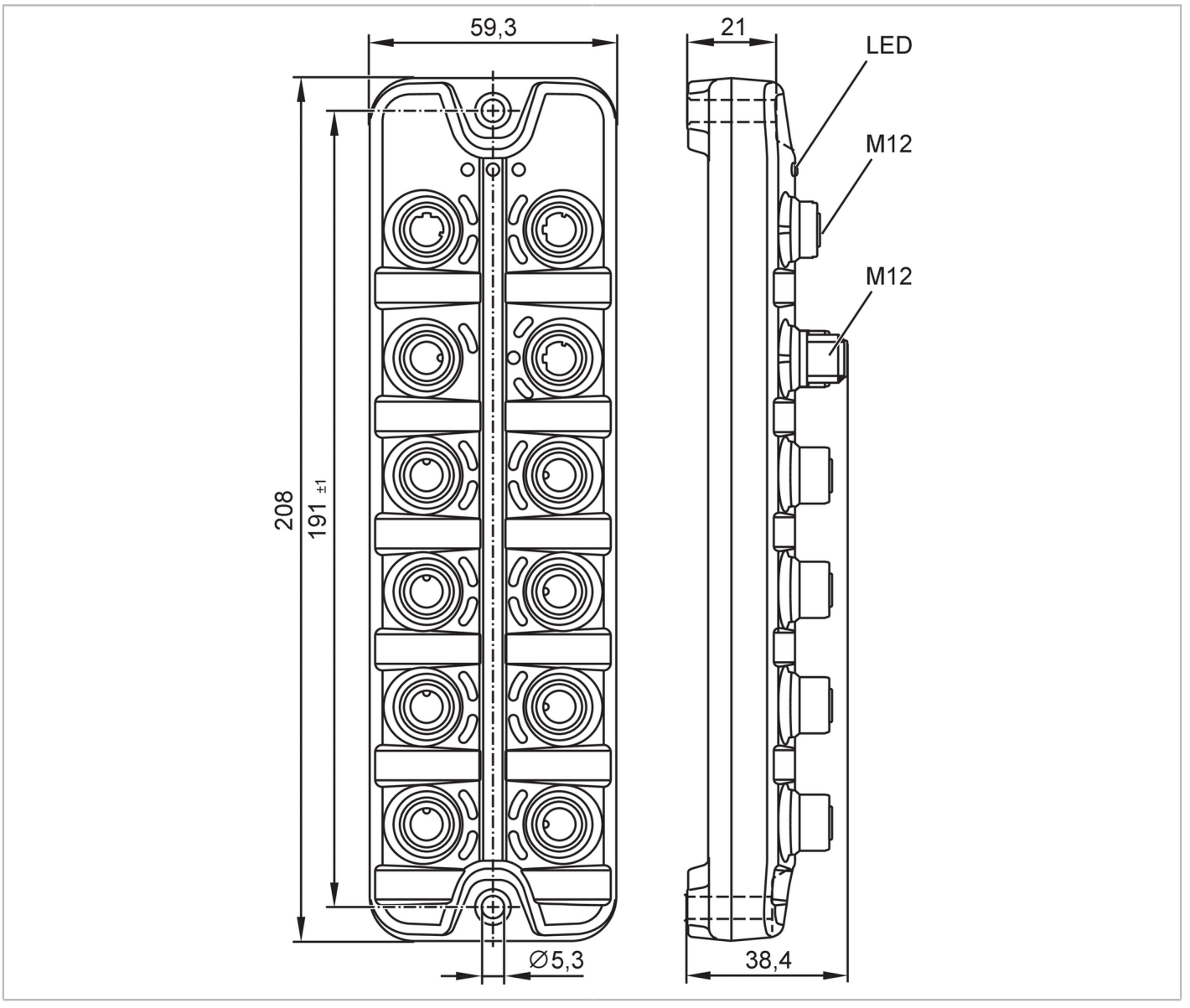
AL1342

IO-Link master with Modbus TCP interface

IO-Link Master DL MB 8P IP67



Application		
Application		I/O modules for field applications
Daisy-chain function		fieldbus interface
Electrical data		
Operating voltage	[V]	20...30 DC; (US ; to SELV/PELV)
Current consumption	[mA]	300...3900; (US)
Protection class		III
Sensor supply US		
Max. current load total	[A]	3.6
Inputs / outputs		
Total number of inputs and outputs		16; (configurable)
Number of inputs and outputs		Number of digital inputs: 16; Number of digital outputs: 8

