algorithm is specified and well-known, anyone can maliciously alter the data and recalculate the CRC signature such that the checks return OK.

A more sophisticated measure is required to secure the firmware against *intended* modifications, for example tampering, hacking, or malicious attacks. In this case, a cryptographic signature should be added to ensure authenticity of the firmware image without modification other than by the manufacturer.

How to create a cryptographic signature?

In a first step, the data stream to be secured can be optionally transformed to a "normalized" form. For example, if the data stream is known to be XML code, white space and comments would be filtered out and numbers would be normalized by removing leading zeros, trailing zeroes, and space characters.

In a second step, the transformed data stream is processed using a cryptographic hash algorithm. A property of hash algorithms is the ease of computing a hash value for a given stream of data. However, it is nearly impossible to generate another stream of data producing the same hash.

In a third step, an asymmetric encryption algorithm is used. Asymmetric means, it uses a pair of keys, a private one and a public one. What has been encrypted with one key can only be decrypted with the other key. It is nearly impossible to decrypt a data stream only knowing the key used for encryption. Thus, the computed hash value from the second step is now encrypted with the private key of the signer.

The overall cryptographic signature attached to the data stream finally contains the hash value, the encrypted hash value, the public key of the signer, and meta information about the used algorithms for the three steps, including meta information about the purpose of the signature.

How to check the signature?

The receiver performs the same transformation and processing of the hash value of the data stream with the algorithms indicated within the signature. The check fails if the hash value deviates from the one stored in the signature.

Decryption of the encrypted hash value contained in the signature can be performed using the public key stored in the signature and using the encryption algorithm indicated in the signature. The check fails if the result deviates from the hash value within the signature.

In case of passed checks, the receiver knows that the data hasn't been changed ever since someone with this public key signed it. However, the authenticity of this public key is not yet known. Thus, the public key stored in the signature shall be verified. Either, the public key has already been distributed via a secure channel and is therefore well-known, or some secure protocol is used to check the key online. The online check is also useful in cases where the key pair had been withdrawn due to a compromised private key.

An overview of algorithms and advice, which one to use and which one not to use, can be retrieved from [21].

In the context of FW-Update, the signature would be created by the Device manufacturer and added to the firmware image prior to its release to the customer. After downloading the firmware to the Device, the bootloader would check the signature and deny the activation of the firmware if not valid.

Conclusions?

e) A transformation step in case of a firmware image is not necessary.

- 1264 f) There are no special requirements for the hash algorithm and the asymmetric encryption
1265 algorithm. It may be sufficient for the Bootloader to support only one fixed hash and encryp-
1266 tion algorithm due to restricted memory space. Thus, meta information about algorithms in
1267 use can also be omitted.
- 1268 g) It may also be sufficient for the Bootloader to contain the public key of the manufacturer
1269 for a comparison:
1270 - No possibility for an online check
1271 - The key is not a secret
1272 - If the key in the Bootloader can be altered, the downloaded firmware can be altered also
- 1273 h) There is no provision for the withdrawal of a key pair in case of a compromised private key
1274 since the public key is built in. Therefore, the key pair used for the authenticity of firmware
1275 images should not be used for any other purposes. It can be reasonable using separate
1276 key pairs for each Device type or Device family to limit the damage.
- 1277 i) Recommendation is to use authentication only in case of a threat of unofficial or tampered
1278 firmware.

1279 **C.3 Encryption of FW-Update files**

1280 Encryption of FW-Update files can be desirable to prevent from an analysis of a Device's func-
1281 tion. Algorithms can be found in [21].

1282 However, it is recommended not to use encryption since the Device requires decrypted code to
1283 perform its intended functions. The key for decryption must be permanently available within the
1284 Device and thus would be accessible for an intruder.

Annex D (informative) Performance estimations

D.1 Assumptions

Clause 6.4.1 describes how the ISDU mechanism of IO-Link is used for the transfer of BLOBs and in particular of FW-Update binaries which can be one up to several 100×10^3 octets. The time required for a BLOB transfer via ISDU dominates the duration of a FW-Update provided the time of an upperlevel fieldbus system is negligible. This estimation focuses only on IO-Link specific parameters. All other related parameters are not considered. This means that the estimations in this Annex E are considerably shorter than in reality.

The length of a binary large object (BLOB) to be transferred between a Tool and a Device is called $Length_{BLOB}$ (see 6.5.3.2). The payload of an ISDU can be in this case up to 231 octets ("ExtLength"). However, a particular Device defines its maximum ISDU payload via a parameter and therefore a BLOB is usually transferred in a segmented manner using the data type OctetString as illustrated in Figure 7.

Segment by segment of the BLOB is placed as OctetString into the BLOB data area of the Write-request of the Master and transferred via the On-request Data communication channel using an appropriate M-sequence type. The Device returns its positive response also via the On-request Data communication channel (see [1]).

D.2 Estimated time for a BLOB transfer

The time required for a BLOB transfer is given by

- the number of ISDUs " N_{ISDU} ", multiplied by
- the number of M-sequences " N_{Mseq} " required to transfer one ISDU, and multiplied by
- the cycle-time " t_{CYC} " for one ISDU transfer.

The result of this performance estimation is shown in equation (1) that allows evaluating the duration of a BLOB transfer " t_{BLOB} ".

$$t_{BLOB} = N_{ISDU} \times N_{Mseq} \times t_{CYC} \quad (1)$$

D.3 Required number of ISDU transfers

The length of the BLOB, divided by the length of the OctetString (decremented by the header), and rounded up to the next integer value determines the number " N_{ISDU} " of ISDUs required for a BLOB transfer as shown in equation (2).

$$N_{ISDU} = \left\lceil \frac{Length_{BLOB}}{Length_{OctetStringT} - 1} \right\rceil \quad (2)$$

D.4 Number of M-sequences per ISDU

ISDUs are cyclically transferred in OPERATE mode using the On-request Data (OD) channel. Its capacity depends on the M-sequence type. TYPE_1_V with 8 or 32 octets of OD are recommended for the BLOB transfer in case of firmware update. For the number of M-sequences " N_{Mseq} " per ISDU transfer, the number of M-sequences for both the write request and the read request are required (see Figure 7).

For the write request this number is "ExtLength" of the ISDU (length of OctetString + 4) divided by the octets of OD (8 or 32). For the read request this number is 2 divided by the octets of OD (8 or 32). The quotients are rounded up to the next integer value and summarized as shown in equation (3).

$$N_{Mseq} = \left\lceil \frac{ExtLength}{OD} \right\rceil + \left\lceil \frac{2}{OD} \right\rceil \quad (3)$$

Table D.1 lists the number of M-sequences required for various lengths of OctetString values and for different M-sequence types.

Table D.1 – Number of M-sequences per ISDU

Length of OctetString	TYPE_0 (OD = 1)	TYPE_1_V (OD = 8)	TYPE_1_V (OD = 32)
28	34	5	2
60	66	9	3
92	98	13	4
124	130	17	5
156	162	21	6
188	194	25	7
220	226	29	8

D.5 Recommended Master cycle times

The duration of an M-sequence depends on the transmission time of UART frames and certain delay times as shown in [1], Annex A.3.6. The (Master) cycle time cannot be shorter than the M-sequence time since there is an idle time between two M-sequences.

The estimation assumes a negligible idle time and the following parameter values for equation A.6 in [1]:

- m : Number of UART frames sent to the Device (corresponds to OD)
- n : Number of UART frames returned by the Device (corresponds to OD)
- t_A : Delay between end of stop bit of last UART frame sent and begin of start bit of first UART frame received (assumed value: $10 T_{BIT}$)
- t_1 : Delay between two UART frames sent (assumed value: $1 T_{BIT}$)
- t_2 : Delay between two UART frames received (assumed value: $3 T_{BIT}$)

Table D.2 shows recommended minimum cycle times with respect to transmission rates and M-sequence types.

Table D.2 – Recommended cycle times (t_{CYC})

Transmission rate (bit/s)	M-sequence types		
	TYPE_0 (OD = 1)	TYPE_1_V (OD = 8)	TYPE_1_V (OD = 32)
COM1	16,0 ms	33,6 ms	94,4 ms
COM2	2,9 ms	5,1 ms	12,8 ms
COM3	1,4 ms	1,7 ms	3,0 ms
NOTE The timings have been extended by 1 ms due to internal Master processing times.			

D.6 Conclusions

Annexes D.3 to D.5 provide the necessary values for equation (1) in Annex D.2 to calculate the timings for BLOB transfers.

Figure D.1 and Figure D.2 show the minimum timings for COM3 and COM2 transmissions at various configurations. "OD 8" refers to an M-sequence type with 8 octets of M-sequence data; "OctetString 220" refers to an OctetString length of 220 octets.

At a COM3 transmission rate, 250 000 octets require a minimum of 20 to 35 seconds.

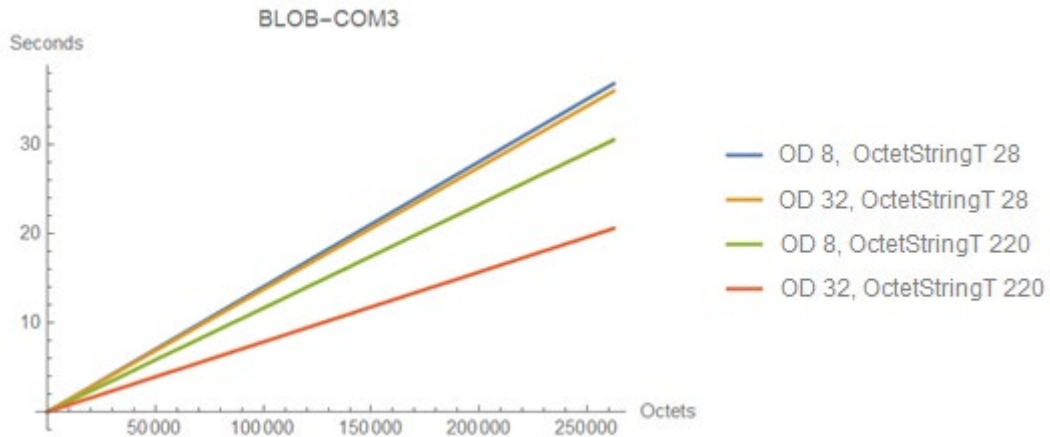


Figure D.1 – BLOB transfer timings for COM3

At a COM2 transmission rate, 30 000 octets require a minimum of 12 to 22 seconds.

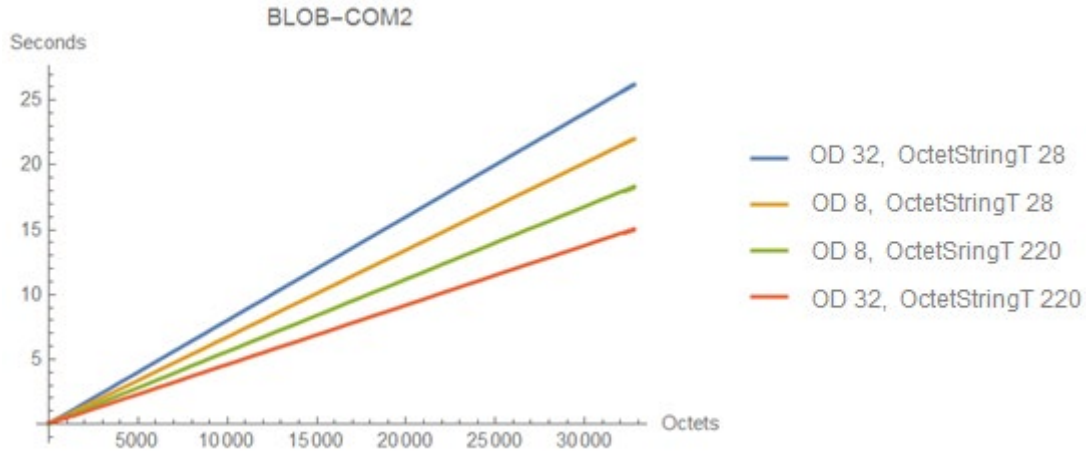


Figure D.2 – BLOB transfer timings for COM2

As expected, it turns out that the performance be best with large OctetString lengths and large On-request Data communication channels.

However, if for some reasons a shorter OctetString needs to be chosen (e.g. because of RAM limitations in the Device), it is preferable for COM2 Devices to choose an M-sequence type with 8 octets OD. It should be noted that selecting an M-sequence type with only 1 octet OD slows down the transfer speed by approximately a factor of 8. The timings provided in the diagrams shall be considered as lower limit. In real implementations, additional delays for transfer and storage shall be expected that can increase these timings by factors.

In case the Device needs the segment to be flashed immediately, a "busy" message is sent. " N_{Mseq} " increases by a rounded up t_{flash}/t_{CYC} . " t_{BLOB} " then follows equation (4):

$$t_{BLOB} = t_{CYC} \times \left\lceil \frac{Length_{BLOB}}{Length_{OctetString} - 1} \right\rceil \times \left(\left\lceil \frac{ExtLength}{OD} \right\rceil + \left\lceil \frac{2}{OD} \right\rceil + \left\lceil \frac{t_{flash}}{t_{CYC}} \right\rceil \right) \quad (4)$$

With typical flash (erase) times of 40 ms, t_{BLOB} increases by 275 % for COM3 and by 50 % for COM2.

Annex E (normative)

Profile specific test cases and tools

E.1 Concept, conventions, and preconditions

E.1.1 Concept

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

E.1.2 Conventions

The tests are performed using predefined BLOB_IDs as specified in Table E.1.

Table E.1 – Predefined test BLOB_IDs

BLOB_ID	Definition
DTWx_ID	BLOB_ID for Device write tests
DTRx_ID	BLOB_ID for Device read tests
HTWx_ID	BLOB_ID for host application write tests
HTRx_ID	BLOB_ID for host application read tests

E.1.3 Test exceptions

All Devices supporting the profile shall comply with the IO-Link test specification [6]. If a Device supports BLOB transfer, the test cases in Table E.2 shall be skipped during Device testing due to contradictions.

Table E.2 – Exempted test cases for Devices

Test case	Reasoning
SDCI_TC_0105	Device fails, because Index 0x32 BLOB_CH does not appear in the IODD and therefore the response is unexpected
SDCI_TC_0136	Device fails, because Index 0x32 BLOB_CH does not appear in the IODD and therefore the response is unexpected
SDCI_TC_0136	Device fails, because the SystemCommands BM_UNLOCK_S, BM_UNLOCK_F, BM_UNLOCK_T, BM_ACTIVATE will generate error responses if the state machine in Figure 30 is not in the corresponding states. The correct behaviour of these SystemCommands is tested with the help of test case BTFU_TC_0001 (see E.2.1)

E.2 Device implementations of BLOB transfer

E.2.1 Reserved BLOB_IDs for Device test

All Devices supporting the BLOB profile shall support these IDs and shall be able to provide the corresponding test payload (random numbers). The random numbers can be generated on the fly or can be taken from a look-up table. The random BLOB which is related to the particular ID shall be available after BLOB_Start. The algorithm to calculate the pseudo random code is described in E.6.5.

Table E.3 shows the reserved BLOB_IDs for the Device test. DTW1_ID, DTW2_ID, DTW3_ID are mandatory for Devices supporting BLOB "write". DTR1_ID, DTR2_ID, DTR3_ID are mandatory for Devices supporting BLOB "read".

Table E.3 – Reserved BLOB_IDs for the Device test

BLOB_ID (write)	BLOB_ID (read)	Definition	Size	Seed value
DTW1_ID = 4093	DTR1_ID = -4093	Test short transfer For this BLOB, Device shall return 5 octets of data generated by the specified random number generator, using seed value 1.	5 Octets	1
DTW2_ID = 4094	DTR2_ID = -4094	Test adaption of BLOB_LAST ISDU size For this BLOB, Device shall return 2,5 x (ISDU max. size octets minus one) of data generated by the specified random number generator, using seed value 2	(2,5 x max. ISDU size) -1	2
DTW3_ID = 4095	DTR3_ID = -4095	Test longer transfer For this BLOB, Device shall return 20 x (ISDU max. size octets minus one) of data generated by the specified random number generator, using seed value 3.	(20 x max. ISDU size) -1	3

E.2.2 Complete BLOB read (positive test)

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

Table E.4 – Complete BLOB read

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0001
Name	TCD_FWDL_BLOB_COMPLETE_UPLOAD
Purpose (short)	Test of a complete BLOB read (reading from device)
Equipment under test (EUT)	Device with support for BLOB transfer (profile characteristic 0x0030)
Test case version	1.0
Category / type	Device BLOB state machine test, test to pass (positive testing)
Occurrence	Conditional BLOB read (at least one BLOB_ID < 0 supported)
Specification (clause)	[1] see 6.5.2 Table 2, 6.5.3.1, 6.5.3.4, 6.5.3.5, 6.5.3.6, 6.5.3.8, 6.5.3.9, 6.6.1, Figure 18 (T5, T6, T7, T8, T9, T10)
Configuration / setup	Device-Tester with BLOB-Profile support
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	<p>Testing the transmission of complete data sequence from device. This test focuses on the state machine of Device BLOB transfer READ section including multiple CRC read and reading of BLOB_ID during the BLOB transfer.</p> <p>Test is done using BLOB_IDs DTR1_ID, DTR2_ID and DTR3_ID</p> <p>Device shall generate BLOB data to transfer using following pseudo random number generator.</p> <p>When the Device received BLOB_Start, it shall initialize the random number generator with the corresponding seed value (see declarations)</p>
Precondition	Master port set to IOL mode. No BLOB transmission active (State of Device BLOB state machine is Idle_0).
Procedure	<p>Repeat the following sequence three times with different BLOB_IDs DTR1_ID to DTR3_ID:</p> <ol style="list-style-type: none"> Write BLOB_Start via BLOB_CH with BLOB_ID id_x (with id_x one of DTR1_ID, DTR2_ID, DTR3_ID) Read BLOB_ID Read BLOB_CH (BLOB_Info_Read) Read BLOB_ID Read BLOB_CH several times, until Device responds with BLOB_Last <p>Additional stability test for DTW3_ID NOTE</p> <ul style="list-style-type: none"> - Interrupt IO-Link communication while one random segment is transferred - Device-Tester to stop communication with Device - Sleep 15 sec - Device-Tester to wake-up Device to OPERATE state - Continue BLOB Transfer <ol style="list-style-type: none"> Read BLOB_CH (BLOB_CRC) Read BLOB_CH (BLOB_CRC) Read BLOB_ID Write BLOB_Finish via BLOB_CH Read BLOB_ID
Input parameter	DTR1_ID, DTR2_ID, DTR3_ID
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	<p>Check Device response of each step-in procedure:</p> <ol style="list-style-type: none"> Positive ISDU write response Positive ISDU read response, value id_x. Check if ISDU length is 2. Positive ISDU read response providing BLOB_Info_Read. Store BLOB length to temporary variable blob_length. Check if blob_length equals defined size for the current BLOB. Check if ISDU length is 5. Positive ISDU read response, value id_x Positive ISDU read response returning BLOB_Segment or BLOB_Last. Check BLOB header. Function and subfunction must be correct. Sum up length of BLOB payload. After BLOB_Last is received, the number of received bytes must be equal to blob_length stored in step c). Check if received data equals the output of