AL1342

IO-Link master with Modbus TCP interface

IO-Link Master DL MB 8P IP67

Inputs			
Number of digital inputs		16; (IO-Link Port Class A: 8 x 2)	
Switching level high	[V]	11...30	
Switching level low	[V]	0...5	
Digital inputs protected against short circuits		yes	
Outputs			
Number of digital outputs		8; (IO-Link Port Class A: 8 x 1)	
Max. current load per output	[mA]	300	
Short-circuit proof		yes	
Interfaces			
Communication interface		Ethernet; IO-Link	
Ethernet - Modbus TCP			
Transmission standard		10Base-T; 100Base-TX	
Transmission rate		10 MBit/s; 100 MBit/s	
Protocol		Modbus TCP	
Factory settings		IP address: 192.168.1.250	
		subnet mask: 255.255.255.0	
		gateway IP address: 0.0.0.0	
		MAC address: see type label	
Ethernet - Internet of Things			
Transmission standard		10Base-T; 100Base-TX	
Transmission rate		10 MBit/s; 100 MBit/s	
Protocol		MQTT JSON	
Factory settings		IP address: 169.254.x.x	
		subnet mask: 255.255.0.0	
		gateway IP address: 0.0.0.0	
		MAC address: see type label	
Note on interfaces		Safety protocol: HTTPS	
IO-Link Master			
Transmission type		COM1 (4,8 kBaud); COM2 (38,4 kBaud); COM3 (230,4 kBaud)	
IO-Link revision		1.1	
Number of ports class A		8	
Operating conditions			
Ambient temperature	[°C]	-25...60	
Storage temperature	[°C]	-25...85	
Max. relative air humidity	[%]	90	
Protection		IP 65; IP 66; IP 67	
Protection rating (NEMA 250)		6P	
Pollution degree		2	
Chemical media		ISO 16750-5	AA, BA, BD, HLP, CC, DB, DC, DD, CA
		NEMA 250 5.13.1	AA
Tests / approvals			
EMC		EN 61000-6-2	
		EN 61000-6-4	

AL1342

IO-Link master with Modbus TCP interface

IO-Link Master DL MB 8P IP67

Shock resistance		DIN EN 60068-2-27	
Vibration resistance		DIN EN 60068-2-64 2009-04	
		DIN EN 60068-2-6 2008-10	
MTTF	[years]	78	

Mechanical data

Weight	[g]	403	
Housing		rectangular	
Type of mounting		Backplane mounting	
Dimensions	[mm]	208 x 59.3 x 38.4	
Material		housing: PA orange; Socket: brass nickel-plated	
Sealing material		FKM	

Remarks

Remarks		Further information is available in the manual.	
Pack quantity		1 pcs.	