	<p>the random number generator.</p> <p>f) Positive ISDU read response returning the CRC of BLOB. Check CRC against expected CRC. Check if ISDU length is 5</p> <p>g) Positive ISDU read response returning the CRC of BLOB. Check CRC against expected CRC. (This step checks, if T9 can be used several times)</p> <p>h) Positive ISDU read response, value id_x</p> <p>i) Positive ISDU write response</p> <p>j) Positive ISDU read response with value 0</p>
Test passed	All steps of evaluation return expected result. IOL communication is not interrupted while testing.
Test failed (examples)	Any of the evaluation steps fails. Negative response or comparison fails. No response or evaluations negative.
Results	
<p>NOTE Equivalent to SDCI_TC_0072. Switch to fallback does not fit, because Devices may clear a BLOB transfer when receiving a fallback command.</p>	

E.2.3 Complete BLOB write (positive test)

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

Table E.5 – Complete BLOB write

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0002
Name	TCD_FWDL_BLOB_COMPLETE_DOWNLOAD
Purpose (short)	Test of a complete BLOB write (writing to Device)
Equipment under test (EUT)	Device with support for BLOB transfer (profile characteristic 0x0030)
Test case version	1.0
Category / type	Device BLOB state machine test, test to pass (positive testing)
Occurrence	Conditional BLOB write (at least one BLOB_ID > 0 supported)
Specification (clause)	[1] see 6.5.2 Table 2, 6.5.3.2, 6.5.3.3, 6.5.3.5, 6.5.3.6, 6.5.3.8, 6.5.3.9, 6.6.1, Figure 18 (T5, T11, T12, T13, T14, T16)
Configuration / setup	Device-Tester with BLOB-Profile support
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	<p>Testing the write transmission of complete data sequence into Device. This test focuses on the state machine of Device BLOB transfer WRITE section reading of BLOB_ID during the BLOB transfer.</p> <p>Test is done using BLOB_IDs DTW1_ID, DTW2_ID and DTW3_ID.</p> <p>Data to transmit via BLOB is created by a pseudo random number generator. The random number generator is initialized with different seed values for each test. The Device should check the received data against the same random number generator running on the Device. In case of mismatch, the Device shall send a negative write response with error code 0x8030 – <i>Parameter value out of range</i>.</p> <p>When the Device received BLOB_Start, it shall initialize the random number generator with the corresponding seed value (see declarations)</p>
Precondition	Master port set to IOL mode. No BLOB transmission active (State of Device BLOB state machine is Idle_0).
Procedure	<p>Repeat the following sequence three times with different BLOB_IDs DTW1_ID to DTW3_ID:</p> <ol style="list-style-type: none"> Write BLOB_Start via BLOB_CH with BLOB_ID id_x (with id_x one of DTW1_ID, DTW2_ID, DTW3_ID) Read BLOB_ID Read BLOB_CH (BLOB_Info_Write). Store blob_size and max_isdu_size. Read BLOB_ID Write BLOB_CH with BLOB segment of size seg_size. (Skip this step, if b_size < seg_size). Use data from random number generator. Repeat this step, until all segments except BLOB_Last are sent. NOTE <p>Additional stability test for DTW3_ID</p> <ul style="list-style-type: none"> - Interrupt IO-Link communication while one random segment is transferred - Device-Tester to stop communication with Device - Sleep 15 sec - Device-Tester to wake-up Device to OPERATE state - Continue BLOB Transfer <ol style="list-style-type: none"> Write BLOB_CH with BLOB_Last and remaining data (telegram length depends on remaining bytes). Read BLOB_ID Write BLOB_CH with correct BLOB_CRC Read BLOB_ID Write BLOB_Finish via BLOB_CH Read BLOB_ID
Input parameter	DTW1_ID, DTW2_ID, DTW3_ID
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	<p>Check Device response of each step-in procedure:</p> <ol style="list-style-type: none"> Positive ISDU write response Positive ISDU read response with value id_x. Check if ISDU length is 2 Positive ISDU read response providing BLOB_Info_Write. ISDU length = 6? Positive ISDU read response with value id_x

TEST CASE RESULTS	CHECK / REACTION
	e) Positive ISDU write response for each segment (skipped for Test 1) f) Positive ISDU write response. g) Positive ISDU read response with value id_x h) Positive ISDU write response. i) Positive ISDU read response with value id_x j) Positive ISDU write response k) Positive ISDU read response with value 0
Test passed	All steps of evaluation return expected result. IOL communication is not interrupted while testing.
Test failed (examples)	Any of the evaluation steps fails. Negative response or comparison fails. No response or evaluations negative.
Results	
NOTE Equivalent to SDCI_TC_0072. Switch to fallback does not fit, because Devices may clear a BLOB transfer when receiving a fallback command.	

E.2.4 Write BLOB with invalid CRC

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

Table E.6 – Write BLOB with invalid CRC

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0003
Name	TCD_FWDL_BLOB_COMPLETE_DOWNLOAD_CRC_NOK
Purpose (short)	Test of a complete BLOB write with invalid CRC
Equipment under test (EUT)	Device with support for BLOB transfer (profile characteristic 0x0030)
Test case version	1.0
Category / type	Device BLOB state machine test, test to fail (negative testing)
Occurrence	Conditional BLOB write (at least one BLOB_ID > 0 supported)
Specification (clause)	[1] see 6.5.3, 6.5.3.2, 6.5.3.3, 6.5.3.6, 6.5.3.8, 6.5.3.9, 6.6.1, Figure 18 (T5, T11, T12, T13, T15, T17)
Configuration / setup	Device-Tester with BLOB-Profile support
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Testing the transmission of complete data sequence into device with invalid CRC provided by the host. This test focuses on the state machine of Device BLOB transfer WRITE section.
Precondition	Master port set to IOL mode. No BLOB transmission active (State of Device BLOB state machine is Idle_0).
Procedure	a) Write BLOB_Start via BLOB_CH, use BLOB_ID DTW1_ID b) Read BLOB_CH c) Write to BLOB_CH (all segments, including BLOB_LAST) DTW1_ID d) Write to BLOB_CH (incorrect BLOB_CRC) e) Read BLOB_ID f) Write BLOB_Abort via BLOB_CH g) Read BLOB_ID
Input parameter	DTW1_ID
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	Check Device response of each step-in procedure a) Positive ISDU write response b) Positive ISDU read response providing BLOB_Info_Write c) Positive ISDU write response (for all BLOB segments) d) ISDU error 0x8040 – <i>Invalid parameter set</i> (T15 of Device BLOB state machine) e) Positive ISDU read response with value id1 (Device BLOB state machine is in state "WaitOn_BLOB_Complete_6") f) Positive ISDU write response g) Positive ISDU read response with value "0"
Test passed	All steps of evaluation return expected result. IOL communication is not interrupted while testing.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.2.5 Write access to read-only parameter

Table E.7 defines the test conditions for this test case.

Table E.7 – Write access to read-only parameter

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0004
Name	TCD_FWDL_BLOB_WRITE_BLOB_ID_STATE_IDLE
Purpose (short)	Test handling of write access to read-only parameter
Equipment under test (EUT)	Device with support for BLOB transfer (profile characteristic 0x0030)
Test case version	1.0
Category / type	Required parameters for the BLOB transmission, test to fail (negative testing)
Occurrence	mandatory (BLOB read/write)
Specification (clause)	[1] see 6.5.1, 6.5.2
Configuration / setup	Device-Tester with BLOB-Profile support
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Test write access on BLOB_ID in state "Idle_0". Error response is expected.
Precondition	Master port set to IOL mode. No BLOB transmission active (State of Device BLOB state machine is Idle_0).
Procedure	a) Write "1" to BLOB_ID
Input parameter	–
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) Check if Device responds with ISDU error 0x8023 – <i>Access denied</i>
Test passed ^{4,4}	All steps of evaluation return expected result. IOL communication is not interrupted while test.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.2.6 Illegal BLOB_CH command

Table E.8 defines the test conditions for this test case.

Table E.8 – Illegal BLOB_CH command

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0005
Name	TCD_FWDL_BLOB_WRITE_BLOB_CH_T3_STATE_IDLE
Purpose (short)	Test of writing BLOB_CH with illegal command - error result
Equipment under test (EUT)	Device with support for BLOB transfer (profile characteristic 0x0030)
Test case version	1.0
Category / type	Device BLOB state machine test, test to fail (negative testing)
Occurrence	mandatory (BLOB read/write)
Specification (clause)	[1] see 6.5.2, 6.5.3.3, 6.5.3.5, 6.5.3.6, 6.5.3.7, 6.5.3.9, 6.6.1, Figure 18 (T3)
Configuration / setup	Device-Tester with BLOB-Profile support
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Test write illegal commands to BLOB_CH while BLOB state machine is in state "Idle_0". Error response expected.
Precondition	Master port set to IOL mode. No BLOB transmission active (State of Device BLOB state machine is Idle_0).
Procedure	a) Write BLOB_Finish command b) Read BLOB_ID c) Write BLOB_Abort command d) Read BLOB_ID e) Write BLOB_Segment (flow control 0x0, 0 byte payload) f) Read BLOB_ID g) Write BLOB_Last (flow control 0x0, 0 byte payload) h) Read BLOB_ID i) Write BLOB_CRC j) Read BLOB_ID
Input parameter	–
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	Check Device response of each step-in procedure: a) ISDU error 0x8030 – <i>Parameter value out of range</i> (T3 of Device BLOB state machine) b) Positive ISDU read response with value "0" c) ISDU error 0x8030 – <i>Parameter value out of range</i> (T3 of Device BLOB state machine) d) Positive ISDU read response with value "0" e) ISDU error 0x8030 – <i>Parameter value out of range</i> (T3 of Device BLOB state machine) f) Positive ISDU read response with value "0" g) ISDU error 0x8030 – <i>Parameter value out of range</i> (T3 of Device BLOB state machine) h) Positive ISDU read response with value "0" i) ISDU error 0x8030 – <i>Parameter value out of range</i> (T3 of Device BLOB state machine) j) Positive ISDU read response with value "0"
Test passed	All steps of evaluation return expected result. IOL communication is not interrupted while test.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.2.7 Invalid BLOB_ID

Table E.9 defines the test conditions for this test case

Table E.9 – Invalid BLOB_ID

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0006
Name	TCD_FWDL_BLOB_WRITE_BLOB_CH_T4_STATE_IDLE
Purpose (short)	Test of writing BLOB_CH with invalid BLOB_ID - error result
Equipment under test (EUT)	Device with support for BLOB transfer (profile characteristic 0x0030)
Test case version	1.0
Category / type	Device BLOB state machine test, test to fail (negative testing)
Occurrence	mandatory (BLOB read/write)
Specification (clause)	[1] see 6.5.1, 6.5.2 Table 2, 6.5.3.8, 6.6.1; Figure 18 (T4)
Configuration / setup	Device-Tester with BLOB-Profile support
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Testing BLOB_Start command via BLOB_CH with invalid BLOB_ID (-32768) in state "Idle_0". Error response expected.
Precondition	Master port set to IOL mode. No BLOB transmission active (State of Device BLOB state machine is Idle_0).
Procedure	a) Write BLOB_Start via BLOB_CH (Function 0xF, Sub function 0x1) with an inadmissible BLOB_ID.
Input parameter	a) id = -32768
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) Check response on write access.
Test passed	a) ISDU error 0x8030 – <i>Parameter value out of range.</i>
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.2.8 Read access on BLOB_CH in state "Idle_0"

Table E.10 defines the test conditions for this test case.

Table E.10 – Read access on BLOB_CH in state "Idle_0"

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0007
Name	TCD_FWDL_BLOB_READ_BLOB_CH_STATE_IDLE
Purpose (short)	Test read access on BLOB_CH in state "Idle_0". Error response is expected.
Equipment under test (EUT)	Device with support for BLOB transfer (profile characteristic 0x0030)
Test case version	1.0
Category / type	Device BLOB state machine test, test to fail (negative testing)
Occurrence	mandatory (BLOB read/write)
Specification (clause)	[1] see 6.5.1, 6.5.3, 6.6.1; Figure 18 (T2)
Configuration / setup	Device-Tester with BLOB-Profile support
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Test read access on BLOB_CH in state "Idle_0". Error response is expected.
Precondition	Master port set to IOL mode. No BLOB transmission active (State of Device BLOB state machine is Idle_0).
Procedure	a) Read BLOB_CH
Input parameter	–
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) Check response on read access
Test passed	a) ISDU error 0x8020 – <i>Service temporarily unavailable</i> (T2 of Device BLOB state machine)
Test failed (examples)	Any of the evaluation steps fails.

E.2.9 Invalid BLOB_Start during BLOB read

Table E.11 defines the test conditions for this test case.

Table E.11 – Invalid BLOB_Start during BLOB read

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0008
Name	TCD_FWDL_BLOB_Start_CMD_STATE_INADMISSIBLE_UP
Purpose (short)	Test of BLOB_Start command in inadmissible operating states for BLOB read - error result
Equipment under test (EUT)	Device with support for BLOB transfer (profile characteristic 0x0030)
Test case version	1.1
Category / type	Device BLOB state machine test.
Occurrence	Conditional BLOB read (at least one BLOB_ID < 0 supported)
Specification (clause)	[1] see 6.5.2, 6.5.3.1, 6.5.3.4, 6.5.3.5, 6.5.3.8, 6.6.1, Figure 18 (T5, T6, T7, T9, T17)
Configuration / setup	Device-Tester with BLOB-Profile support
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Testing the BLOB_Start command in Device BLOB state machine states 1 to 3. In these states, the BLOB_Start command is not allowed.
Precondition	Master port set to IOL mode. No BLOB transmission active (State of Device BLOB state machine is Idle_0).
Procedure	a) Write BLOB_Start via BLOB_CH, use BLOB_ID DTR1_ID b) Read BLOB_ID c) Write BLOB_Start via BLOB_CH, use BLOB_ID DTR1_ID NOTE 1 d) Read BLOB_ID e) Write BLOB_Start via BLOB_CH, use BLOB_ID DTR1_ID f) Read BLOB_CH g) Write BLOB_Start via BLOB_CH, use BLOB_ID DTR1_ID NOTE 2 h) Read BLOB_ID i) Write BLOB_Start via BLOB_CH, use BLOB_ID DTR1_ID j) Read BLOB_CH k) Read BLOB_CH several times until Device responds with BLOB_LAST l) Write BLOB_Start via BLOB_CH, use BLOB_ID DTR1_ID NOTE 3 m) Read BLOB_ID
Input parameter	DTR1_ID
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	Check Device response of each step-in procedure: a) Positive ISDU write response b) Positive ISDU read response with value id1 c) ISDU error response with error code 0x8022 – <i>Function temporarily unavailable</i> (according to 6.5.3.8 and T17 in Figure 18) d) Positive ISDU read response with value "0" e) Positive ISDU write response f) Positive ISDU response providing BLOB_Info_Read g) ISDU error response with error code 0x8022 – <i>Function temporarily unavailable</i> (according to 6.5.3.8 and T17 in Figure 18) h) Positive ISDU read response with value "0" i) Positive ISDU write response j) Positive ISDU response providing BLOB_Info_Read k) Positive ISDU response with BLOB_SEGMENT or BLOB_LAST l) ISDU error response with error code 0x8022 – <i>Function temporarily unavailable</i> (according to 6.5.3.8 and T17 in Figure 18) m) Positive ISDU read response with value "0"
Test passed	All steps of evaluation return expected result. IOL communication is not interrupted while test.
Test failed (examples)	Any of the evaluation steps fails.
Results	

TEST CASE RESULTS	CHECK / REACTION
NOTE 1	Illegal start command in state: "WaitOn_Read_BLOB_CH_1"
NOTE 2	Illegal start command in state: "SupplySegment_2"
NOTE 3	Illegal start command in state: "SupplyCRC_3"

1447

E.2.10 Invalid BLOB_Start during BLOB write

Table E.12 defines the test conditions for this test case.

Table E.12 – Invalid BLOB_Start during BLOB write

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE	
Identification (ID)	BTFU_TC_0009	
Name	TCD_FWDL_BLOB_Start_CMD_STATE_INADMISSIBLE_DL	
Purpose (short)	Test of BLOB_Start command in inadmissible operating states for BLOB write - error result	
Equipment under test (EUT)	Device with support for BLOB transfer (profile characteristic 0x0030)	
Test case version	1.1	
Category / type	Device BLOB state machine test.	
Occurrence	Conditional BLOB write (at least one BLOB_ID > 0 supported)	
Specification (clause)	[1] see 6.5.2, 6.5.3.2, 6.5.3.3, 6.5.3.5, 6.5.3.6, 6.5.3.8, 6.6.1 Figure 18 (T5, T11, T12, T13, T14, T17)	
Configuration / setup	Device-Tester with BLOB-Profile support	
TEST CASE	CONDITIONS / PERFORMANCE	
Purpose (detailed)	Testing the BLOB_Start command in Device BLOB state machine states 1 and 4 to 6. In these states, the BLOB_Start command is not allowed.	
Precondition	Master port set to IOL mode. No BLOB transmission active (State of Device BLOB state machine is Idle_0).	
Procedure	a) Write BLOB_Start via BLOB_CH, use BLOB_ID DTW1_ID b) Read BLOB_ID c) Write BLOB_Start via BLOB_CH, use BLOB_ID DTW1_ID d) Read BLOB_ID e) Write BLOB_Start via BLOB_CH, use BLOB_ID DTW1_ID f) Read BLOB_CH g) Write BLOB_Start via BLOB_CH, use BLOB_ID DTW1_ID h) Read BLOB_ID i) Write BLOB_Start via BLOB_CH, use BLOB_ID DTW1_ID j) Read BLOB_CH k) Write all segments using BLOB_SEGMENT via BLOB_CH l) Write the last segment using BLOB_LAST via BLOB_CH m) Write BLOB_Start via BLOB_CH, use BLOB_ID DTW1_ID n) Read BLOB_ID o) Write BLOB_Start via BLOB_CH, use BLOB_ID DTW1_ID p) Read BLOB_CH q) Write all segments using BLOB_SEGMENT via BLOB_CH r) Write the last segment using BLOB_LAST via BLOB_CH s) Write CRC using BLOB_CRC via BLOB_CH t) Write BLOB_Start via BLOB_CH, use BLOB_ID DTW1_ID u) Read BLOB_ID	
Input parameter	DTW1_ID	
Post condition	–	
TEST CASE RESULTS	CHECK / REACTION	
Evaluation	Check Device response of each step-in procedure: a) Positive ISDU write response b) Positive ISDU read response with value id1 c) ISDU error 0x8022 – <i>Function temporarily unavailable</i> (according to 6.5.3.8 and T17 in Figure 18) d) Positive ISDU read response with value "0" e) Positive ISDU write response f) Positive ISDU read response with BLOB_Info_Write g) ISDU error 0x8022 – <i>Function temporarily unavailable</i> (according to 6.5.3.8 and T17 in Figure 18) h) Positive ISDU read response with value "0" i) Positive ISDU write response j) Positive ISDU read response with BLOB_Info_Write k) Positive ISDU write response l) Positive ISDU write response	

NOTE 1

NOTE 2

NOTE 3

NOTE 4

TEST CASE RESULTS	CHECK / REACTION
	m) ISDU error 0x8022 – <i>Function temporarily unavailable</i> (according to 6.5.3.8 and T17 in Figure 18) n) Positive ISDU read response with value "0" o) Positive ISDU write response p) Positive ISDU read response with BLOB_Info_Write q) Positive ISDU write response r) Positive ISDU write response s) Positive ISDU write response t) ISDU error 0x8022 – <i>Function temporarily unavailable</i> (according to 6.5.3.8 and T17 in Figure 18) u) Positive ISDU read response with value "0"
Test passed	All steps of evaluation return expected result. IOL communication is not interrupted while test.
Test failed (examples)	Any of the evaluation steps fails.
Results	
NOTE 1 Illegal start command in state: "WaitOn_Read_BLOB_CH_1" NOTE 2 Illegal start command in state: "ReveiveSegment_4" NOTE 3 Illegal start command in state: "WaitOnCRC_5" NOTE 4 Illegal start command in state: "WaitOn_BLOB_complete_6"	

E.2.11 BLOB Abort during BLOB read

Table E.13 defines the test conditions for this test case.