Electrical connection - Ethernet

Connector: 2 x M12; coding: D; Contacts: 4; sealing: FKM



Modbus TCP X21, X22

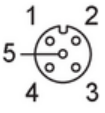
1	TX +
2	RX +
3	TX -
4	RX -
5	not used

IoT X23

1	TX +
2	RX +
3	TX -
4	RX -
5	not used

Electrical connection - IO-Link

Connector: 8 x M12; coding: A; Contacts: 4; sealing: FKM



IO-Link Port Class A X01...X08

1	Sensor supply (US) L+
2	Digital input
3	Sensor supply (US) L-
4	C/Q IO-Link
5	not used

AL1342

IO-Link master with Modbus TCP interface

IO-Link Master DL MB 8P IP67

Electrical connection - voltage supply

Connector: 1 x M12; coding: A; Contacts: 4



X31	
1	+ 24 V DC (US)
2	not used
3	GND (US)
4	not used

Operating instructions

IO-Link master with Modbus TCP interface
DataLine
8 ports
IP 65 / IP 66 / IP 67

AL1342

Firmware: 3.1.x

English

80284136 / 03 10 / 2021

Contents

1 Preliminary note 5

1.1 Legal and copyright information 5
1.2 Purpose of the document 5
1.3 Explanation of Symbols 5
1.4 Change history 6

2 Safety instructions 7

2.1 General 7
2.2 Required background knowledge 7
2.3 Safety symbols on the device 7
2.4 IT security 8

3 Intended use 9

4 Function 10

4.1 Communication, parameter setting, evaluation 11
4.1.1 IO-Link 11
4.1.2 Modbus TCP 11
4.1.3 Internet of Things (IoT) 11
4.1.4 Security mode 11
4.1.5 Parameter setting 11
4.1.6 Visual indication 12
4.2 Digital inputs 12
4.3 IO-Link supply 12

5 Mounting 13

5.1 Mount the device 13

6 Electrical connection 14

6.1 Notes 14
6.2 Connecting the Modbus TCP ports 15
6.3 Connecting the IoT port 15
6.4 IO-Link ports 16
6.4.1 Connecting IO-Link devices for Class A operation 16
6.4.2 Connecting IO-Link devices for Class B operation 17
6.5 Connecting the device 18

7 Operating and display elements 19

7.1 Overview 19
7.2 LED indicators 20
7.2.1 Status LEDs 20
7.2.2 Ethernet ports 20
7.2.3 IoT port 21
7.2.4 Voltage supply 21
7.2.5 IO-Link Ports (Class A) 21

8 Set-up 22

9 Configuration 23

9.1 LR DEVICE 24
9.1.1 Remarks 25
9.1.2 IoT: Configure IP settings 26
9.1.3 IoT: Configure security mode 27
9.1.4 IoT: Configuring access rights 28
9.1.5 IoT: Configure the interface to LR AGENT or LR SMARTOBSERVER 29
9.1.6 Fieldbus: Configure IP settings 30
9.1.7 Fieldbus: set the length of the process data 31
9.1.8 IO-Link ports: Activate data transfer to LR AGENT or LR SMARTOBSERVER 32
9.1.9 IO-Link ports: Configure operating mode 33
9.1.10 IO-Link ports: Set the device validation and data storage 34
9.1.11 IO-Link ports: set fail-safe values 35
9.1.12 Info: Show device information 35
9.1.13 Firmware: Reset device to factory settings 36
9.1.14 Firmware: Reboot the device 36
9.1.15 Configure IO-Link devices 37
9.2 ifm IoT Core 38
9.2.1 Programmers' notes 39
9.2.2 First steps 43
9.2.3 General functions 43
9.2.4 IoT: Configuring access rights 47
9.2.5 IoT: Configuring IP settings 47
9.2.6 IoT: Configuring the LR AGENT or LR SMARTOBSERVER interface 48
9.2.7 IoT: Configuring security mode 48
9.2.8 Fieldbus: Configuring the IP settings 50
9.2.9 Fieldbus: Confguring the fieldbus interface 51
9.2.10 Fieldbus: Setting fail-safe values 52
9.2.11 IO-Link ports: Setting the operating mode of pin 4 (US) 52
9.2.12 IO-Link ports: Configuring device validation and data storage 53
9.2.13 IO-Link ports: Configuring data transfer to LR AGENT or LR SMARTOBSERVER 55
9.2.14 IO-Link ports: Reading / writing process data 55
9.2.15 IO-Link ports: Indicating port events 58
9.2.16 IO-Link devices: Accessing parameters 58
9.2.17 IO-Link devices: Reading an writing device information 59
9.2.18 IO-Link devices: Indicating IO-Link events 60
9.2.19 Gateway: Resetting, rebooting and localising the device 60
9.2.20 Gateway: Reading device information 60
9.2.21 Gateway: Reading status and diagnostic information 60
9.2.22 Gateway: Updating the firmware 61
9.2.23 Gateway: Setting the application tag 63
9.2.24 Subscribing to notifications 64
9.2.25 Using Web Socket 68
9.2.26 MQTT support 70
9.2.27 Using the IoT-Core Visualizer 74
9.3 Modbus TCP 81
9.3.1 Integrate the AL1342 into the Modbus project 81
9.3.2 Set IO-Link master 82
9.3.3 Set IO-Link ports 83
9.3.4 Read input data of several IO-Link ports 84
9.3.5 Read input data of individual IO-Link ports 85
9.3.6 Write output data of several IO-Link ports 86
9.3.7 Write output data of individual IO-Link ports 87
9.3.8 Read diagnostic information and events 88
9.3.9 Read device information 89
9.3.10 Control IO-Link master 89
9.3.11 Configure IO-Link devices 89
9.3.12 Modbus TCP: Programmers' notes 90

10 Operation 92
10.1 Using web-based management 92

11 Maintenance, repair and disposal 93
11.1 Cleaning process 93
11.2 Updating the firmware 93
11.3 Replacing IO-Link device 93

12 Factory settings 94

13 Accessories 95

14 Appendix 96
14.1 Technical data 97
14.1.1 Application 97
14.1.2 Electrical data 97
14.1.3 Inputs / outputs 97
14.1.4 Inputs 98
14.1.5 Outputs 98
14.1.6 Interfaces 98
14.1.7 Operating conditions 99
14.1.8 Approvals / tests 99
14.1.9 Mechanical data 99
14.1.10 Electrical connection 100
14.2 Modbus TCP 101
14.2.1 Register 102
14.2.2 Acyclic commands 118
14.3 ifm IoT Core 127
14.3.1 Overview: IoT profile 128
14.3.2 Overview: IoT types 135
14.3.3 Overview: IoT services 136

15 Index 150

1 Preliminary note

Content

Legal and copyright information 5
Purpose of the document 5
Explanation of Symbols 5
Change history 6

33203

1.1 Legal and copyright information

33117

© All rights reserved by ifm electronic gmbh. No part of this manual may be reproduced and used without the consent of ifm electronic gmbh.

All product names, pictures, companies or other brands used on our pages are the property of the respective rights owners:

- AS-i is the property of the AS-International Association, (→ www.as-interface.net)
- CAN is the property of the CiA (CAN in Automation e.V.), Germany (→ www.can-cia.org)
- CODESYS™ is the property of the CODESYS GmbH, Germany (→ www.codesys.com)
- DeviceNet™ is the property of the ODVA™ (Open DeviceNet Vendor Association), USA (→ www.odva.org)
- EtherNet/IP® is the property of the → ODVA™
- EtherCAT® is a registered trade mark and patented technology, licensed by Beckhoff Automation GmbH, Germany
- IO-Link® is the property of the → PROFIBUS Nutzerorganisation e.V., Germany (→ www.io-link.com)
- ISOBUS is the property of the AEF – Agricultural Industry Electronics Foundation e.V., Deutschland (→ www.aef-online.org)
- Microsoft® is the property of the Microsoft Corporation, USA (→ www.microsoft.com)
- Modbus® is the property of the Schneider Electric SE, France (→ www.schneider-electric.com)
- PROFIBUS® is the property of the PROFIBUS Nutzerorganisation e.V., Germany (→ www.profibus.com)
- PROFINET® is the property of the → PROFIBUS Nutzerorganisation e.V., Germany
- Windows® is the property of the → Microsoft Corporation, USA

1.2 Purpose of the document

34227

This document is only for device types "IO-Link master - Modbus TCP gateway (DataLine) 8 port IP 65 / IP 66 / IP 67" (art. no.: AL1342).