Table E.13 – BLOB Abort during BLOB read

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0010
Name	TCD_FWDL_BLOB_Abort_CMD_UL
Purpose (short)	Test of BLOB Abort command in BLOB read states - positive result
Equipment under test (EUT)	Device with support for BLOB transfer (profile characteristic 0x0030)
Test case version	1.0
Category / type	Device BLOB state machine test.
Occurrence	Conditional BLOB read (at least one BLOB_ID < 0 supported)
Specification (clause)	[1] see 6.5.2, 6.5.3.1, 6.5.3.4, 6.5.3.5, 6.5.3.7, 6.5.3.8, 6.6.1 Figure 18 (T5, T6, T7, T9, T17)
Configuration / setup	Device-Tester with BLOB-Profile support
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Test of BLOB_Abort command in all read states. The write request will lead to a positive acknowledgement. Tested States: - "WaitOn_Read_BLOB_CH_1" - "SupplySegment_2" - "SupplyCRC_3"
Precondition	Master port set to IOL mode. No BLOB transmission active (State of Device BLOB state machine is Idle_0).
Procedure	a) Write BLOB_Start via BLOB_CH, use BLOB_ID DTR1_ID b) Write BLOB_Abort via BLOB_CH c) Read BLOB_ID d) Write BLOB_Start via BLOB_CH, use BLOB_ID id1 e) Read BLOB_CH f) Write BLOB_Abort via BLOB_CH g) Read BLOB_ID h) Write BLOB_Start via BLOB_CH, use BLOB_ID id1 i) Read BLOB_CH j) Read all segments including last segment via BLOB_CH k) Write BLOB_Abort via BLOB_CH l) Read BLOB_ID
Input parameter	DTR1_ID
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	Check Device response of each step-in procedure: a) Positive ISDU write response b) Positive ISDU write response c) Positive ISDU read response with value "0" d) Positive ISDU write response e) Positive ISDU read response providing BLOB_Info_Read f) Positive ISDU write response g) Positive ISDU read response with value "0" h) Positive ISDU write response i) Positive ISDU read response providing BLOB_Info_Read j) Positive ISDU read response providing BLOB segments k) Positive ISDU write response l) Positive ISDU read response with value "0"
Test passed	All steps of evaluation return expected result. IOL communication is not interrupted while test.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.2.12 BLOB_Abort during BLOB write

Table E.14 defines the test conditions for this test case.

Table E.14 – BLOB_Abort during BLOB write

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0011
Name	TCD_FWDL_BLOB_Abort_CMD_DL
Purpose (short)	Test of BLOB_Abort command in BLOB write states - positive result
Equipment under test (EUT)	Device with support for BLOB transfer (profile characteristic 0x0030)
Test case version	1.0
Category / type	Device BLOB state machine test.
Occurrence	Conditional BLOB write (at least one BLOB_ID > 0 supported)
Specification (clause)	[1] see 6.5.2, 6.5.3.2, 6.5.3.3, 6.5.3.5, 6.5.3.6, 6.5.3.8, 6.6.1, Figure 18 (T5, T11, T12, T13, T14, T17)
Configuration / setup	Device-Tester with BLOB-Profile support
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Test of BLOB_Abort command in all write states. The write request will lead to a positive acknowledgement. Tested States: - "WaitOn_Read_BLOB_CH_1" - "ReceiveSegment_4" - "WaitOnCRC_5" - "WaitOn_BLOB_complete_6"
Precondition	Master port set to IOL mode. No BLOB transmission active (State of Device BLOB state machine is Idle_0).
Procedure	a) Write BLOB_Start via BLOB_CH, use BLOB_ID DTR1_ID b) Write BLOB_Abort via BLOB_CH c) Read BLOB_ID d) Write BLOB_Start via BLOB_CH, use BLOB_ID id1 e) Read BLOB_CH f) Write BLOB_Abort via BLOB_CH g) Read BLOB_ID h) Write BLOB_Start via BLOB_CH, use BLOB_ID id1 i) Read BLOB_CH j) Write all BLOB segments including last segment via BLOB_CH k) Write BLOB_Abort via BLOB_CH l) Read BLOB_ID m) Write BLOB_Start via BLOB_CH, use BLOB_ID id1 n) Read BLOB_CH o) Write all BLOB segments including last segment via BLOB_CH p) Write valid CRC via BLOB_CH q) Write BLOB_Abort via BLOB_CH r) Read BLOB_ID
Input parameter	DTW1_ID
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	Check Device response of each step-in procedure: a) Positive ISDU write response b) Positive ISDU write response c) Positive ISDU read response with value "0" d) Positive ISDU write response e) Positive ISDU read response providing BLOB_Info_Write f) Positive ISDU write response g) Positive ISDU read response with value "0" h) Positive ISDU write response i) Positive ISDU read response providing BLOB_Info_Write j) Positive ISDU write response to all received segments k) Positive ISDU write response l) Positive ISDU read response with value "0" m) Positive ISDU write response

TEST CASE RESULTS	CHECK / REACTION
	n) Positive ISDU read response providing BLOB_Info_Write o) Positive ISDU write response to all received segments p) Positive ISDU write response q) Positive ISDU write response r) Positive ISDU read response with value "0"
Test passed	All steps of evaluation return expected result. IOL communication is not interrupted while test.
Test failed (examples)	Describe reaction and describe the reasons for failing
Results	Any of the evaluation steps fails.

E.2.13 Inadmissible write to BLOB_CH during BLOB read

Table E.15 defines the test conditions for this test case.

Table E.15 – Inadmissible write to BLOB_CH during BLOB read

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0012
Name	TCD_FWDL_BLOB_WRITE_BLOB_CH_STATE_UL
Purpose (short)	Test handling of inadmissible write access to BLOB_CH in BLOB read states
Equipment under test (EUT)	Device with support for BLOB transfer (profile characteristic 0x0030)
Test case version	1.0
Category / type	Device BLOB state machine test, test to fail (negative testing)
Occurrence	Conditional BLOB read (at least one BLOB_ID < 0 supported)
Specification (clause)	[1] see 6.5.2, 6.5.3.1, 6.5.3.4, 6.5.3.5, 6.5.3.6, 6.5.3.7, 6.5.3.8, 6.6.1, Figure 18 (T5, T6, T7, T9, T18)
Configuration / setup	Device-Tester with BLOB-Profile support
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	In all read states BLOB expects only read accesses on BLOB_CH. If a host writes to BLOB_CH the Device shall signal an error and go to state "Idle_0". Tested States: - "WaitOn_Read_BLOB_CH_1" - "SupplySegment_2" - "SupplyCRC_3"
Precondition	Master port set to IOL mode. No BLOB transmission active (State of Device BLOB state machine is Idle_0).
Procedure	a) Write BLOB_Start via BLOB_CH, use BLOB_ID id1 b) Write BLOB_CH with BLOB_Info_Read c) Read BLOB_ID d) Write BLOB_Start via BLOB_CH, use BLOB_ID id1 e) Read BLOB_CH f) Write BLOB_CH with previously read content g) Read BLOB_ID h) Write BLOB_Start via BLOB_CH, use BLOB_ID id1 i) Read BLOB_CH j) Read BLOB_CH (read all BLOB segments, including BLOB_LAST) k) Write BLOB_CH with BLOB_CRC l) Read BLOB_ID
Input parameter	DTR1_ID
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	Check Device response of each step-in procedure: a) Positive ISDU write response b) ISDU error 0x8030 – <i>Parameter value out of range</i> (T18 of Device BLOB state machine) c) Positive ISDU read response with value "0" d) Positive ISDU write response e) Positive ISDU read response providing BLOB_Info_Read f) ISDU error 0x8030 – <i>Parameter value out of range</i> (T18 of Device BLOB state machine) g) Positive ISDU read response with value "0" h) Positive ISDU write response i) Positive ISDU read response providing BLOB_Info_Read j) Positive ISDU read responses (for BLOB segments) k) ISDU error 0x8030 – <i>Parameter value out of range</i> (T18 of Device BLOB state machine) l) Positive ISDU read response with value "0"
Test passed	All steps of evaluation return expected result. IOL communication is not interrupted while test.
Test failed (examples)	Any of the evaluation steps fails.

TEST CASE RESULTS	CHECK / REACTION
Results	

1470

E.2.14 Inadmissible read to BLOB_CH during BLOB write

Table E.16 defines the test conditions for this test case.

Table E.16 – Inadmissible read to BLOB_CH during BLOB write

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0013
Name	TCD_FWDL_BLOB_READ_BLOB_CH_STATE_DL
Purpose (short)	Test handling of inadmissible read access to BLOB_CH in BLOB write states
Equipment under test (EUT)	Device with support for BLOB transfer (profile characteristic 0x0030)
Test case version	1.0
Category / type	Device BLOB state machine test, test to fail (negative testing)
Occurrence	Conditional BLOB write (at least one BLOB_ID > 0 supported)
Specification (clause)	[1] see 6.5.2, 6.5.3.2, 6.5.3.3, 6.5.3.5, 6.5.3.6, 6.5.3.8, 6.5.3.9, 6.6.1 Figure 18 (T5, T11, T12, T13, T14, T18)
Configuration / setup	Device-Tester with BLOB-Profile support
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	In all write states BLOB expects only write accesses on BLOB_CH. If a host reads BLOB_CH, the Device shall signal an error and go to state "Idle_0". Tested States: - "WaitOn_Read_BLOB_CH_1" - "ReceiveSegment_4" - "WaitOnCRC_5" - "WaitOn_BLOB_complete_6"
Precondition	Master port set to IOL mode. No BLOB transmission active (State of Device BLOB state machine is Idle_0).
Procedure	a) Write BLOB_Start via BLOB_CH, use BLOB_ID id1 b) Write BLOB_CH with BLOB_Info_Read c) Read BLOB_ID d) Write BLOB_Start via BLOB_CH, use BLOB_ID id1 e) Read BLOB_CH f) Read BLOB_CH g) Read BLOB_ID h) Write BLOB_Start via BLOB_CH, use BLOB_ID id1 i) Read BLOB_CH j) Write to BLOB_CH (all segments, including BLOB_LAST) k) Read BLOB_CH l) Read BLOB_ID m) Write BLOB_Start via BLOB_CH, use BLOB_ID id1 n) Read BLOB_CH o) Write to BLOB_CH (all segments, including BLOB_LAST) p) Write to BLOB_CH (correct BLOB_CRC) q) Read BLOB_CH r) Read BLOB_ID
Input parameter	DTW1_ID
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	Check Device response of each step-in procedure: a) Positive ISDU write response b) ISDU error 0x8030 – <i>Parameter value out of range</i> (T18 of Device BLOB state machine) c) Positive ISDU read response with value "0" d) Positive ISDU write response e) Positive ISDU read response providing BLOB_Info_Write f) ISDU error 0x8030 – <i>Parameter value out of range</i> (T18 of Device BLOB state machine) g) Positive ISDU read response with value "0" h) Positive ISDU write response i) Positive ISDU read response providing BLOB_Info_Write j) Positive ISDU write response (for all BLOB segments) k) ISDU error 0x8030 – <i>Parameter value out of range</i> (T18 of Device BLOB state machine)

TEST CASE RESULTS	CHECK / REACTION
	machine) l) Positive ISDU read response with value "0" m) Positive ISDU write response n) Positive ISDU read response providing BLOB_Info_Write o) Positive ISDU write response (for all BLOB segments) p) Positive ISDU write response q) ISDU error 0x8030 – <i>Parameter value out of range</i> (T18 of Device BLOB state machine) r) Positive ISDU read response with value "0"
Test passed	All steps of evaluation return expected result. IOL communication is not interrupted while test.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.2.15 Write BLOB_Segment at WaitOnCRC and WaitOn_BLOB_complete

Table E.17 defines the test conditions for this test case.

Table E.17 – Write BLOB_Segment at WaitOnCRC and WaitOn_BLOB_complete

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0014
Name	TCD_FWDL_BLOB_WRITE_SEGMENT_STATE_WOCRC_WOBC
Purpose (short)	Test the write BLOB_Segment handling in the unauthorized state WaitOnCRC and WaitOn_BLOB_complete.
Equipment under test (EUT)	Device with support for BLOB transfer (profile characteristic 0x0030)
Test case version	1.0
Category / type	Device BLOB state machine test, test to fail (negative testing)
Occurrence	Conditional BLOB write (at least one BLOB_ID > 0 supported)
Specification (clause)	[1] see 6.5.2, 6.5.3.2, 6.5.3.3, 6.5.3.5, 6.5.3.6, 6.5.3.8, 6.6.1 Figure 18 (T5, T11, T12, T13, T14, T18)
Configuration / setup	Device-Tester with BLOB-Profile support
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Test of write BLOB_Segment in state "WaitOnCRC_5" and "WaitOn_BLOB_complete_6". The write request will lead to a negative acknowledgement.
Precondition	Master port set to IOL mode. No BLOB transmission active (State of Device BLOB state machine is Idle_0).
Procedure	a) Write BLOB_Start via BLOB_CH, use BLOB_ID id1 b) Read BLOB_CH c) Write to BLOB_CH (all segments, including BLOB_LAST) d) Write BLOB_CH with BLOB_Segment (any data) e) Read BLOB_ID f) Write BLOB_Start via BLOB_CH, use BLOB_ID id1 g) Read BLOB_CH h) Write to BLOB_CH (all segments, including BLOB_LAST) i) Write to BLOB_CH (correct BLOB_CRC) j) Write BLOB_CH with BLOB_Segment (any data) k) Read BLOB_ID
Input parameter	DTW1_ID
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	Check Device response of each step-in procedure: a) Positive ISDU write response b) Positive ISDU read response providing BLOB_Info_Write c) Positive ISDU write response (for all BLOB segments) d) ISDU error 0x8030 – <i>Parameter value out of range</i> (T18 of Device BLOB state machine) e) Positive ISDU read response with value "0" f) Positive ISDU write response g) Positive ISDU read response providing BLOB_Info_Write h) Positive ISDU write response (for all BLOB segments) i) Positive ISDU write response j) ISDU error 0x8030 – <i>Parameter value out of range</i> (T18 of Device BLOB state machine) k) Positive ISDU read response with value "0"
Test passed	All steps of evaluation return expected result. IOL communication is not interrupted while test.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.2.16 BFlowCtrl overflow during BLOB write

Table E.18 defines the test conditions for this test case.

Table E.18 – BFlowCtrl overflow during BLOB write

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0015
Name	TCD_FWDL_BLOB_WRITE_SEGMENT_OK_STATE_RS
Purpose (short)	Test the BLOB write handling with BFlowCtrl ok and receive more than expected segments.
Equipment under test (EUT)	Device with support for BLOB transfer (profile characteristic 0x0030)
Test case version	1.1
Category / type	Device BLOB state machine test, test to fail (negative testing)
Occurrence	Conditional BLOB write (at least one BLOB_ID > 0 supported)
Specification (clause)	[1] see 6.5.2, 6.5.3.2, 6.5.3.3, 6.5.3.5, 6.5.3.8, 6.5.3.9, 6.6.1, Figure 18 (T5, T11, T12, T13, T18)
Configuration / setup	Device-Tester with BLOB-Profile support
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Test of BLOB write with more than the expected segments (overflow). The write request will lead to a negative acknowledgement.
Precondition	Master port set to IOL mode. No BLOB transmission active (State of Device BLOB state machine is Idle_0).
Procedure	a) Write BLOB_Start via BLOB_CH, use BLOB_ID id1 b) Read BLOB_CH -> store max BLOB size to temporary variable blob_size and store maximum ISDU size to max_isdu_size c) Write to BLOB_CH: write n BLOB segments with segment length = max_isdu_size and n = (int)(blob_size/(max_isdu_size-1)). Use correct flow control. d) Write to BLOB_CH: write BLOB_Last with max_isdu_size bytes. This will exceed the maximum blob size. This step is conditional and only executed, if previous step was followed by positive ISDU response. e) Read BLOB_ID
Input parameter	DTW3_ID
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) Check if Device responds with positive ISDU write response to step a) of procedure. b) Check if Device responds with positive ISDU read response (BLOB_Info_Write) to step b) of procedure c) Check if Device responds with positive ISDU write response to the first n-1 segments received in step c) of procedure. d) Check Device response to the n-th segment in step c) of procedure. Allowed response is either positive ISDU write response. Or, in case n*(max_isdu_size-1) = blob_size, a negative ISDU response with error code 0x8030 – <i>Parameter value out of range</i> because the Device can already know, that BLOB_Last will exceed the maximum BLOB size. e) If step d) of procedure was executed: Check if the Device response is negative ISDU response with error code 0x8030 – <i>Parameter value out of range</i> f) Check if Device response to step e) of procedure is positive ISDU read response with value 0.
Test passed	All steps of evaluation return expected result. IOL communication is not interrupted while test.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.2.17 BFlowCtrl nok during BLOB write

Table E.19 defines the test conditions for this test case.

Table E.19 – BFlowCtrl nok during BLOB write

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0016
Name	TCD_FWDL_BLOB_WRITE_SEGMENT_NOK_STATE_RS
Purpose (short)	Test the BLOB write handling with BFlowCtrl nok - error result
Equipment under test (EUT)	Device with support for BLOB transfer (profile characteristic 0x0030)
Test case version	1.0
Category / type	Device BLOB state machine test, test to fail (negative testing)
Occurrence	Conditional BLOB write (at least one BLOB_ID > 0 supported)
Specification (clause)	[1] see 6.5.2, 6.5.3.1, 6.5.3.4, 6.5.3.5, 6.5.3.7, 6.5.3.8, 6.6.1 Figure 18 (T5, T11, T12, T13, T18)
Configuration / setup	Device-Tester with BLOB-Profile support
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Test of write segments in wrong sequence (failure in flow control). The write request will lead to a negative acknowledgement. Combined same flow_ctrl and skipped flow_ctrl counter.
Precondition	Master port set to IOL mode. No BLOB transmission active (State of Device BLOB state machine is Idle_0).
Procedure	a) Write BLOB_Start via BLOB_CH, use BLOB_ID DTW3_ID b) Read BLOB_CH c) Write to BLOB_CH: BLOB segment with flow control = 0x00 (correct) d) Write to BLOB_CH: BLOB segment with flow control = 0x01 (correct) e) Write to BLOB_CH: BLOB segment with flow control = 0x02 (correct) f) Write to BLOB_CH: BLOB segment with flow control = 0x04 (incorrect) g) Read BLOB_ID h) Write BLOB_Start via BLOB_CH, use BLOB_ID DTW3_ID i) Read BLOB_CH j) Write to BLOB_CH: BLOB segment with flow control = 0x00 (correct) k) Write to BLOB_CH: BLOB segment with flow control = 0x01 (correct) l) Write to BLOB_CH: BLOB segment with flow control = 0x02 (correct) m) Write to BLOB_CH: BLOB segment with flow control = 0x03 (correct) n) Write to BLOB_CH: BLOB segment with flow control = 0x04 (correct) o) Write to BLOB_CH: BLOB segment with flow control = 0x04 (incorrect) p) Read BLOB_ID
Input parameter	DTW3_ID
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	Check Device response of each step-in procedure: a) Positive ISDU write response b) Positive ISDU read response providing BLOB_Info_Write c) Positive ISDU write response d) Positive ISDU write response e) Positive ISDU write response f) ISDU error 0x8030 – <i>Parameter value out of range</i> g) Positive ISDU read response with value 0 h) Positive ISDU write response i) Positive ISDU read response providing BLOB_Info_Write j) Positive ISDU write response k) Positive ISDU write response l) Positive ISDU write response m) Positive ISDU write response n) Positive ISDU write response o) ISDU error 0x8030 – <i>Parameter value out of range</i> p) Positive ISDU read response with value 0
Test passed	All steps of evaluation return expected result. IOL communication is not interrupted while test.