It is part of the device and contains information about the correct handling of the product.

► Read this document before using the device.

► Keep this document during the service life of the device.

1.3 Explanation of Symbols

34171

WARNING

Warning of serious personal injury.

Death or serious irreversible injuries may result.

CAUTION
Warning of personaly injury.
Slight reversible injuries may result.

NOTICE
Warning of damage to property

Important note
Non-compliance can result in malfunction or interference

Information
Supplementary note

► ... Request for action

> ... Reaction, result

→ ... "see"

abc Cross-reference

123 Decimal number
0x123 Hexadecimal number
0b010 Binary number

[...] Designation of pushbuttons, buttons or indications

1.4 Change history

61443

Version	Topic	Date
00	New creation of the document	04 / 2019
01	Correction: Technical data - current rating per output	09 / 2019
02	▪ Added: New IoT Core functions ▪ Added: IoT Core Visualizer ▪ Correction: Description SENS PWR and AUX PWR ▪ Correction: Description of IoT Core service getsubscriptioninfo	10 / 2020
03	Deleted: ifm IoT Core – DNS support	10 / 2021

2 Safety instructions

Content

General 7
Required background knowledge 7
Safety symbols on the device 7
IT security 8

28333

2.1 General

58525

- The device described is a subcomponent for integration into a system. The manufacturer is responsible for the safety of the system. The system manufacturer undertakes to perform a risk assessment and to create documentation in accordance with legal and normative requirements to be provided to the operator and user of the system. This documentation must contain all necessary information and safety instructions for the operator, the user and, if applicable, for any service personnel authorised by the manufacturer of the system.
- Read this document before setting up the product and keep it during the entire service life.
- The product must be suitable for the corresponding applications and environmental conditions without any restrictions.
- Only use the product for its intended purpose (→ **Intended use** (→ p. 9)).
- If the operating instructions or the technical data are not adhered to, personal injury and/or damage to property may occur.
- The manufacturer assumes no liability or warranty for any consequences caused by tampering with the product or incorrect use by the operator.
- Installation, electrical connection, set-up, programming, configuration, operation and maintenance of the product must be carried out by personnel qualified and authorised for the respective activity.
- Protect units and cables against damage.

2.2 Required background knowledge

34185

This document is intended for specialists. Specialists are people who, based on their relevant training and experience, are capable of identifying risks and avoiding potential hazards that may be caused during operation or maintenance of the product.

The document contains information about the correct handling of the product.

2.3 Safety symbols on the device

34199



General warning
Observe instructions in chapter "Electrical connection" (→ **Electrical connection** (→ p. 14))!

2.4 IT security

54678

NOTICE!

If the device is operated in an unprotected network environment.

> Unauthorised read or write access to data is possible.

> Unauthorised manipulation of the device function is possible.

► Check and restrict access options to the device:

- Restrict access to authorised persons.
- Do not connect the device to open networks or the internet.

If access from the internet is inevitable:

► choose a safe method to connect with the device (e. g. VPN).

► Use encrypted data transmission (e. g. https / TLS).

3 Intended use

58283

The device may only be used for the following purposes:

- as IO-Link master for configuration, administration and operation of IO-Link devices
- as gateway between IO-Link devices and AS-i network

The device is designed for use without a control cabinet in plant construction.

► Only use the device within the limits of the technical data (→ **Technical data** (→ p. 97)).