TEST CASE RESULTS	CHECK / REACTION
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.2.18 Profile specific IODD content

Table E.20 describes the test conditions for this test case.

Table E.20 – Profile specific IODD content

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0017
Name	TCD_FWDL_BLOB_CHECK_IODD
Purpose (short)	Test of profile specific IODD content.
Equipment under test (EUT)	IODD with BLOB transfer specific content (profile characteristic 0x0030)
Test case version	1.1
Category / type	BLOB IODD test
Occurrence	Conditional (IODD available)
Specification (clause)	[1] A.2
Configuration / setup	IODD Checker
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Test of profile specific IODD content. Existence of defined attributes and consistence check of the values
Precondition	IODD loaded and Features/@profileCharacteristic is present and contains the value "48".
Procedure	a) Identify VariableCollection/Variable[@id="V_BT_BLOBID" b) Read VariableCollection/Variable[@id="V_BT_BLOBID"]/@index c) Read VariableCollection/Variable[@id="V_BT_BLOBID"]/@accessRights d) Read VariableCollection/Variable[@id="V_BT_BLOBID"]/Datatype/@xsi:type e) Read VariableCollection/Variable[@id="V_BT_BLOBID"]/Datatype/@bitLength f) Read all VariableCollection/Variable[@id = "V_BT_BLOBID"]/Datatype / SingleValue/@value g) Read all VariableCollection/Variable[@id = "V_BT_BLOBID"]/Datatype/ SingleValue/Name/@textId h) Identify Read ExternalTextCollection/PrimaryLanguage/Text/@id = "TN_V_BT_BLOBID" i) Read ExternalTextCollection/PrimaryLanguage/Text[@id="TN_V_BT_BLOBID"]/@value j) Identify Read ExternalTextCollection/PrimaryLanguage/Text/@id = "TN_SV_BT_BLOBID_idle" k) Read ExternalTextCollection/PrimaryLanguage/Text[@id = "TN_SV_BT_BLOBID_idle"]/@value l) Look VariableCollection/UserInterface/MenuCollection/Menu/VariableRef/@variableId = "V_BT_BLOBID" m) Look for VariableCollection/Variable/@index = "50" n) Read Features/@profileCharacteristic
Input parameter	–
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	Check of each step-in procedure: a) check if attribute is existing b) index = "49" c) accessRights = "ro" d) xsi:type = "IntegerT" e) bitLength = "16" f) 1. for all: "-8191" ≤ value ≤ "8191" 2. one value = "0" 3. no value = "1" g) textId = "TN_SV_BT_BLOBID_idle" (for BLOB_ID = "0"); textID starts with "TN_SV_BT_BLOBID_" (for all other BLOB_IDs) h) check if attribute is existing i) value = "ID of BLOB that is currently transferred" j) check if attribute is existing k) value = "Idle, no BLOB transfer active" l) check if variableId is not existing (Test to fail)

NOTE

TEST CASE RESULTS	CHECK / REACTION
	m) check if index is not existing (Test to fail) n) check if string contains value "16384"
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
Results	
NOTE BLOB_ID 1 is reserved for firmware download. IODD is not present in bootload mode.	

E.2.19 Reserved indices

Table E.21 defines the test conditions for this test case.

Table E.21 – Reserved indices

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0018
Name	TCD_FWDL_CHECK_IODD
Purpose (short)	Test for using reserved indices
Equipment under test (EUT)	Every IODD <i>without</i> BLOB transfer specific content (profile characteristic not 0x0030).
Test case version	1.0
Category / type	IODD test
Occurrence	Mandatory for all IODDs
Specification (clause)	[1] A.2
Configuration / setup	IODD Checker
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	The indices BLOB_ID 0x0031 (49) and BLOB_CH 0x0032 (50) are reserved for the BLOB profile and shall not be set in all other IODDs
Precondition	IODD loaded
Procedure	a) Look for VariableCollection/Variable/@index="49" b) Look for VariableCollection/Variable/@index="50"
Input parameter	–
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	Check of each step-in procedure: a) check if index is not existing (Test to fail) b) check if index is not existing (Test to fail)
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.3 Host implementation of BLOB transfer

E.3.1 Reserved BLOB_IDs for host application test purposes

The FWBD (see E.6.2) supports the BLOB_IDs, which are used for testing the BLOB host application, for example function blocks for programmable logic controllers (PLC).

Table E.22 defines the test BLOB IDs for testing the host application.

Table E.22 – Reserved BLOB_IDs for host application test purposes

BLOB-ID	Number	Comment
HTW_ID	4096	Host Test Write ID
HTR_ID	-4096	Host Test Read ID

A test sequence is defined as an octet stream according to the algorithm in E.6.5.

The test sequences in Table E.23 are used by different test cases.

Table E.23 – Test sequences for host application tests

Sequence Name	Length(octets)	Seed value
BS_200S	5	1
BS_200M	579	2
BS_200L	4639	3
BS_200L_M_E	16863	3
BS_200L_M_NE	1199	3
BS_200L_L_E	1400	3
BS_200L_L_NE	402	3

E.3.2 Complete BLOB write (positive Test)

Table E.24 defines the test conditions for this test case.

Table E.24 – Complete BLOB write (positive Test)

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0019
Name	TCD_HOST_BLOB_WRITE
Purpose (short)	Complete BLOB write (positive Test)
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Test of HOST BLOB Write, test to pass (positive testing)
Occurrence	Host application supports BLOB write
Specification (clause)	[1] see 6.6.2; Figure 19 (T1, T3, T7, T9, T10, T14)
Configuration / setup	Host application connected via IOL-Master to FWBD device
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Testing all steps of a correct BLOB write procedure
Precondition	An FWBD Device is connected to an IO-Link Master that is controlled by a host application supporting BLOB write commands. On the host system 3 files are available that contain in a host specific format the BLOB data corresponding to "BS_200S", "BS_200M" and "BS_200L" (see Table E.23).
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Configure host to send selected "write sequence" with "BS_200S" c) Start a BLOB Transfer to the FWBD on the host application d) FWBD checks that correct sequence is received via BLOB_ID HTW_ID. FWBD checks if host uses correct ISDU lengths. e) Repeat b) to d) with "write_sequence" "BS_200M" and "BS_200L"
Input parameter	a) "TOM_ID" = 200 b) "write sequence" = "BS_200S", "BS_200M", "BS_200L" c) "BLOB_ID" = "HTW_ID" d) IODD (for all Tests)
Post condition	FWBD displays "OK" in step d) of each and every iteration
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) FWBD displays "ACTIVE" b) – c) Blob transfer executes without any errors d) FWBD indicates "OK" after each and every iteration
Test passed	After each iteration of the procedure the check on the FWBD indicates result code "OK"
Test failed (examples)	After any iteration of the procedure the on the FWBD does not indicate result code "OK"
Results	

E.3.3 Complete BLOB read (positive Test)

Table E.25 defines the test conditions for this test case.

Table E.25 – Complete BLOB read (positive Test)

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0020
Name	TCD_HOST_BLOB_READ
Purpose (short)	Positive test of BLOB read
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Test of HOST BLOB Read, test to pass (positive testing)
Occurrence	Host application supports BLOB read
Specification (clause)	[1] see 6.6.2; Figure 19 (T4, T6, T16, T17, T18, T22, T23)
Configuration / setup	Host application connected via IOL-Master to FWBD device
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Testing all steps of a correct BLOB read procedure
Precondition	An FWBD device is connected to an IO-Link Master that is controlled by a host application supporting BLOB read commands.
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Configure host application to read BLOB via "BLOB_ID" "HTR_ID" c) Execute a BLOB Transfer d) Host application generates a file e) FWBD checks content and length of each ISDU request f) Compare generated file with read_sequence "BS_200S" g) Repeat steps b) to f) with read sequence "BS_200M" and "BS_200L"
Input parameter	a) "TOM_ID" = 201 b) "BLOB_ID" = "HTR_ID" c) "read_sequence" = "BS_200S", "BS_200M" and "BS_200L"
Post condition	FWBD indicates "OK" after every iteration
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) FWBD is properly configured and displays "ACTIVE" b) Host application correctly configured c) BLOB transfer runs without any error d) Host application has generated a new file e) FWBD indicates "OK" after every iteration f) Content of generated file equals read_sequence g) repeat checks b) to f)
Test passed	Content of generated file equals read_sequence
Test failed (examples)	FWBD does not indicate "OK" in post condition Any error shows up during test execution File compare indicates a difference
Results	

E.3.4 Unsupported BLOB_ID read

Table E.26 defines the test conditions for this test case.

Table E.26 – Unsupported BLOB_ID read

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0021
Name	TCD_HOST_BLOB_INCORRECT_RID
Purpose (short)	Test Activation of read BLOB transfer with unsupported read BLOB_ID
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Required parameters for the BLOB transmission, test to fail (negative testing)
Occurrence	Host application supports BLOB read
Specification (clause)	[1] See 6.6.2, Figure 19
Configuration / setup	Host application connected via IOL-Master to FWBD device
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Host shall be able to handle the situation when a BLOB read with an unsupported BLOB_ID for reading is started. An appropriate error message should be displayed by the host application. Checks ReadChannel_2, T4 and T5.
Precondition	An FWBD Device is connected to an IO-Link Master controlled by a host application
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Configure host application to read a BLOB via "BLOB_ID" "HTR_ID" c) Start BLOB Read Transfer by host application d) FWBD does not accept reading BLOB with wrong BLOB_ID and returns error code 0x8030 – <i>Parameter value out of range</i> to host. FWBD checks, if host does not continue with ISDU read on BLOB_CH and does not send BLOB_Abort. FWBD indicates error message "WBID" e) Host application indicates wrong BLOB_ID message to user
Input parameter	a) "TOM_ID" = 202 b) "BLOB_ID" = "HTR_ID" c) IODD
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) FWBD is properly configured and displays "ACTIVE" b) Host application is correctly configured for BLOB read c) BLOB read transfer is started by host application d) FWBD indicates "WBID" and host application receives error code from FWBD e) Host application terminates transfer and indicates error message to user
Test passed	Host indicates a message that the "BLOB-ID" = "HTR_ID" is not supported in step e)
Test failed (examples)	d) FWBD does not indicate "WBID" e) Host application does not indicate a wrong BLOB-ID message
Results	

E.3.5 Unsupported BLOB_ID write

Table E.27 defines the test conditions for this test case.

Table E.27 – Unsupported BLOB_ID write

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0022
Name	TCD_HOST_BLOB_INCORRECT_WID
Purpose (short)	Test Activation of write BLOB transfer with unsupported write BLOB_ID
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Required parameters for the BLOB transmission, test to fail (negative testing)
Occurrence	Host application supports BLOB write
Specification (clause)	[1] See 6.6.2 Figure 19 (T1, T2)
Configuration / setup	Host application connected via IOL-Master to FWBD device
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Host shall be able to handle the situation when a BLOB write with an unsupported write BLOB_ID is started. An appropriate error message shall be displayed by the host application. Checks WriteChannel_1 and transitions T1 and T2.
Precondition	An FWBD Device is connected to an IO-Link Master controlled by a host application
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Configure host application to transmit "write sequence" via BLOB_ID c) Start a BLOB Transfer to the FWBD on the host application d) FWBD does not accept BLOB write on BLOB_ID and returns error code 0x8030 – <i>Parameter value out of range</i> to host. FWBD checks that host does not continue with ISDU write on BLOB_CH and does not send BLOB_Abort. FWBD indicates error message "WBID" e) Host application indicates wrong BLOB_ID message to user
Input parameter	a) "TOM_ID"=203 b) "write_sequence" = "BS_200S" c) "BLOB_ID" = "HTW_ID"
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) FWBD is properly configured and displays "ACTIVE" b) Host application is correctly configured for BLOB write c) BLOB write transfer is started d) FWBD indicates "WBID" and host application receives error code from FWBD e) Host application terminates transfer and indicates an error message about usage of inappropriate BLOB ID
Test passed	Host indicates a message indicating that the "BLOB_ID" = "HTW_ID" is not supported in step e)
Test failed (examples)	d) FWBD does not indicate "WBID" at its display e) Host application does not indicate an error message about usage of inappropriate BLOB_ID
Results	

E.3.6 Check active BLOB transfer

Table E.28 defines the test conditions for this test case.

Table E.28 – Check active BLOB transfer

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0023
Name	TCD_HOST_BLOB_BUSY
Purpose (short)	Test that the host application checks BLOB_ID before it starts a BLOB transfer
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Required parameters for the BLOB transmission, test to fail (negative testing)
Occurrence	Host application supports BLOB read
Specification (clause)	[1] See 6.6.2, Figure 19 (T4, T5, T6, T16, T17, T18, T22, T23), 6.5.3.8 Figure 16,
Configuration / setup	Host application connected via IOL-Master to FWBD device
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	The host application shall detect that a BLOB is active and give information to the user that the BLOB transfer can currently not be started because another BLOB is active. Optionally, the host application can offer aborting the current transfer.
Precondition	An FWBD Device is connected to an IO-Link Master controlled by a host application. When starting the test, the FWBD simulates that a BLOB transfer is already running and returns a BLOB_ID different from zero. If the Host sends BLOB_Start, the Device shall return specified error code.
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Configure host application to read BLOB via "BLOB_ID" "HTR_ID" c) User starts BLOB Read Transfer (Host will read the BLOB_ID, Device simulates that a BLOB transfer is running and returns BLOB_ID! = 0) d) User acknowledges abort of active BLOB transfer (Host application will send BLOB_Abort and start the new BLOB transfer) e) Host application executes read BLOB transfer with "HTR_ID"
Input parameter	a) "TOM_ID" = 204 b) "BLOB_ID" = "HTR_ID"
Post condition	FWBD indicates "OK". BLOB transfer not executed.
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) FWBD is properly configured and displays "ACTIVE" b) Host application is correctly configured for BLOB read c) Host application indicates to user BLOB transfer with read BLOB_ID is active and asks user to acknowledge abort of pending BLOB transfer or try later. d) FWBD checks if it received the following sequence: BLOB_ID read, BLOB_Abort, BLOB_Start. Indicate "ERROR" if check fails and stops. e) FWBD checks BLOB read procedure. Indicate "OK" on success.
Test passed	Host indicates a message indicating that the BLOB transfer is currently not possible because another BLOB transfer is active. Host application is able to terminate ongoing transfer and to start a new one.
Test failed (examples)	FWBD does not indicate "OK" at its display. Host application does not indicate the correct message.
Results	

E.3.7 Error handling BLOB Read

Table E.29 defines the test conditions for this test case.

Table E.29 – Error handling BLOB Read

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0024
Name	TCD_HOST_BLOB_READ_ERROR
Purpose (short)	Test that host application terminates on BLOB Read error with proper message
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Execution of BLOB transmission, test to fail (negative testing)
Occurrence	Host application supports BLOB read
Specification (clause)	[1] see 6.6.2 Figure 19 (T4, T6, T16, T25)
Configuration / setup	Host application connected to Master. Test Device (FWBD) configured to issue an error during BLOB Read process
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Specific Test Device (FWBD) is responding with an error. Host application reacts via transition T25 in Figure 19. The host application shall terminate the BLOB transfer immediately and provide an error indication to the user
Precondition	An FWBD Device is connected to an IO-Link Master controlled by a host application.
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Configure host application to read BLOB via "BLOB_ID" "HTR_ID" c) Start BLOB Read Transfer d) FWBD sends ISDU error response with error code 0x8030 – <i>Parameter value out of range</i> when host reads the first BLOB segment. FWBD checks that host does not continue with ISDU read on BLOB_CH. FWBD indicates "BLOBERROR" if check fails and otherwise "OK". e) Host application issues error message to user and terminates BLOB transfer
Input parameter	a) "TOM_ID" = 205 b) "BLOB_ID" = "HTR_ID"
Post condition	FWBD indicates "OK". BLOB transfer not executed.
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) FWBD is properly configured and displays "ACTIVE" b) Host application is correctly configured for BLOB read c) BLOB read transfer is started by host application d) FWBD indicates "OK" e) Host application terminates transfer and indicates a proper error message to user
Test passed	e) Host indicates a message indicating that the BLOB was terminated with "error during BLOB transfer".
Test failed (examples)	d) FWBD does not indicate "OK" at its display e) Host application does not indicate the correct message to the user
Results	

E.3.8 FLOW control error handling

Table E.30 defines the test conditions for this test case.

Table E.30 – FLOW control error handling

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0025
Name	TCD_HOST_BLOB_READ_FLOWCTRL_ERROR
Purpose (short)	Test that host application detects an FLOW control error in the device properly and terminates
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Execution of BLOB transmission, test to fail (negative testing)
Occurrence	Host application supports BLOB read
Specification (clause)	[1] see 6.6.2, Figure 19 (T4, T6, T16, T17, T20, T24)
Configuration / setup	Host application connected to Master. Test Device (FWBD) configured to issue an Flow Control error during BLOB Read process
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Specific Test Device (FWBD) sends the wrong BFlowCtrl. The host application shall send the BLOB_Abort command immediately and provide an error indication to the user. Checks Read_Segments_11 and T20.
Precondition	An FWBD Device is connected to an IO-Link Master controlled by a host application.
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Configure host application to read BLOB via "BLOB_ID" "HTR_ID" c) Start BLOB read Transfer d) FWBD sends flow control 0 when sending the first and the second BLOB segment. e) Host application sends BLOB_Abort command f) FWBD checks if it receives BLOB_Abort command. If yes, it indicates "ABORT". If not, it indicates "BLOBERROR" g) Host application issues error message to user
Input parameter	a) "TOM_ID" = 206 b) "BLOB_ID" = "HTR_ID"
Post condition	FWBD indicates "ABORT", transfer not executed.
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) FWBD is properly configured and displays "ACTIVE" b) Host application is correctly configured for BLOB read c) BLOB read transfer is started by host application d) Device issues incorrect flow control e) host application detects incorrect flow control, sends abort to the device and terminates BLOB f) FWBD indicates "ABORT" g) Host application indicates error message to user
Test passed	g) Host indicates a message indicating that the BLOB was terminated with "error during BLOB transfer".
Test failed (examples)	f) FWBD does not indicate "ABORT" at its display g) Host application does not indicate the correct message to the user
Results	

E.3.9 Error handling BLOB Write

Table E.31 defines the test conditions for this test case.