4 Function

Content

Communication, parameter setting, evaluation 11
Digital inputs 12
IO-Link supply 12

33836

4.1 Communication, parameter setting, evaluation

Content

IO-Link 11
Modbus TCP 11
Internet of Things (IoT) 11
Security mode 11
Parameter setting 11
Visual indication 12

33860

4.1.1 IO-Link

34084

The device offers the following IO-Link functions:

- IO-Link master (IO-Link revision 1.0 and 1.1)
- 8 IO-Link ports for connection of IO-Link devices
- Provision of process data of the connected IO-Link devices for LR SMARTOBSERVER monitoring software (→ www.ifm.com)

4.1.2 Modbus TCP

33676

The device offers the following Modbus TCP functions:

- Provision of the functions of a Modbus TCP Slave
- 2 port switch for access to the Modbus TCP interface (X21/X22)
- Gateway for transmission of the process and parameter data between the connected IO-Link devices and the higher-level Modbus TCP controller

4.1.3 Internet of Things (IoT)

54679

The device offers the following IoT functions:

- Gateway for the transmission of process, parameter and monitoring data between IO-Linkmaster / IO-Link devices and the IT network level
- REST-API to access process and parameter data
- Supported protocols: TCP/IP JSON, MQTT

4.1.4 Security mode

54697

The IoT interface offers the following optional sercurity functions:

- Secure data transfer via encrypted connection (Secure Layer Transport - TLS)
- Access protection via authentification

4.1.5 Parameter setting

34210

The device provides the following configuration options:

- Parameter setting of the IO-Link master of the AL1342 with LR DEVICE parameter setting software, Modbus TCP projection software or ifm IoT-Core services.
- Parameter setting of the connected IO-Link devices (sensors, actuators) with LR DEVICE parameter setting software, Modbus TCP projection software or ifm IoT-Core services

- Storage of parameter sets of the connected IO-Link devices for automatic recovery (data storage)

4.1.6 Visual indication

34192

The device has the following visual indicators:

- Status and error indication of the gateway, of the Modbus TCP connection and of the system
- Status display of the voltage supply
- Status and activity display of the Ethernet connection
- Status, error and short circuit/overload indication of the IO-Link ports

4.2 Digital inputs

33817

The device has 8 additional digital inputs (type 2 according to EN 61131-2).

The digital inputs are on pin 2 of the ports X01...X04.

All inputs refer to the potential of the device supply (pin 3).

4.3 IO-Link supply

34077

The device has 8 supplies for IO-Link devices.

The IO-Link ports X01...X08 are ports class A.

Every supply provides short circuit monitoring.

The device ensures fire protection for the connected IO-Link devices by providing a power-restricted circuit at the IO-Link ports (according to IEC61010-1 and Class 2 according to UL1310).

5 Mounting

Content

Mount the device 13

34058

5.1 Mount the device

34059



► Disconnect the system from power before installation.
► For installation choose a flat mounting surface.
► Please observe the maximum tightening torque.

► Fix the unit to the mounting surface using 2 M5 mounting screws and washers.
  ▪ Tightening torque: 1.8 Nm
► Ground the unit via the two mounting screws of the upper mounting lugs.

6 Electrical connection

Content

Notes 14
Connecting the Modbus TCP ports 15
Connecting the IoT port 15
IO-Link ports 16
Connecting the device 18

33805

6.1 Notes

51957

A qualified electrician must connect the unit.

► The national and international for regulations setting up electrical equipment must be complied with.

The unit is only suitable for operation using SELV/PELV voltages.

► Please note the information concerning IO-Link wiring!

This unit contains components that may be damaged or destroyed by electrostatic discharge (ESD).

► Please observe the required precautions against electrostatic discharge!

The M12 connection parts in the device comply with the ingress resistance requirements of the standard EN 61076-2-101. To adhere to the protection rating, only cables certified to this standard must be used. The system creators undertake to ensure ingress resistance for cables which they cut to length themselves.

► Carry out the fitting according to the indications of the cable manufacturer. A maximum of 0.8 Nm is permitted.

► During installation, place the M12 connector vertically so that the coupling nut will not damage the thread.

► Depending on the mounting conditions, cables must be provided with a strain relief to avoid unacceptable loads on the mounting points and M12 connections.

► Make sure that the M12 connection parts are correctly seated and mounted correctly. The specified protection rating can not be guaranteed if this is not observed.