Table E.31 – Error handling BLOB Write

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0026
Name	TCD_HOST_BLOB_WRITE_ERROR
Purpose (short)	Test that host application terminates on BLOB Write error with proper message
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Execution of BLOB transmission, test to fail (negative testing)
Occurrence	Host application supports BLOB write
Specification (clause)	[1] see 6.6.2, Figure 19 (T1, T3, T7, T15)
Configuration / setup	Host application connected to Master. Test Device (FWBD) configured to issue an error during BLOB Write process
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Specific Test Device (FWBD) responds with an error to Write BLOB_Segment command. The host application shall terminate the BLOB transfer immediately and provide an error indication to the user (Write Segments 6).
Precondition	An FWBD Device is connected to an IO-Link Master controlled by a host application.
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Configure host application to transmit "write sequence" via BLOB_ID c) Start BLOB write Transfer d) FWBD sends ISDU error response with error code 0x8030 – <i>Parameter value out of range</i> when host writes the first BLOB segment. FWBD checks that host does not continue with Write BLOB_Segment command FWBD indicates "BLOBERROR" if check fails and "OK" otherwise e) Host application issues error message to user and terminates BLOB transfer
Input parameter	a) "TOM_ID" = 207 b) "write_sequence" = "BS_200M" c) "BLOB_ID" = "HTW_ID"
Post condition	FWBD indicates "OK", BLOB transfer not executed.
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) FWBD is properly configured and displays "ACTIVE" b) Host application is correctly configured for BLOB write c) BLOB write transfer is started by host application d) FWBD indicates "OK" e) Host application terminates transfer and indicates error message to user
Test passed	e) Host indicates a message that the BLOB was terminated with "error during BLOB transfer".
Test failed (examples)	d) FWBD does not indicate "OK" at its display e) Host application does not indicate the correct message to the user
Results	

E.3.10 Error handling CRC read

Table E.32 defines the test conditions for this test case.

Table E.32 – Error handling CRC read

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0027
Name	TCD_HOST_BLOB_READ_CRC_ERROR
Purpose (short)	Test that host application reacts correctly if the CRC sent by the Device does not match
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Execution of BLOB transmission, test to fail (negative testing)
Occurrence	Host application supports BLOB read
Specification (clause)	[1] see 6.6.2; Figure 19 (T4, T6, T16, T17, T21, T24)
Configuration / setup	Host application connected to Master. Test Device (FWBD) configured to send an incorrect CRC.
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	If the CRC supplied to the host after the last segment has been received does not match with the CRC that is calculated by the host application, this is an indication for a transmission error. The host application shall send an abort message to the Device and issue an appropriate error message to the user.
Precondition	An FWBD Device is connected to an IO-Link Master that is controlled by a host application.
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Configure host application to read BLOB via "BLOB_ID" "HTR_ID" c) Start BLOB read Transfer d) FWBD supplies incorrect CRC ((calculated CRC) + 1) when hosts read the BLOB_CRC (D_T9) e) Host application issues error message to user (H_T24) executes a garbage collection and sends BLOB_Abort command f) FWBD checks, if it received the BLOB_Abort command. If yes, it indicates "ABORT". Otherwise it indicates "BLOBERROR".
Input parameter	a) "TOM_ID" = 208 b) "BLOB_ID" = "HTR_ID"
Post condition	FWBD indicates "ABORT", BLOB transfer not executed.
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) FWBD is properly configured and displays "ACTIVE" b) Host application is correctly configured for BLOB read c) BLOB read transfer is started by host application d) – e) Host application sends BLOB_Abort and indicates error message to user. Optionally, a repetition can be offered to the user. f) FWBD indicates "ABORT"
Test passed	e) Host indicates a message indicating that the BLOB was executed with a transmission error. f) FWBD indicates "ABORT".
Test failed (examples)	e) Host application does not indicate the correct message to the user f) FWBD does not indicate "ABORT" at its display
Results	

E.3.11 Error handling CRC write

Table E.33 defines the test conditions for this test case.

Table E.33 – Error handling CRC write

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0028
Name	TCD_HOST_BLOB_WRITE_CRC_ERROR
Purpose (short)	Test that host application reacts correctly if the Device detects an CRC mismatch
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Execution of BLOB transmission, test to fail (negative testing)
Occurrence	Host application supports BLOB write
Specification (clause)	[1] see 6.6.2; Figure 19 (T1, T3, T7, T9, T10, T11, T12)
Configuration / setup	Host application connected to Master. Test Device (FWBD) configured to treat the received CRC as incorrect.
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	After the last segment has been transferred, the FWBD has evaluated a local CRC which is compared with the CRC received from the host application. On a mismatch it indicates it to the host application (D_T15). The host application executes (H_T11) followed by a garbage collection and indicates an error message to the user. The BLOB transfer is terminated. The purpose of this test is to check the behavior of the host application, if a CRC mismatch is indicated by the Device.
Precondition	An FWBD Device is connected to an IO-Link Master controlled by a host application.
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Configure host application to transmit "write sequence" via BLOB_ID c) Start BLOB write Transfer d) Device is configured to produce always a CRC mismatch in (D_T15) which is indicated to the host application as error 0x8040 – <i>Invalid parameter set</i> . e) Host application sends BLOB_Abort command, executes a garbage collection and issues an error message to the user (H_T12). f) If Device receives the BLOB_Abort command, it indicates this with "ABORT", otherwise it indicates "BLOBERROR".
Input parameter	a) "TOM_ID" = 209 b) "write_sequence" = "BS_200S" c) "BLOB_ID" = "HTW_ID"
Post condition	FWBD indicates "ABORT", BLOB transfer not executed.
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) FWBD is properly configured and displays "ACTIVE" b) Host application is correctly configured for BLOB write c) BLOB write transfer is started by host application d) FWBD sends ISDU error response e) Host application terminates transfer and indicates error message "transmission error" to user. Optionally, a repetition can be offered to the user. f) FWBD indicates "ABORT"
Test passed	e) Host indicates a message indicating that the BLOB was executed with a transmission error. f) FWBD indicates "ABORT".
Test failed (examples)	f) FWBD does not indicate "ABORT" at its display e) Host application does not indicate the correct message to the user
Results	

E.3.12 Write segments of equal length at ISDUs of maximum size

Table E.34 defines the test conditions for this test case.

Table E.34 – Write segments of equal length at ISDUs of maximum size

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0029
Name	TCD_HOST_BLOB_EQUAL_LENGTH_MAX
Purpose (short)	Positive test of BLOB with segments of equal length where the Device accepts ISDUs of maximum size
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Execution of BLOB transmission, test to pass
Occurrence	Host application supports BLOB write
Specification (clause)	[1] see 6.6.2; Figure 19 (T1, T3, T7, T9, T10, T13, T14)
Configuration / setup	Host application connected to Master. Test Device (FWBD) configured to use maximum ISDU size (232 byte).
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Check that host application handles correctly the situation where all transferred segments have the same size.
Precondition	An FWBD Device is connected to an IO-Link Master that is controlled by a host application.
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Configure host application to transmit "write sequence" via BLOB_ID c) Start a BLOB Transfer to the FWBD on the host application d) FWBD checks the length of each received BLOB segment. Indicate result code "OK" if all segments have the length maximum ISDU size minus one. Otherwise indicate result code "BLOBERROR". e) Host application indicates transfer result to the user
Input parameter	a) "TOM_ID" = 210 b) "write_sequence" = "BS_200L_M_E" c) "BLOB_ID" = "HTW_ID"
Post condition	FWBD indicates "OK"
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) FWBD displays "ACTIVE" b) host is correctly configured c) Blob transfer executes without any errors d) FWBD indicates result code "OK" e) Host application indicates success
Test passed	d) FWBD indicates result code "OK" e) Host application does indicate success to the user
Test failed (examples)	d) FWBD does not indicate result code "OK" e) Host application does not indicate success to the user
Results	

E.3.13 Write segments of unequal length at ISDUs of maximum size

Table E.35 defines the test conditions for this test case.

Table E.35 – Write segments of unequal length at ISDUs of maximum size

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0030
Name	TCD_HOST_BLOB_UNEQUAL_LENGTH_MAX
Purpose (short)	Positive test of BLOB write with segments of unequal length where Device accepts ISDUs of maximum size
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Execution of BLOB transmission, test to pass
Occurrence	Host application supports BLOB write
Specification (clause)	[1] see 6.6.2; Figure 19 (T1, T3, T7, T9, T10, T13, T14)
Configuration / setup	Host application connected to Master. Test Device (FWBD) configured to use maximum ISDU size (232 byte).
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Check that host application handles correctly the situation where the last transferred segment has a shorter size.
Precondition	An FWBD Device is connected to an IO-Link Master controlled by a host application.
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Configure host application to transmit "write sequence" via BLOB_ID c) Start a BLOB Transfer to the FWBD on the host application d) FWBD checks the length of each received BLOB segment. Indicate result code "OK" if all segments except the last have the length maximum ISDU size minus one. Length of last segment shall be equal to remaining bytes. Otherwise indicate result code "BLOBERROR". e) Host application indicates transfer result to the user
Input parameter	a) "TOM_ID" = 211 b) "write_sequence" = "BS_200L_M_NE" c) "BLOB_ID" = "HTW_ID"
Post condition	FWBD indicates "OK"
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) FWBD displays "ACTIVE" b) host is correctly configured c) Blob transfer executes without any errors d) FWBD indicates result code "OK" e) Host application indicates success
Test passed	d) FWBD indicates result code "OK" e) Host application does indicate success to the user
Test failed (examples)	d) FWBD does not indicate result code "OK" e) Host application does not indicate success to the user
Results	

E.3.14 Write segments of equal length at ISDUs of limited size

Table E.36 defines the test conditions for this test case.

Table E.36 – Write segments of equal length at ISDUs of limited size

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0031
Name	TCD_HOST_BLOB_EQUAL_LENGTH_LIMITED
Purpose (short)	Positive test of BLOB write with segments of equal length where the Device accepts ISDUs of limited size
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Execution of BLOB transmission, test to pass
Occurrence	Host application supports BLOB write
Specification (clause)	[1] see 6.6.2; Figure 19 (T1, T3, T7, T10, T13, T14)
Configuration / setup	Host application connected to Master. Test Device (FWBD) configured to use reduced ISDU size of 15 bytes.
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Check that host application handles correctly the situation where all transferred segments have the same size.
Precondition	An FWBD Device is connected to an IO-Link Master controlled by a host application.
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Configure host application to transmit "write sequence" via BLOB_ID c) Start a BLOB Transfer to the FWBD on the host application d) FWBD checks the length of each received BLOB segment. Indicate result code "OK" if all segments have the length ISDU size minus one. Otherwise indicate result code "BLOBERROR". e) Host application indicates transfer result to the user.
Input parameter	a) "TOM_ID" = 212 b) "write_sequence" = "BS_200L_L_E" c) "BLOB_ID" = "HTW_ID"
Post condition	FWBD indicates "OK"
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) FWBD displays "ACTIVE" b) host is correctly configured c) Blob transfer executes without any errors d) FWBD indicates result code "OK" e) Host application indicates success
Test passed	d) FWBD indicates result code "OK" e) Host application does indicate success to the user
Test failed (examples)	d) FWBD does not indicate result code "OK" e) Host application does not indicate success to the user
Results	

E.3.15 Write with segments of unequal length at ISDUs of limited size

Table E.37 defines the test conditions for this test case.

Table E.37 – Write with segments of unequal length at ISDUs of limited size

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0032
Name	TCD_HOST_BLOB_UNEQUAL_LENGTH_LIMITED
Purpose (short)	Positive test of BLOB write with segments of unequal length where Device accepts ISDUs of limited size
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Execution of BLOB transmission, test to pass
Occurrence	Host application supports BLOB write
Specification (clause)	[1] see 6.6.2; Figure 19 (T1, T3, T7, T9, T10, T13, T14)
Configuration / setup	Host application connected to Master. Test Device (FWBD) configured to use reduced ISDU size of 15 bytes.
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Check that host application handles correctly the situation where the last transferred segment has shorter size and the Device does not support maximum size ISDU transfers on BLOB_CH.
Precondition	An FWBD Device is connected to an IO-Link Master controlled by a host application.
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Configure host application to transmit "write sequence" via BLOB_ID c) Start a BLOB Transfer to the FWBD on the host application d) FWBD checks the length of each received BLOB segment. Indicate result code "OK" if all segments except the last have the length ISDU size minus one. Length of last segment shall be equal to remaining bytes. Otherwise indicate result code "BLOBERROR". e) Host application indicates transfer result to the user
Input parameter	a) "TOM_ID" = 213 b) "write_sequence" = "BS_200L_L_NE" c) "BLOB_ID" = "HTW_ID"
Post condition	FWBD indicates "OK"
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) FWBD displays "ACTIVE" b) host is correctly configured c) Blob transfer executes without any errors d) FWBD indicates result code "OK" e) Host application indicates success
Test passed	d) FWBD indicates result code "OK" e) Host application does indicate success to the user
Test failed (examples)	d) FWBD does not indicate result code "OK" e) Host application does not indicate success to the user
Results	

E.3.16 BLOB size is too big for Device

Table E.38 defines the test conditions for this test case.

Table E.38 – BLOB size is too big for Device

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0033
Name	TCD_HOST_BLOB_TOO_BIG
Purpose (short)	Test that host application handles situation correctly where BLOB size is too big for Device
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Execution of BLOB transmission, test to fail
Occurrence	Host application supports BLOB write
Specification (clause)	[1] see 6.6.2; Figure 19 (T1, T3, T8, T12)
Configuration / setup	Host application connected to Master. Test Device (FWBD) configured to accept smaller BLOB size than Host will use.
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Check that host application issues a message to the user that the BLOB file to be transferred is bigger than the Device can accept and that the host application terminates the transfer.
Precondition	An FWBD Device is connected to an IO-Link Master controlled by a host application.
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Configure host application to transmit "write sequence" via BLOB_ID c) Start a BLOB Transfer to the FWBD on the host application d) FWBD returns in BLOB_Info_Write a maximum BLOB size that is smaller than the size of the "write_sequence". FWBD checks if the host sends BLOB_Abort instead of sending BLOB segments. If yes, it indicates "ABORT", otherwise it indicates "BLOBERROR". e) Host application indicates error message to the user and terminates BLOB
Input parameter	a) "TOM_ID" = 214 b) "write_sequence" = "BS_200L" c) "BLOB_ID" = "HTW_ID"
Post condition	FWBD indicates "ABORT"
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) FWBD displays "ACTIVE" b) host is correctly configured c) FWBD indicates too small Maximum ISDU size to d) Host sends BLOB_Abort command e) Host application error message to user
Test passed	d) FWBD indicates result code "ABORT" at its display e) Host application does indicate "BLOB size too big" message to user
Test failed (examples)	d) FWBD does not indicate result code "ABORT" at its display e) Host application does not indicate "BLOB size too big" message to user
Results	

E.4 Device implementation of Firmware Update

E.4.1 Test Modules

E.4.1.1 General

The modules of the test steps are used in several firmware update test cases.

E.4.1.2 Complete Unlock Sequence

Table E.39 defines the test steps for this test module.

Table E.39 – Complete Unlock Sequence

TEST CASE	CONDITIONS / PERFORMANCE
Procedure	1) Write SystemCommand BM_UNLOCK_S 0x50 2) Write SystemCommand BM_UNLOCK_F 0x51 3) Write SystemCommand BM_UNLOCK_T 0x52 4) Write SystemCommand BM_UNLOCK_F 0x51 5) Write SystemCommand BM_UNLOCK_F 0x51 6) Write SystemCommand BM_UNLOCK_F 0x51 7) Write SystemCommand BM_UNLOCK_F 0x51 8) Write SystemCommand BM_UNLOCK_T 0x52 9) Write SystemCommand BM_UNLOCK_F 0x51
TEST CASE RESULTS	CHECK / REACTION
Evaluation	1) Positive SystemCommand response 2) Positive SystemCommand response 3) Positive SystemCommand response 4) Positive SystemCommand response 5) Positive SystemCommand response 6) Positive SystemCommand response 7) Positive SystemCommand response 8) Positive SystemCommand response 9) Positive SystemCommand response
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.

E.4.1.3 Wait for BootmodeStatus = 0x01

Table E.40 defines the test steps for this test module.

Table E.40 – Wait for BootmodeStatus = 0x01

TEST CASE	CONDITIONS / PERFORMANCE
Procedure	1) loop counter = "0"; loop_max = 10 x fwActivationRetryCount 2) Read index 0x43BF and store BootmodeStatus 3) Wait 100 ms 4) Retry until BootmodeStatus = 0x01 or loop counter ≥ loop_max, increment loop counter; NOTE
TEST CASE RESULTS	CHECK / REACTION
Evaluation	1) – 2) Positive ISDU read response with value 0x00 or 0x01 3) – 4) BootmodeStatus = 0x01 and loop counter < loop_max
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
NOTE fwActivationRetryCount x 10 × 100 ms. The test is 10 times faster to measure the response time more precisely.	

E.4.1.4 Wait for BootmodeStatus = 0x00

Table E.41 defines the test steps for this test module.

Table E.41 – Wait for BootmodeStatus = 0x00

TEST CASE	CONDITIONS / PERFORMANCE
Procedure	1) loop counter = "0"; NOTE loop_max = 10 × fwActivationRetryCount 2) Read index 0x43BF and store BootmodeStatus 3) Wait 100 ms 4) Retry until BootmodeStatus = 0x01 or loop counter ≥ loop_max, increment loop counter;
TEST CASE RESULTS	CHECK / REACTION
Evaluation	1) – 2) Positive ISDU read response with value 0x00 or 0x01 3) – 4) BootmodeStatus = 0x01 and loop counter < loop_max
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
NOTE fwActivationRetryCount × 10 × 100 ms. The test is 10 times faster to measure the response time more precisely.	

E.4.1.5 Wait for BM_ACTIVATE = successful

Table E.42 defines the test steps for this test module.

Table E.42 – Wait for BM_ACTIVATE = successful

TEST CASE	CONDITIONS / PERFORMANCE
Procedure	1) loop counter = "0"; NOTE loop_max = 10 × fwActivationRetryCount 2) Write System Command BM_ACTIVATE 0x53 and store iSDU response 3) Wait 100 ms 4) Retry 2) while negative ISDU response "SERV_NOTAVAIL_DEVCTRL" loop counter < loop_max, increment loop counter;
TEST CASE RESULTS	CHECK / REACTION
Evaluation	1) – 2) Positive ISDU read response or "SERV_NOTAVAIL_DEVCTRL" 3) – 4) Positive ISDU read response and loop counter < loop_max
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
NOTE fwActivationRetryCount × 10 × 100 ms. The test is 10 times faster to measure the response time more precisely.	

E.4.2 Complete firmware update with/without password

Table E.43 defines the test conditions for this test case.

Table E.43 – Complete firmware update with/without password

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0034
Name	TCD_FWDL_BLOB_COMPLETE_FW_UPDATE
Purpose (short)	Test of a complete firmware update with/without password
Equipment under test (EUT)	Device with support of Firmware Update (profile characteristic 0x0031)
Test case version	1.1
Category / type	Device FW-Update state machine test, test to pass (positive testing)
Occurrence	mandatory FW-Update
Specification (clause)	[1] see 7.6.7.1, 7.6.9, 7.7.2; Figure 30 (T1, T2, T5, T6, T7, T8, T9, T10, T11, T12, T15, T17, T18, T19)
Configuration / setup	Device Tester with FW-Update-Profile support, valid FW-Update File
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Testing of a complete firmware update procedure. Including unlocking, switch to boot load mode, BLOB-Transfer, activation and switch back to technology application.
Precondition	Device has a valid firmware and is running in the technology application. OPERATE or PREOPERATE. State "WaitingOnSysCmd_3". BootmodeStatus = 0x00, Bootloader inactive.
Procedure	a) Read index 0x43BE (HW_ID_KEY) b) Read index 0x0016 (HardwareRevision) c) Read index 0x0017 (FirmwareRevision) d) Read index 0x43BF (BootmodeStatus) e) Read VendorID and store f) Read DeviceID and store g) Write password to index 0x43BD (FW-password) - skip if fwPasswordRequired is FALSE h) Identify Variable/[@index="17341"] - skip if fwPasswordRequired is FALSE i) Module: "Complete Unlock Sequence" j) Module: "Wait for BootmodeStatus = 0x01" k) Read VendorID l) Read DeviceID and store m) Read index 0x43BE (HW_ID_KEY) n) BLOB-Transfer o) Module: "Wait for BM_ACTIVATE = successful" p) Module: "Wait for BootmodeStatus = 0x00" q) Write SystemCommand BM_UNLOCK_S 0x50 - skip if fwPasswordRequired is FALSE r) Read VendorID s) Read DeviceID <div style="text-align: right;">NOTE</div>
Input parameter	Valid firmware data
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) Positive ISDU read response HW_ID_Key shall match HardwareIdKeyCollection b) Positive ISDU read response c) Positive ISDU read response d) Positive ISDU read response with value 0x00 e) Positive ISDU read response VendorID shall match attribute in IOLFW file f) Positive ISDU read response g) Positive ISDU write response h) check if index is existing i) Positive SystemCommand response (each SystemCommand) j) no error k) Positive ISDU read response VendorID shall match stored value (step e) l) Positive ISDU read response DeviceID shall NOT match stored value (step f), DeviceID > 0 m) Positive ISDU read response HW_ID_Key shall match HardwareIdKeyCollection n) see BLOB-Transfer TC o) no error

TEST CASE RESULTS	CHECK / REACTION
	p) no error q) SystemCommand error 0x8036 – <i>Function temporarily unavailable</i> r) Positive ISDU read response VendorID shall match stored value (step e) s) Positive ISDU read response DeviceID shall not match stored value (step l)
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
Results	
NOTE Check if password flag is reset	

E.4.3 Unlock with a wrong password

Table E.44 defines the test conditions for this test case.

Table E.44 – Unlock with a wrong password

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0035
Name	TCD_FWDL_BLOB_INCORRECT_PASSWORD
Purpose (short)	Test unlock with a wrong password
Equipment under test (EUT)	Device with support of Firmware Update (profile characteristic 0x0031)
Test case version	1.1
Category / type	Device FW-Update state machine test, test to fail (negative test)
Occurrence	Conditional FW-Update (fwPasswordRequired is TRUE)
Specification (clause)	[1] see 7.6.7.1, 7.6.9, 7.7.2, Figure 30 (T1, T21)
Configuration / setup	Device Tester with FW-Update-Profile support, valid FW-Update File
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Testing of the protection against a firmware update with an illegal or missing password. The Device shall acknowledge the incorrect FW-password with an error message and reject the unlock sequence if the internal FW-PasswordFlag is FALSE.
Precondition	Device is running in the technology application. OPERATE or PREOPERATE. State "WaitingOnSysCmd_3". BootmodeStatus = 0x00, Bootloader inactive.
Procedure	a) Write SystemCommand BM_UNLOCK_S 0x50 b) Read BootmodeStatus 0x43BF c) Write password to index 0x43BD (FW-password = "#?\${@}") d) Read BootmodeStatus 0x43BF e) Write SystemCommand BM_UNLOCK_S 0x50 f) Read BootmodeStatus 0x43BF
Input parameter	–
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) SystemCommand error 0x8036 – <i>Function temporarily unavailable</i> b) Positive ISDU read response with value 0x00 c) ISDU error 0x8030 – <i>Parameter value out of range</i> d) Positive ISDU read response with value 0x00 e) SystemCommand error 0x8036 – <i>Function temporarily unavailable</i> f) Positive ISDU read response with value 0x00
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.4.4 Inadmissible BM_UNLOCK_F

Table E.45 defines the test conditions for this test case.

Table E.45 – Inadmissible BM_UNLOCK_F

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0036
Name	TCD_FWDL_BLOB_WRITE_UNLOCK_F_STATE_INADMISSIBLE
Purpose (short)	Test to write BM_UNLOCK_F SystemCommand in inadmissible operation states
Equipment under test (EUT)	Device with support of Firmware Update (profile characteristic 0x0031)
Test case version	1.1
Category / type	Device FW-Update state machine test, test to fail (negative test)
Occurrence	mandatory FW-Update
Specification (clause)	[1] see 7.7.2; Figure 30 (T4, T14, T16, T22)
Configuration / setup	Device Tester with FW-Update-Profile support, valid FW-Update File
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Testing SystemCommand BM_UNLOCK_F in states "WaitingOnSysCmd_3", "Unlocking_4", "FW_Update_Bootload_14", and "WaitOnActivation_15".
Precondition	Device is running in the technology application. OPERATE or PREOPERATE. State "WaitingOnSysCmd_3". BootmodeStatus = 0x00, Bootloader inactive FW-Password written, if fwPasswordRequired is TRUE.
Procedure	a) Write SystemCommand BM_UNLOCK_F 0x51 b) Write SystemCommand BM_UNLOCK_S 0x50 c) Write SystemCommand BM_UNLOCK_F 0x51 d) Write SystemCommand BM_UNLOCK_F 0x51 e) Read BootmodeStatus 0x43BF f) Module: "Complete Unlock Sequence" g) Module: "Wait for BootmodeStatus = 0x01" h) Write SystemCommand BM_UNLOCK_F 0x51 i) Read BootmodeStatus 0x43BF j) BLOB-Transfer k) Write SystemCommand BM_UNLOCK_F 0x51 l) Read BootmodeStatus 0x43BF m) Module: "Wait for BM_ACTIVATE = successful"< n) wait for BootmodeStatus = 0x00
Input parameter	Valid firmware data
Post condition	a) –
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) SystemCommand error 0x8036 – <i>Function temporarily unavailable</i> b) Positive SystemCommand response c) Positive SystemCommand response d) SystemCommand error 0x8036 – <i>Function temporarily unavailable</i> e) Positive ISDU read response with value 0x00 f) Positive SystemCommand response (each SystemCommand) g) no error h) SystemCommand error 0x8036 – <i>Function temporarily unavailable</i> i) Positive ISDU read response with value 0x01 j) see BLOB transfer k) SystemCommand error 0x8036 – <i>Function temporarily unavailable</i> l) Positive ISDU read response with value 0x01 m) no error n) no error
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.4.5 Inadmissible BM_UNLOCK_T

Table E.46 defines the test conditions for this test case.

Table E.46 – Inadmissible BM_UNLOCK_T

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0037
Name	TCD_FWDL_BLOB_WRITE_UNLOCK_T_STATE_INADMISSIBLE
Purpose (short)	Test to write BM_UNLOCK_T SystemCommand in inadmissible operation states
Equipment under test (EUT)	Device with support of Firmware Update (profile characteristic 0x0031)
Test case version	1.1
Category / type	Device FW-Update state machine test, test to fail (negative test)
Occurrence	mandatory FW-Update
Specification (clause)	[1] see 7.7.2; Figure 30 (T3, T14, T16, T22)
Configuration / setup	Device Tester with FW-Update-Profile support, valid FW-Update File
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Testing SystemCommand BM_UNLOCK_T in states "WaitingOnSysCmd_3", "Unlocking_4", "FW_Update_Bootload_14", and "WaitOnActivation_15".
Precondition	Device is running in the technology application. OPERATE or PREOPERATE. State "WaitingOnSysCmd_3". BootmodeStatus = 0x00, Bootloader inactive. FW-Password written, if fwPasswordRequired is TRUE.
Procedure	a) Write SystemCommand BM_UNLOCK_T 0x52 b) Write SystemCommand BM_UNLOCK_S 0x50 c) Write SystemCommand BM_UNLOCK_T 0x52 d) Read BootmodeStatus 0x43BF e) Module: "Complete Unlock Sequence" f) Module: "Wait for BootmodeStatus = 0x01" g) Write SystemCommand BM_UNLOCK_T 0x52 h) Read BootmodeStatus 0x43BF i) BLOB-Transfer j) Write SystemCommand BM_UNLOCK_T 0x52 k) Read BootmodeStatus 0x43BF l) Module: "Wait for BM_ACTIVATE = successful" m) Module: "Wait for BootmodeStatus = 0x00"
Input parameter	Valid firmware data
Post condition	a) –
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) SystemCommand error 0x8036 – <i>Function temporarily unavailable</i> b) Positive SystemCommand response c) SystemCommand error 0x8036 – <i>Function temporarily unavailable</i> d) Positive ISDU read response with value 0x00 e) Positive SystemCommand response (each SystemCommand) f) No error g) SystemCommand error 0x8036 – <i>Function temporarily unavailable</i> h) Positive ISDU read response with value 0x01 i) See BLOB transfer j) SystemCommand error 0x8036 – <i>Function temporarily unavailable</i> k) Positive ISDU read response with value 0x01 l) No error m) No error
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.4.6 Inadmissible BM_ACTIVATE

Table E.47 defines the test conditions for this test case.

Table E.47 – Inadmissible BM_ACTIVATE

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0038
Name	TCD_FWDL_BLOB_WRITE_ACTIVATE_STATE_INADMISSIBLE
Purpose (short)	Test to write BM_ACTIVATE SystemCommand in inadmissible operation states
Equipment under test (EUT)	Device with support of Firmware Update (profile characteristic 0x0031)
Test case version	1.1
Category / type	Device FW-Update state machine test, test to fail (negative test)
Occurrence	mandatory FW-Update
Specification (clause)	[1] see 7.7.2; Figure 30 (T23, T14, T16, T22)
Configuration / setup	Device Tester with FW-Update-Profile support, valid FW-Update File
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Testing SystemCommand SysCmd_BM_ACTIVATE in states "WaitingOnSysCmd_3", "Unlocking_4", "FW_Update_Bootload_14", and "WaitOnActivation_15".
Precondition	Device is running in the technology application. OPERATE or PREOPERATE. State "WaitingOnSysCmd_3". BootmodeStatus = 0x00, Bootloader inactive FW-Password written, if fwPasswordRequired is TRUE.
Procedure	a) Write SystemCommand BM_ACTIVATE 0x53 b) Write SystemCommand BM_UNLOCK_S 0x50 c) Write SystemCommand BM_ACTIVATE 0x53 d) Read BootmodeStatus 0x43BF e) Module: "Complete Unlock Sequence" f) Module: "Wait for BootmodeStatus = 0x01" g) Write SystemCommand BM_ACTIVATE 0x53 h) Read BootmodeStatus 0x43BF i) BLOB-Transfer j) Write SystemCommand BM_ACTIVATE 0x53 k) Module: "Wait for BootmodeStatus = 0x00"
Input parameter	Valid firmware data
Post condition	a) –
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) SystemCommand error 0x8036 – <i>Function temporarily unavailable</i> b) Positive SystemCommand response c) SystemCommand error 0x8036 – <i>Function temporarily unavailable</i> d) Positive ISDU read response with value 0x00 e) Positive SystemCommand response (each SystemCommand) f) No error g) SystemCommand error 0x8036 – <i>Function temporarily unavailable</i> h) Positive ISDU read response with value 0x01 i) See BLOB transfer j) Positive SystemCommand response k) No error
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.4.7 Inadmissible BM_UNLOCK_S and restart unlock sequence

Table E.48 defines the test conditions for this test case.

Table E.48 – Inadmissible BM_UNLOCK_S and restart unlock sequence

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0039
Name	TCD_FWDL_BLOB_WRITE_UNLOCK_S_STATE_INADMISSIBLE
Purpose (short)	Test to write BM_UNLOCK_S SystemCommand in inadmissible operation states and restart unlock sequence
Equipment under test (EUT)	Device with support of Firmware Update (profile characteristic 0x0031)
Test case version	1.1
Category / type	Device FW-Update state machine test, test to fail (negative test)
Occurrence	mandatory FW-Update
Specification (clause)	[1] see 7.7.2; Figure 30 (T13, T14, T16, T22)
Configuration / setup	Device Tester with FW-Update-Profile support, valid FW-Update File
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Testing SystemCommand BM_UNLOCK_S in states "Unlocking_4", "FW_Update_Bootload_14", and "WaitOnActivation_15". Additionally: Restart of unlock sequence
Precondition	Device is running in the technology application. OPERATE or PREOPERATE. State "WaitingOnSysCmd_3". BootmodeStatus = 0x00, Bootloader inactive. FW-Password written, if fwPasswordRequired is TRUE.
Procedure	a) Write SystemCommand BM_UNLOCK_S 0x50 b) Write SystemCommand BM_UNLOCK_S 0x50 c) Read BootmodeStatus 0x43BF d) Write SystemCommand BM_UNLOCK_S 0x50 e) Write SystemCommand BM_UNLOCK_F 0x51 f) Write SystemCommand BM_UNLOCK_T 0x52 g) Module: "Complete Unlock Sequence" h) Module: "Wait for BootmodeStatus = 0x01" i) Write SystemCommand BM_UNLOCK_S 0x50 j) Read BootmodeStatus 0x43BF k) BLOB-Transfer l) Write SystemCommand BM_UNLOCK_S 0x50 m) Read BootmodeStatus 0x43BF n) Module: "Wait for BM_ACTIVATE = successful" o) Module: "Wait for BootmodeStatus = 0x00"
Input parameter	Valid firmware data
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) Positive SystemCommand response b) Positive SystemCommand response c) Positive ISDU read response with value 0x00 d) Positive SystemCommand response e) Positive SystemCommand response f) Positive SystemCommand response g) Positive SystemCommand response (each SystemCommand) h) No error i) SystemCommand error 0x8036 – <i>Function temporarily unavailable</i> j) Positive ISDU read response with value 0x01 k) See BLOB transfer l) SystemCommand error 0x8036 – <i>Function temporarily unavailable</i> m) Positive ISDU read response with value 0x01 n) No error o) No error
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.

TEST CASE RESULTS	CHECK / REACTION
Results	

1671

E.4.8 Wrong firmware-image with & without password

Table E.49 defines the test conditions for this test case.

Table E.49 – Wrong firmware-image with & without password

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0040
Name	TCD_FWDL_BLOB_SEND_INCORRECT_FW
Purpose (short)	Test to send wrong firmware-image with & without password
Equipment under test (EUT)	Device with support of Firmware Update (profile characteristic 0x0031)
Test case version	1.1
Category / type	Device BLOB state machine test, test to fail (negative testing)
Occurrence	mandatory FW-Update
Specification (clause)	[1] see 7.6.7.1, 7.6.9, 7.7.2 Figure 30 (T1, T2, T5, T6, T7, T8, T9, T10, T11, T12, T15, T17, T18, T19, T20)
Configuration / setup	Device Tester with FW-Update-Profile support, invalid and valid FW-Update File
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Testing of a complete firmware update procedure with an invalidated firmware image. Including unlocking, switch to boot load mode, BLOB-Transfer, activation. The Device shall restart in boot load mode after detecting an invalid firmware for recovery. The complete recovery procedure is tested by downloading a valid firmware image. The Device shall restart in normal operation after the complete test.
Precondition	Device has a valid firmware and is running in the technology application. OPERATE or PREOPERATE. State "WaitingOnSysCmd_3". BootmodeStatus = 0x00, Bootloader inactive.
Procedure	a) Read index 0x43BF (BootmodeStatus) b) Write password to index 0x43BD (FW-password) - skip if fwPasswordRequired is FALSE c) Module: "Complete Unlock Sequence" d) wait for BootmodeStatus = 0x01 e) BLOB-Transfer invalid firmware f) Write SystemCommand BM_ACTIVATE g) Module: "Wait for BootmodeStatus = 0x00" h) Read index 0x43BF (BootmodeStatus) i) BLOB-Transfer valid firmware j) Module: "Wait for BM_ACTIVATE = successful" k) Module: "Wait for BootmodeStatus = 0x00"
Input parameter	Password, invalid firmware data (The content of the downloaded payload/firmware image is manufacturer dependent. Also, the encryption and authentication mechanisms. The manufacturer shall generate a special faulty firmware image for this test, which should be rejected by the Device, for example checksum or encryption violation), and valid firmware data.
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) Positive ISDU read response with value 0x00 b) Positive ISDU write response c) Positive SystemCommand response (each SystemCommand) d) No error e) See BLOB transfer f) Positive ISDU read response g) Time out h) Positive ISDU read response with value 0x01 i) See BLOB transfer j) No error k) No error
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.4.9 Random power fail or reset with & without password

Table E.50 defines the test conditions for this test case.

Table E.50 – Random power fail or reset with & without password

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0041
Name	TCD_FWDL_BLOB_RETRY_AFTER_COM_LOST
Purpose (short)	Test of the recover functionality after random power fail or reset with & without password
Equipment under test (EUT)	Device with support of Firmware Update (profile characteristic 0x0031)
Test case version	1.1
Category / type	Device BLOB state machine test, test to fail (negative testing)
Occurrence	Conditional FW-Update, Test is only needed if the device cannot keep two firmwares
Specification (clause)	[1] see 7.6.7.1, 7.6.9, 7.7.2 Figure 30 (T1, T2, T5, T6, T7, T8, T9, T10, T11, T12, T15, T17, T18, T19)
Configuration / setup	Device Tester with FW-Update-Profile support, valid FW-Update File
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Testing of a complete firmware update procedure. Including unlocking, switch to boot load mode, BLOB-Transfer, activation. The BLOB transfer is interrupted by a random (power on) reset. The Device shall restart in boot load mode, if the old firmware in the Device was overridden. The complete recovery procedure is tested by down-loading a valid firmware image. The Device shall restart in normal operation after the complete test.
Precondition	Device has a valid firmware and is running in the technology application. OPERATE or PREOPERATE. State "WaitingOnSysCmd_3". BootmodeStatus = 0x00, Bootloader inactive.
Procedure	a) Read index 0x43BF (BootmodeStatus) b) Write password to index 0x43BD (FW-password) - skip if fwPasswordRequired is FALSE c) Module: "Complete Unlock Sequence" d) Module: "Wait for BootmodeStatus = 0x01" e) Start BLOB-Transfer firmware f) Power ON Reset at a random sequence g) restart communication (wake up) h) Read index 0x43BF (BootmodeStatus) i) BLOB-Transfer valid firmware j) Module: "Wait for BM_ACTIVATE = successful" k) Module: "Wait for BootmodeStatus = 0x00"
Input parameter	Password, valid firmware data.
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) Positive ISDU read response with value 0x00 b) Positive ISDU read response c) Positive SystemCommand response (each SystemCommand) d) No error e) See BLOB transfer f) COMLOST by the master g) COMREADY by the master h) Positive ISDU read response with value 0x01 i) See BLOB transfer j) No error k) No error
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.4.10 Error cases during BLOB transfer

Table E.51 defines the test conditions for this test case.

Table E.51 – Error cases during BLOB transfer

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0042
Name	TCD_FWDL_BLOB_BLOB_ERRORS
Purpose (short)	Test of error cases during BLOB transfer
Equipment under test (EUT)	Device with support of Firmware Update (profile characteristic 0x0031)
Test case version	1.1
Category / type	Device BLOB state machine test, test to fail (negative testing)
Occurrence	mandatory
Specification (clause)	[1] see 7.6.7.1, 7.6.9, 7.7.2, Figure 30
Configuration / setup	Device Tester with FW-Update-Profile support.
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Testing of all relevant BLOB transfer error test cases
Precondition	Device is running in the boot load application. State "WaitingOnFW_Update_Bootload_14". BootmodeStatus = 0x01, Bootloader active.
Procedure	a) BTFU_TC_0003 b) BTFU_TC_0004 c) BTFU_TC_0005 d) BTFU_TC_0006 e) BTFU_TC_0007 f) BTFU_TC_0009 g) BTFU_TC_0011 h) BTFU_TC_0013 i) BTFU_TC_0014 j) BTFU_TC_0015 k) BTFU_TC_0016
Input parameter	–
Post condition	BLOB transfer valid firmware, activate, and wait for rechnology application.
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) TC passed b) TC passed c) TC passed d) TC passed e) TC passed f) TC passed g) TC passed h) TC passed i) TC passed j) TC passed k) TC passed
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.4.11 Firmware Download specific IODD content

Table E.52 defines the test conditions for this test case.