For UL applications:

► To connect the IO-Link master and the IO-Link devices, only use UL-certified cables of the CYJV or PVVA category with a minimum temperature of 80 °C (75 °C in case of maximum ambient temperature of 40 °C).

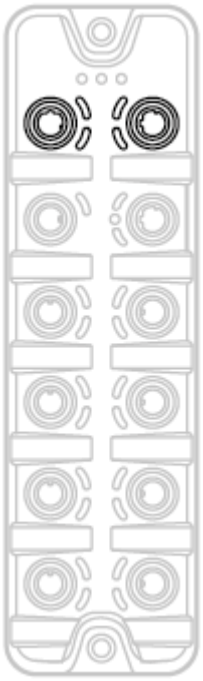
Wiring: → **Technical data** (→ p. 97)

By means of basic insulation according to EN61010-1, the circuits are separated from each other and from device surfaces that could be touched (secondary circuit with 30 V DC maximum, supplied from mains circuit up to 300 V overvoltage category II).

By means of basic insulation according to EN61010-1, the communication interfaces are separated from each other and from device surfaces that could be touched (secondary circuit with 30 V DC maximum, supplied from mains circuit up to 300 V overvoltage category II). They are designed for network environment 0 according to IEC TR62102.

6.2 Connecting the Modbus TCP ports

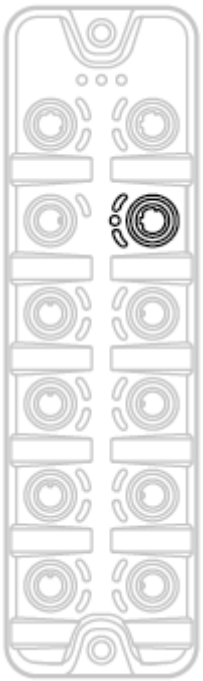
33671



- ► Connect the device via the M12 socket X21 and/or X22 to the Modbus TCP network
  (e.g. Modbus TCP PLC, additional Modbus TCP device)
- ► To connect the devices, use M12 connectors with protection rating IP 65 / IP 66 / IP 67 or higher (→ **Accessories** (→ p. 95)).
- ► Cover the unused sockets with M12 protective caps (art no. E73004).

6.3 Connecting the IoT port

34044



- ► Connect the device via the M12 socket X23 to the IT network (e.g. laptop/PC with LR DEVICE parameter setting software, laptop/PC with LR SMARTOBSERVER monitoring software, laptop/PC with software capable of processing http requests).
- ► To connect the devices, use M12 connectors with protection rating IP 65 / IP 66 / IP 67 or higher (→ **Accessories** (→ p. 95)).
- ► Cover the unused sockets with M12 protective caps (art no. E73004)

6.4 IO-Link ports

51958

The IO-Link ports of the AL1342 meet the requirements of the IO-Link specifications 1.0 to 1.1.2.

► Please note the information concerning IO-Link wiring!

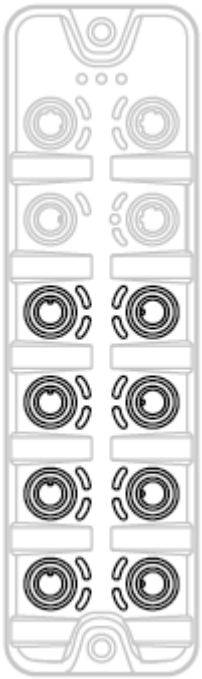
► Cover unused sockets with M12 protective caps (art. no.: E73004).

6.4.1 Connecting IO-Link devices for Class A operation

51959

Wiring information:

- The connected IO-Link devices must be supplied exclusively via the IO-Link master.
- The additional digital inputs IO-Link ports X01...X08 (pin 2) have a type 2 behaviour according to the standard EN61131-2. The connected electronics must be electrically suited for this.



► Connect the connectors of the IO-Link devices with the M12 sockets of the IO-Link ports X01...X08.