Table E.52 – Firmware Download specific IODD content

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0043
Name	TCD_FWDL_BLOB_CHECK_IODD
Purpose (short)	Test of firmware download specific IODD content
Equipment under test (EUT)	IODD with FW-Update specific content (profile characteristic 0x0031)
Test case version	1.1
Category / type	FW-Update IODD test
Occurrence	mandatory
Specification (clause)	[1] A.2
Transitions	--
Configuration / setup	IODD Checker
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Test of profile specific IODD content. Existence of defined attributes and consistence check of the values.
Precondition	IODD loaded and Features/@profileCharacteristic is present and contains the value "49".
Procedure	a) Identify VariableCollection/StdVariableRef[@id = "V_SystemCommand"]/SingleValue/@value = "80" b) Read VariableCollection/StdVariableRef[@id = "V_SystemCommand"]/SingleValue[@value = "80"]/Name/@textId c) Identify VariableCollection/StdVariableRef[@id = "V_SystemCommand"]/SingleValue/@value = "81" d) Read VariableCollection/StdVariableRef[@id = "V_SystemCommand"]/SingleValue[@value = "81"]/Name/@textId e) Identify VariableCollection/StdVariableRef[@id = "V_SystemCommand"]/SingleValue/@value = "82" f) Read VariableCollection/StdVariableRef[@id = "V_SystemCommand"]/SingleValue[@value = "82"]/Name/@textId g) Look for VariableCollection/StdVariableRef[@id = "V_SystemCommand"]/SingleValue/@value = "83" h) Identify Read ExternalTextCollection/PrimaryLanguage/Text/@id = "TN_SV_FU_SystemCommand_BM_UNLOCK_S" i) Read ExternalTextCollection/PrimaryLanguage/Text[@id = "TN_SV_FU_SystemCommand_BM_UNLOCK_S"]/@value j) Identify Read ExternalTextCollection/PrimaryLanguage/Text/@id = "TN_SV_FU_SystemCommand_BM_UNLOCK_F" k) Read ExternalTextCollection/PrimaryLanguage/Text[@id = "TN_SV_FU_SystemCommand_BM_UNLOCK_F"]/@value l) Identify Read ExternalTextCollection/PrimaryLanguage/Text/@id = "TN_SV_FU_SystemCommand_BM_UNLOCK_T" m) Read ExternalTextCollection/PrimaryLanguage/Text[@id = "TN_SV_FU_SystemCommand_BM_UNLOCK_T"]/@value n) Look for ExternalTextCollection/PrimaryLanguage/Text/@id = "TN_SV_FU_SystemCommand_BM_ACTIVATE" o) Identify VariableCollection/Variable/@index="17341" p) Read VariableCollection/Variable[@index="17341"]/@accessRights q) Read VariableCollection/Variable[@index="17341"]/@id r) Read VariableCollection/Variable[@index="17341"]/DataType/@xsi:type s) Read VariableCollection/Variable[@index="17341"]/DataType/@encoding t) Read VariableCollection/Variable[@index="17341"]/DataType/@fixedLength u) Read VariableCollection/Variable[@index="17341"]/Name/@textId v) Identify VariableCollection/Variable/@index="17342" w) Read VariableCollection/Variable[@index="17342"]/@accessRights x) Read VariableCollection/Variable[@index="17342"]/@id y) Read VariableCollection/Variable[@index="17342"]/DataType/@xsi:type z) Read VariableCollection/Variable[@index="17342"]/DataType/@encoding aa) Read VariableCollection/Variable[@index="17342"]/DataType/@fixedLength ab) Read VariableCollection/Variable[@index="17342"]/Name/@textId ac) Identify VariableCollection/Variable/@index="17343"

TEST CASE	CONDITIONS / PERFORMANCE
	ad) Read VariableCollection/Variable[@index="17343"]/@accessRights ae) Read VariableCollection/Variable[@index="17343"]/@id af) Read VariableCollection/Variable[@index="17343"]/DataType/@xsi:type ag) Read VariableCollection/Variable[@index="17343"]/DataType/@bitLength ah) Read VariableCollection/Variable[@index="17343"]/Name/@textId ai) Identify VariableCollection/Variable[@index="17343"]/DataType/SingleValue/@value="0" aj) Read VariableCollection/Variable[@index="17343"]/DataType/SingleValue[@value="0"]/Name/@textId ak) Identify VariableCollection/Variable[@index="17343"]/DataType/SingleValue/@value="1" al) Read VariableCollection/Variable[@index="17343"]/DataType/SingleValue[@value="1"]/Name/@textId am) Identify Read ExternalTextCollection/PrimaryLanguage/Text/@id="TN_V_FU_FW_Password" an) Read ExternalTextCollection/PrimaryLanguage/Text[@id="TN_V_FU_FW_Password"]/@value ao) Identify Read ExternalTextCollection/PrimaryLanguage/Text/@id="TN_V_FU_HW_ID_Key" ap) Read ExternalTextCollection/PrimaryLanguage/Text[@id="TN_V_FU_HW_ID_Key"]/@value aq) Identify Read ExternalTextCollection/PrimaryLanguage/Text/@id="TN_V_FU_BootmodeStatus" ar) Read ExternalTextCollection/PrimaryLanguage/Text[@id="TN_V_FU_BootmodeStatus"]/@value as) Identify Read ExternalTextCollection/PrimaryLanguage/Text/@id="TN_SV_FU_BootmodeStatus_inactive" at) Read ExternalTextCollection/PrimaryLanguage/Text[@id="TN_SV_FU_BootmodeStatus_inactive"]/@value au) Identify Read ExternalTextCollection/PrimaryLanguage/Text/@id="TN_SV_FU_BootmodeStatus_active" av) Read ExternalTextCollection/PrimaryLanguage/Text[@id="TN_SV_FU_BootmodeStatus_active"]/@value aw) Look for VariableCollection/UserInterface/MenuCollection/Menu/VariableRef/@variableId="V_SystemCommand" & .../Button/@buttonValue="80" ax) Look for VariableCollection/UserInterface/MenuCollection/Menu/VariableRef/@variableId="V_SystemCommand" & .../Button/@buttonValue="81" ay) Look for VariableCollection/UserInterface/MenuCollection/Menu/VariableRef/@variableId="V_SystemCommand" & .../Button/@buttonValue="82" az) Look for VariableCollection/UserInterface/MenuCollection/Menu/VariableRef/@variableId="V_SystemCommand" & .../Button/@buttonValue="83" ba) Read Features/@profileCharacteristic bb) Navigate to menu referenced by SpecialistRoleMenuSet/Identification/@menuId. Look for VariableRef with variableId="V_FU_HW_ID_Key" within this menu or menu referenced by this menu.
Input parameter	–
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) check if value is existing b) textId="TN_SV_FU_SystemCommand_BM_UNLOCK_S" c) check if value is existing d) textId="TN_SV_FU_SystemCommand_BM_UNLOCK_F" e) check if value is existing f) textId="TN_SV_FU_SystemCommand_BM_UNLOCK_T" g) check if value is not existing (Test to fail) h) check if id is existing i) value="Start unlocking sequence" j) check if id is existing k) value="Unlocking command 1" l) check if id is existing m) value="Unlocking command 2" n) check if id is not existing (Test to fail) o) check if index is existing p) accessRights="wo" q) id="V_FU_FW_Password" r) xsi:type="StringT" s) encoding="UTF-8" t) fixedLength <= 64 u) textId="TN_V_FU_FW_Password" v) check if index is existing w) accessRights="ro" x) id="V_FU_HW_ID_Key"

TEST CASE RESULTS	CHECK / REACTION
	y) xsi:type="StringT" z) encoding="UTF-8" aa) fixedLength <= 64 ab) textId="TN_V_FU_HW_ID_Key" ac) check if index is existing ad) accessRights="ro" ae) id="V_BootmodeStatus" af) xsi:type="UIntegerT" ag) bitLength="8" ah) textId="TN_V_FU_BootmodeStatus" ai) check if value is existing aj) textId="TN_SV_FU_BootmodeStatus_inactive" ak) check if value is existing al) textId="TN_SV_FU_BootmodeStatus_active" am) check if value is existing an) value="Firmware Password" ao) check if value is existing ap) value="Hardware Identification Key" aq) check if value is existing ar) value="Bootmode Status" as) check if value is existing at) value="Bootloader is inactive" au) check if value is existing av) value="Bootloader is active" aw) check if variableId is not existing (Test to fail) ax) check if variableId is not existing (Test to fail) ay) check if variableId is not existing (Test to fail) az) check if variableId is not existing (Test to fail) ba) check if string contains value "16384" bb) check if variableId is existing
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.5 Host implementation of Firmware Update

E.5.1 Required test firmware files

The IOLFW files listed in Table E.53 shall be provided with the FWBD to support test-to-pass and test-to-fail of firmware updates.

Table E.53 – Required test firmware files

File	Description
IOLFW_FILE_1	Default firmware
IOLFW_FILE_2	Test firmware 2 (no password required, used for test to pass)
IOLFW_FILE_3	Test firmware 3 (password required, used for test to pass)
IOLFW_FILE_4	Test firmware with different VendorID (used for test to fail)
IOLFW_FILE_5	Test firmware with different HW_ID_KEY (used for test to fail)

E.5.2 Write w/o password and Device in TFW mode

Table E.54 defines the test conditions for this test case.

Table E.54 – Write w/o password and Device in TFW mode

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0044
Name	TCD_HOST_FWUP_WITHOUT_PW_TFW
Purpose (short)	Positive test of FWUP write without password and Device in TFW mode (Device BootmodeStatus = 0x00)
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Execution of Device Firmware Update, test to pass
Occurrence	Host application supports firmware update
Specification (clause)	[1] 7.7.5 Figure 32 (T1, T2, T4, T6, T8, T9, T11, T13, T14, T15, T17, T20)
Configuration / setup	Host application connected via IOL-Master to FWBD device
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Testing of all steps of a correct Firmware update without password.
Precondition	An FWBD Device is connected to an IO-Link Master controlled by a host application. Current firmware in FWBD is different to firmware provided by IOLFW_FILE_2. FWBD indicates its current firmware via display or IOL parameter.
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Start the firmware update using IOLFW_FILE_2 c) FWBD checks the received firmware. Only if it is valid, it indicates the firmware revision via display or IOL parameter.
Input parameter	a) "TOM_ID" = 250 b) IOLFW_FILE_2
Post condition	Mode of the EUT and its environment
TEST CASE RESULTS	CHECK / REACTION
Evaluation	c) Host does not indicate any error message and FWBD indicates its firmware revision after host system indicates firmware update finished.
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.5.3 FW-Update w/o password Device in bootloader mode

Table E.55 defines the test conditions for this test case.

Table E.55 – FW-Update w/o password Device in bootloader mode

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0045
Name	TCD_HOST_FWUP_WITHOUT_PW_BOOT
Purpose (short)	Positive test of FWUP write without password Device in bootloader mode (Device BootmodeStatus = 0x01)
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Execution of Device Firmware Update, test to pass
Occurrence	Host application supports firmware update
Specification (clause)	[1] 7.7.5; Figure 32 (T1, T3, T13, T14, T15, T17, T20)
Configuration / setup	Host application connected via IOL-Master to FWBD device
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Testing of all steps of a correct Firmware update without password when Device is already in bootmode.
Precondition	An FWBD Device is connected to an IO-Link Master that is controlled by a host application. FWBD indicates "BOOTMODE".
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Start the firmware update using IOLFW_FILE_2 c) FWBD checks the received firmware. Only if it is valid, it indicates the firmware revision via display or IOL parameter
Input parameter	a) "TOM_ID" = 251 b) IOLFW_FILE_2
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	c) Host does not indicate any error message and FWBD indicates its firmware revision after host system indicates firmware update being finished.
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.5.4 FW-Update with password

Table E.56 defines the test conditions for this test case.

Table E.56 – FW-Update with password

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0046
Name	TCD_HOST_FWUP_WITH_PW
Purpose (short)	Positive test of FWUP write with password
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Execution of Device Firmware Update, test to pass
Occurrence	Host application supports firmware update
Specification (clause)	[1] 7.7.5; Figure 32 (T1, T2, T4, T6, T8, T9, T11, T13, T14, T15, T17, T20)
Configuration / setup	Host application connected via IOL-Master to FWBD device
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Testing of all steps of a correct Firmware update with password
Precondition	An FWBD Device is connected to an IO-Link Master controlled by a host application. Current firmware in FWBD is different to firmware provided by IOLFW_FILE_2. FWBD indicates its current firmware via display or IOL parameter.
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Start the firmware update using IOLFW_FILE_3 c) Host asks the user to enter the firmware password d) Host starts performs the firmware update e) FWBD checks the received firmware. Only if it is valid, it indicates the firmware revision via display or IOL parameter
Input parameter	a) "TOM_ID" = 252 b) IOLFW_FILE_3 (firmware update password required) c) Password for IOLFW_FILE_3
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	e) Host does not indicate any error message and FWBD indicates its firmware revision after host system indicates firmware update finished.
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.5.5 Incorrect VendorID

Table E.57 defines the test conditions for this test case.

Table E.57 – Incorrect VendorID

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0047
Name	TCD_HOST_FWUP_WRONG_VENDOR_ID_TFW
Purpose (short)	Test of host application behavior if VendorID is incorrect
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Execution of Device Firmware Update, test to fail
Occurrence	Host application supports firmware update
Specification (clause)	[1] 7.5.4 Figure 27
Configuration / setup	Host application connected via IOL-Master to FWBD device
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Host shall check if the VendorID of the IOLFW file matches the VendorID of the Device to update.
Precondition	An FWBD Device is connected to an IO-Link Master controlled by a host application.
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Start the firmware update using IOLFW_FILE_4 c) Host application indicates error message (VendorID mismatch) d) FWBD checks if unlock sequence is not started. If unlock sequence is started, it displays "FWUPERROR"
Input parameter	a) "TOM_ID" = 253 b) IOLFW_FILE_4 (VendorID different to VendorID of Device)
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) FWBD displays "ACTIVE" c) Host indicates error message and does not start the firmware update d) FWBD does not show "FWUPERROR"
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.5.6 Incorrect VendorID and Device in bootloader mode

Table E.58 defines the test conditions for this test case.

Table E.58 – Incorrect VendorID and Device in bootloader mode

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0048
Name	TCD_HOST_FWUP_WRONG_VENDOR_ID_BOOT
Purpose (short)	Test of host application behavior if VendorID is incorrect and Device in bootloader mode
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Execution of Device Firmware Update, test to fail
Occurrence	Host application supports firmware update
Specification (clause)	[1] 7.5.4 Figure 27
Configuration / setup	Host application connected via IOL-Master to FWBD device
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Host shall check if the VendorID of the IOLFW file matches the VendorID of the Device to be updated.
Precondition	A FWBD device is connected to an IO-Link Master controlled by a host application. Device is in bootloader mode.
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Start the firmware update using IOLFW_FILE_4 c) Host application indicates error message (Vendor ID mismatch) d) FWBD checks if BLOB transfer via BLOB_ID 1 is started. If it is started, it displays "FWUPERROR"
Input parameter	a) "TOM_ID" = 254 b) IOLFW_FILE_4 (VendorID different from VendorID of the Device)
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) FWBD displays "ACTIVE" c) Host indicates error message and does not start the firmware update d) FWBD does not show "FWUPERROR"
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.5.7 No matching HW_ID_KEY in the IOLFW file

Table E.59 defines the test conditions for this test case.

Table E.59 – No matching HW_ID_KEY in the IOLFW file

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0049
Name	TCD_HOST_FWUP_HW_ID_KEY_MISMATCH_TFW
Purpose (short)	Test of host application behavior if no matching HW_ID_KEY is in the IOLFW file
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Execution of Device Firmware Update, test to fail
Occurrence	Host application supports firmware update
Specification (clause)	[1] 7.5.4 Figure 27
Configuration / setup	Host application connected via IOL-Master to FWBD device
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Host shall check if one of the HW_ID_KEYS in IOLFW file matches the HW_ID_KEY of the Device to be updated and shall not start the firmware update.
Precondition	An FWBD Device is connected to an IO-Link Master controlled by a host application.
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Start the firmware update using IOLFW_FILE_5 c) Host application indicates error message (HW_ID_KEY mismatch) d) FWBD checks if unlock sequence is not started. If unlock sequence is started, it displays "FWUPERROR"
Input parameter	a) "TOM_ID" = 255 b) IOLFW_FILE_5 (HW_ID_KEY different to HW_ID_KEY of the Device)
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) FWBD displays "ACTIVE" c) Host indicates error message and does not start the firmware update d) FWBD does not show "FWUPERROR"
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.5.8 No matching HW_ID_KEY in the IOLFW file and Device in bootloader mode

Table E.60 defines the test conditions for this test case.

Table E.60 – No matching HW_ID_KEY in the IOLFW file & Device in bootloader mode

TEST CASE ATTRIBUTES	IDENTIFICATION / REFERENCE
Identification (ID)	BTFU_TC_0050
Name	TCD_HOST_FWUP_HW_ID_KEY_MISMATCH_BOOT
Purpose (short)	Test of host application behavior if no matching HW_ID_KEY is in the IOLFW file and Device in bootloader mode
Equipment under test (EUT)	Host (PC tool or PLC program)
Test case version	1.0
Category / type	Execution of Device Firmware Update, test to fail
Occurrence	Host application supports firmware update
Specification (clause)	[1] 7.5.4 Figure 27
Configuration / setup	Host application connected via IOL-Master to FWBD device
TEST CASE	CONDITIONS / PERFORMANCE
Purpose (detailed)	Host shall check if one of the HW_ID_KEYS in IOLFW file matches the HW_ID_KEY of the Device to be updated and shall not start the firmware update.
Precondition	An FWBD Device is connected to an IO-Link Master controlled by a host application. Device is in bootload mode.
Procedure	a) Configure FWBD to run test with "TOM_ID" b) Start the firmware update using IOLFW_FILE_5 c) Host application indicates error message (HW_ID_KEY mismatch) d) FWBD checks BLOB transfer via BLOB_ID 1 is started. If it is started, it displays "FWUPERROR"
Input parameter	a) "TOM_ID" = 256 b) IOLFW_FILE_5 (HW_ID_KEY different to HW_ID_KEY of the Device)
Post condition	–
TEST CASE RESULTS	CHECK / REACTION
Evaluation	a) FWBD displays "ACTIVE" c) Host indicates error message and does not start the firmware update d) FWBD does not show "FWUPERROR"
Test passed	All steps of evaluation return expected result.
Test failed (examples)	Any of the evaluation steps fails.
Results	

E.6 Test Tools

E.6.1 Device Tester

Figure E.1 shows the principle of a Device-Tester system comprising:

- A Device-Tester hardware with at least one SDCI port, which can be a modified standard SDCI Master with an adequate communication interface to a personal computer,
- A personal computer supporting the communication interface of the Device-Tester hardware,
- Device-Tester software running on that personal computer serving as a control and monitoring program for the Device-Tester hardware,
- An SDCI Device, the "equipment under test" (EUT) that shall be tested for conformity.

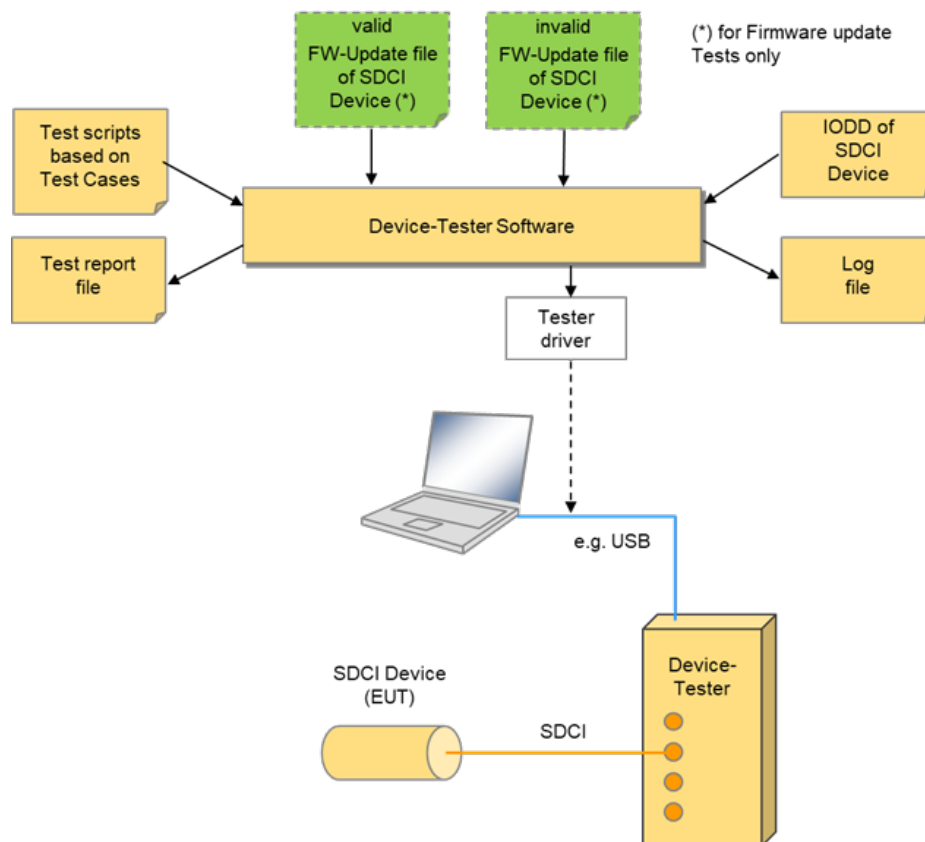


Figure E.1 – BLOB and Firmware Update Device Tester system

Table E.61 shows the system requirements for the Device Tester.

Table E.61 – System requirements for the Device Tester

Requirement	Description
SR1	The Device-Tester system shall execute and evaluate the test cases defined in this specification. This can include some functions or behavior not defined in the SDCI specification but is necessary to run the EUT into a specific state, e.g. generation of checksum errors.
SR2	The result of each test case and also additional information about the test execution shall be reported to the user (test report, log file). The user shall be able to store and print this information.
SR3	The conformity test cases shall be secured against manipulation.
SR4	Optional requirement: The Device-Tester can interpret a valid IODD and generate different settings which are required for the conformity test.

E.6.2 Firmware Update and BLOB test Device (FWBD)

It is nearly impossible to test BLOB transfer and firmware update manually by evaluating ISDU requests. Therefore, it is recommended to implement a test Device, based on a standard IO-Link Device with support for BLOB transfer and firmware update.

This test Device (FWBD) shall support some special features to check if commands from the host application are correct and it must be able to generate errors (e.g. send an invalid BLOB checksum). Features of such a test Device are described in the test cases.

The basic features of an FWBD are:

- Configurable, in order to support the different test cases. The "TOM_ID" (Test operation mode) defines the current test case;
- Provide a BLOB ID for read, returning one of the defined test sequences depending on the current "TOM_ID";
- Provide a BLOB ID for write, checking if the received data equals one of the defined test sequences depending on the current "TOM_ID";
- Support different maximum ISDU sizes;
- Create some error responses depending on current "TOM_ID";
- Support firmware update.

The FWBD hardware can be used to test the BLOB host application and the Firmware Update Tool.

The FWBD shall indicate the test result by its state. The result state can be shown on a display or by several LEDs. Table E.62 shows the currently defined result states (list might be extended when more test cases are defined). The expected result is defined by the particular test cases.

Table E.62 – FWBD result states

Device State	Description
ACTIVE	FWBD is configured with a TOM_ID and ready to run the test
OPERATE	FWBD test is operational
OK	No error, test has been finished successfully
WBID	Wrong BLOB ID
BLOBERROR	Error during BLOB transfer
FWUPERROR	Error during Firmware Update
ABORT	BLOB Abort received
ERROR	If any other error occurs
BOOTMODE	Incorrect BootmodeStatus, Device is in Bootloader (Bootmode-Status = 1)

The FWBD shall support the BLOB_IDs specified in Table E.22.

E.6.3 BLOB Host Application Tester System

Figure E.2 shows the principle of a BLOB Host Application Tester System comprising:

- An FWBD Application-Tester hardware with a communication interface to a personal computer, for example USB (Universal Serial Bus);
- A personal computer, supporting the communication interface of the FWBD Application-Tester hardware and a communication interface to the host running the BLOB host application, for example Ethernet.

- A BLOB Host Application-Tester software running on that personal computer serving as a control and monitoring program for the FWBD Application-Tester hardware.
- A BLOB Host Application "Equipment under Test" (EUT) that shall be tested for conformity. The Host Application can be a function block (FB) running on a host system like a PLC. The host usually has a communication interface to parametrize and control the function.

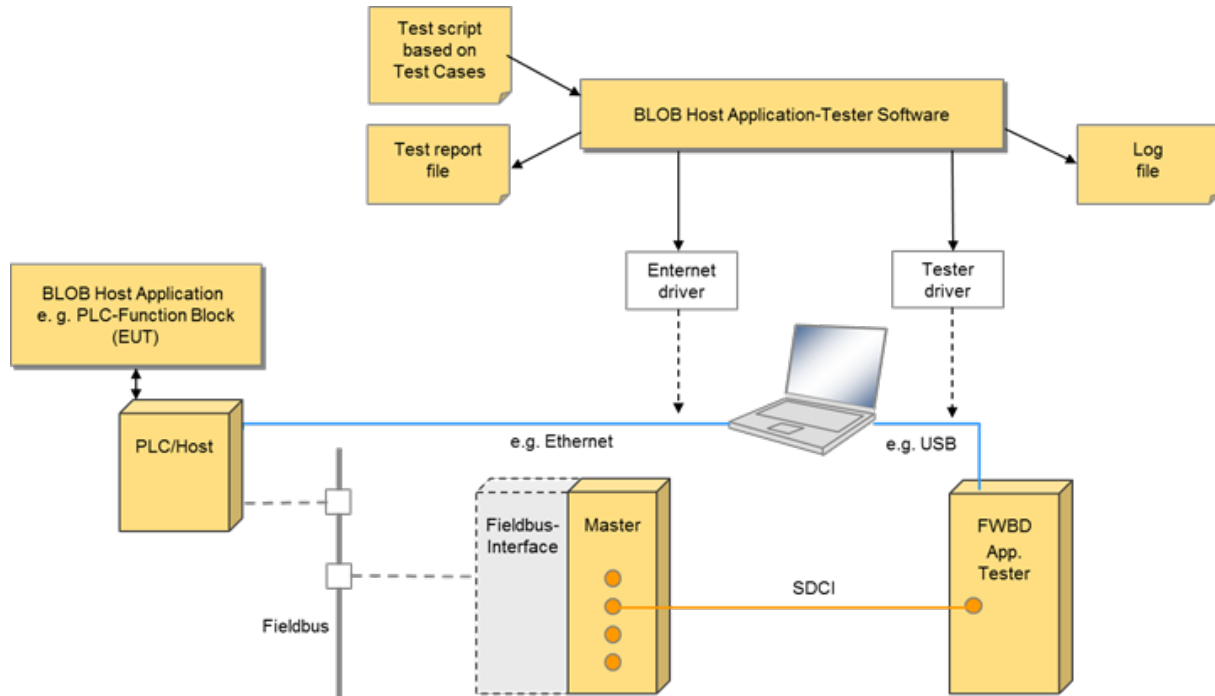


Figure E.2 – BLOB Host Application Tester System

The FWBD Application-Tester hardware can be used to test the BLOB host application and the Firmware Update Tool.

E.6.4 Firmware Update Application Test System

Figure E.3 shows the principle of a Firmware Update Application-Tester system comprising:

- An FWBD Application-Tester hardware with any communication interface to a personal computer, e.g. USB (Universal Serial Bus);
- A personal computer supporting the communication interface of the FWBD Application-Tester hardware;
- A BLOB Host Application-Tester software running on that personal computer serving as a control and monitoring program for the FWBD Application-Tester hardware;
- A personal computer supporting the communication interface to the SCDI-Master;
- A Firmware Update Application "equipment under test" (EUT) that shall be tested for conformity.

The FWBD Application-Tester hardware can be used to test the BLOB host application and the Firmware Update Tool

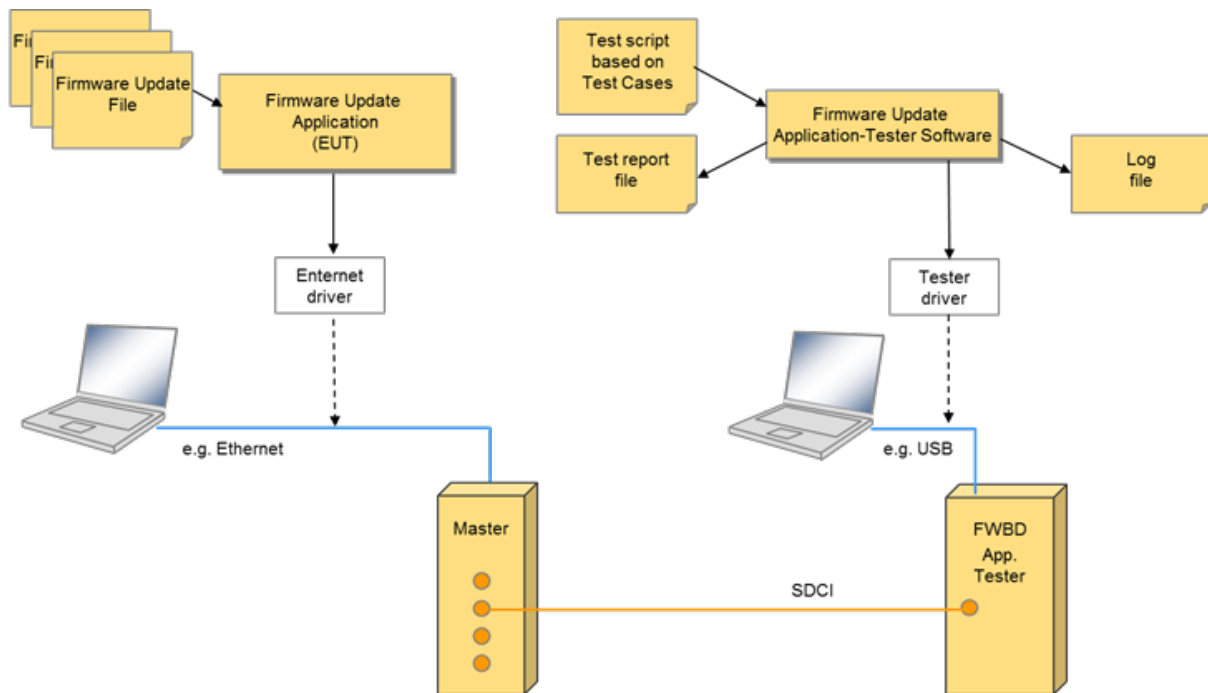


Figure E.3 – Firmware Update Application Tester System

E.6.5 Random Data

E.6.5.1 Random number generator

Pseudo-random test sequences allow for the generation of a well-defined payload at the host application and at the Device. A uniform payload enables a simple test for consistency. The payload is defined as an octet stream and shall be calculated using the function "iol_rand()" (see Figure E.4). The respective seed values (SEED) can be found in Table E.3 and Table E.23.

```
static uint16_t rand16 = SEED;
static uint16_t previousRand16 = 1;

uint8_t iol_rand()
{
    uint16_t tmp;
    tmp = (previousRand16 ^ (previousRand16 << 5));
    previousRand16 = rand16;
    rand16 = (rand16 ^ (rand16 >> 1)) ^ (tmp ^ (tmp >> 3));
    return (rand16 & 0xff); // return new 8-bit random number
}
```

Figure E.4 – C code sample of the "iol_rand()" function

All Devices supporting the BLOB profile shall support these IDs and shall be able to provide the corresponding test payload (random numbers). The random numbers can be generated on the fly or picked from a look-up table (see Table E.63). The random BLOB which is related to the ID shall be available after BLOB_Start.

E.6.5.2 Pre-calculated random data

The pseudo random numbers are identical for each run and can therefore be pre-calculated. For some applications it makes sense to store the data as an array in the system.

Table E.63 shows pre-calculated values for seed values 1 to 3 and the corresponding BLOB lengths.

1833

Table E.63 – Pre-calculated random numbers

SEED	Length	Values															
1	5	36	19	42	2	68											
2	579	38	127	58	75	123	200	52	127	66	47	58	49	60	241	2	168
		246	208	8	102	124	208	27	60	246	6	53	49	174	42	92	21
		152	55	135	201	102	225	67	91	49	213	12	17	196	145	245	8
		115	195	211	45	158	45	14	53	56	67	93	20	252	68	117	211
		149	122	20	99	20	61	197	169	20	166	51	160	73	217	49	47
		235	200	150	140	214	16	172	168	117	242	68	98	47	158	39	193
		126	188	11	181	73	122	131	95	175	48	238	15	99	224	83	166
		160	154	131	35	45	16	68	84	2	13	193	249	56	198	122	193
		28	111	55	18	182	245	55	83	119	185	32	154	51	11	215	58
		150	200	176	249	195	64	183	164	203	174	167	247	35	103	191	151
		136	181	150	170	35	37	28	135	107	149	245	252	61	208	150	247
		80	13	241	241	162	88	202	92	137	234	163	141	112	160	238	109
		240	188	236	33	144	89	247	90	226	42	73	202	235	37	237	206
		237	148	122	209	100	121	189	17	231	131	197	184	45	180	146	201
		133	19	187	155	150	249	153	215	18	170	7	179	209	144	55	14
		36	97	1	136	9	52	2	65	171	83	232	137	152	200	167	5
		40	237	86	217	113	207	83	208	205	193	1	28	23	115	67	147
		61	142	167	115	69	182	116	118	103	148	233	11	94	183	65	92
		127	7	136	151	197	254	40	197	42	22	250	203	195	124	245	236
		165	246	168	76	55	249	40	62	12	235	99	136	47	65	87	177
		232	191	117	187	137	45	39	168	163	15	211	72	73	108	158	32
		229	19	151	161	104	205	78	105	194	195	240	95	126	201	196	82
		55	28	95	159	56	244	145	3	133	200	141	154	31	177	73	78
		141	148	22	235	242	81	29	172	112	97	15	33	99	19	153	232
		50	62	29	242	1	101	178	54	97	89	156	132	25	177	151	95
		73	133	227	211	169	120	220	5	208	105	55	140	167	89	90	33
		232	189	150	227	110	202	212	165	41	184	140	197	10	134	134	75
		224	158	173	14	37	160	5	179	155	31	52	78	107	97	219	127
		108	214	172	110	176	179	12	227	207	187	140	74	34	216	218	116
		103	126	214	212	234	129	222	132	91	18	231	76	173	206	165	56
		193	190	28	67	237	60	230	126	131	139	145	79	207	50	33	173
		122	75	243	228	119	4	35	166	254	43	103	182	58	159	5	119
		189	198	91	144	4	132	18	255	120	216	163	17	34	132	40	56
		137	18	135	252	233	142	121	162	33	175	185	35	88	223	227	138
		28	193	110	228	173	19	214	64	244	230	143	202	45	224	236	102
		84	76	132	211	94	52	163	144	67	192	167	44	181	22	138	195
		145	46	190													
3	4639	39	91	41	97	121	140	24	121	94	99	127	227	254	210	130	99
		104	127	229	59	171	238	235	181	181	56	19	5	26	70	244	120

SEED	Length	Values															
		30	38	144	2	97	251	15	128	178	91	122	27	171	202	189	56
		122	120	25	130	125	25	115	20	175	190	71	213	44		

1834

1835

Annex F (informative)

Information on conformance testing of profile Devices

Information about testing profile Devices for conformity can be obtained from the following organization:

IO-Link Community

c/o PROFIBUS Nutzerorganisation e.V.

Ohiostrasse 8

76149 Karlsruhe

Germany

Phone: +49 (0) 721 / 98 61 97 0

Fax: +49 (0) 721 / 98 61 97 11

E-mail: info@io-link.com

Web site: <http://www.io-link.com>

Bibliography

- 1853
- 1854 [1] IO-Link Community, *IO-Link Interface and System*, V1.1.4, March 2024, Order No.
1855 10.002
- 1856 [2] IO-Link Community, *IO Device Description (IODD)*, V1.1.4, March 2024, Order No.
1857 10.012
- 1858 [3] IEC/TR 62390:2005, *Common automation device profile guideline*
- 1859 [4] IEC 60050 (all parts), International Electrotechnical Vocabulary
- 1860 NOTE See also the IEC Multilingual Dictionary – Electricity, Electronics and Telecommunications (availa-
1861 ble on CD-ROM and at <<http://domino.iec.ch/iev>>).
- 1862 [5] IO-Link Community, *IO-Link Communication*, V1.0, January 2009, Order No. 10.002
- 1863 [6] IO-Link Community, *IO-Link Test Specification*, V1.1.4, March 2024, Order No. 10.032
- 1864 [7] IO-Link Community, *IO-Link Smart Sensor Profile 2nd Ed*, V1.2, January 2024, Order
1865 No. 10.042
- 1866 [8] IO-Link Community, *IO-Link Common Profile*, V1.2, January 2024, Order No. 10.072
- 1867 [9] <http://www.faqs.org/faqs/compression-faq/>
- 1868 [10] https://en.wikipedia.org/wiki/Ross_Williams_%28computer_scientist%29
- 1869 [11] <https://en.wikipedia.org/wiki/LZRW>
- 1870 [12] <http://www.ross.net/compression/>
- 1871 [13] <https://en.wikipedia.org/wiki/DEFLATE>
- 1872 [14] <https://tools.ietf.org/html/rfc1951>
- 1873 [15] https://en.wikipedia.org/wiki/Phil_Katz
- 1874 [16] <https://en.wikipedia.org/wiki/Zlib>
- 1875 [17] <https://tools.ietf.org/html/rfc1950>
- 1876 [18] <https://tools.ietf.org/html/rfc1952>
- 1877 [19] <https://en.wikipedia.org/wiki/Lempel%E2%80%93Ziv%E2%80%93Oberhumer>
- 1878 [20] <http://www.oberhumer.com/opensource/lzo/>
- 1879 [21] http://www.internet-sicherheit.de/fileadmin/docs/service/tipps_zur_internet-sicherheit/sicherheit_von_verschluesselungsalgorithmen/2014-05-26_Cryptographic_Algorithms_A4.pdf
- 1880
- 1881
- 1882 [22] https://en.wikipedia.org/wiki/Cyclic_redundancy_check
- 1883 [23] Koopman, P., *32-Bit Cyclic Redundancy Codes for Internet Applications*, The Interna-
1884 tional Conference on Dependable Systems and Networks (DSN), 2002
- 1885 [24] Castagnoli, G., Braeuer, S. & Herrman, M., *Optimization of Cyclic Redundancy-Check*
1886 *Codes with 24 and 32 Parity Bits*, IEEE Transactions on Communications, Vol. 41, No.
1887 6, June 1993.
- 1888 [25] <http://create.stephan-brumme.com/crc32/>
- 1889
-

© Copyright by:

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