  ▪ Maximum cable length per IO-Link port: 20 m

► To connect the devices, use M12 connectors with protection rating IP 65 / IP 66 / IP 67 or higher (→ **Accessories** (→ p. 95)).

6.4.2 Connecting IO-Link devices for Class B operation

51960

Notes on wiring:

- For Class B operation, the IO-Link device must be supplied with an additional auxiliary voltage UA using a Y connection cable.

WARNING

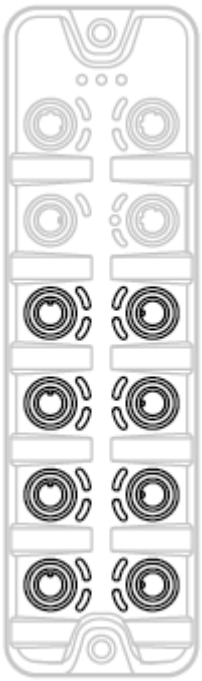
Non-compliance with the electrical separation of the circuits

> Risk of fire!

► Ensure that the external supply UA is galvanically separated from the circuit of the IO-Link Master by assuring basic insulation (according to IEC 61010-1, secondary circuit with 30 V DC maximum, supplied from mains circuit up to 300 V of overvoltage category II).

► Ensure that the IO-Link devices and the connection technology support the galvanic separation.

In case of operation as port class B, the additional digital input of the IO-Link port (pin 2) is not available!



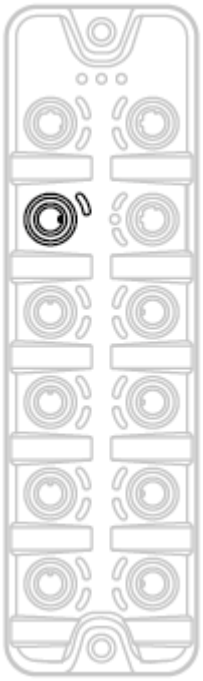
► Connect the connectors of the IO-Link devices via a Y connection cable with the M12 sockets of the IO-Link ports X01...X08.

► Connect the Y cable to 24 V DC (20...30 V SELV/PELV)

► To connect the devices, use M12 connectors with protection rating IP 65 / IP 66 / IP 67 or higher (→ **Accessories** (→ p. 95))!

6.5 Connecting the device

33882



- ► Disconnect power.
- ► Connect the IO-Link Master via M12 socket X31 to 24 V DC (20...30 V SELV/PELV; according to EN61010-1, secondary circuit with maximum 30 V DC supplied by mains circuit up to 300 V of overvoltage category II).
  - Maximum cable length: 25 m
- ► To connect the device, use M12 connectors with protection rating IP 65 / IP 66 / IP 67 or higher (→ **Accessories** (→ p. 95)).

When using cable length greater than 25 m keep in mind the voltage drop as well as the required minimum voltage supply of 20 V!

7 Operating and display elements

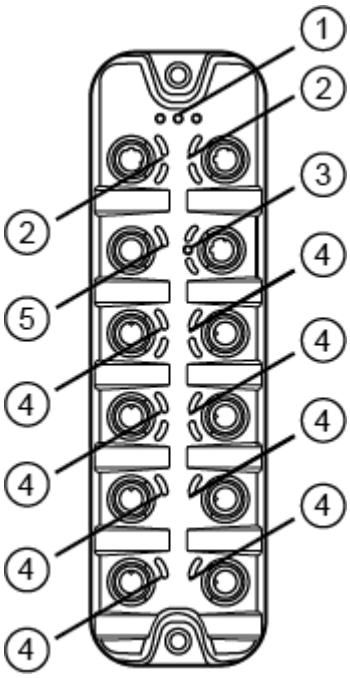
Content

Overview....19
LED indicators....20

34063

7.1 Overview

34356



1. RDY, RUN and ERR status LEDs
   → **Status LEDs** (→ p. 20)
2. LNK and ACT status LEDs of the Modbus TCP interfaces 1 (X21) and 2 (X22)
   → **Ethernet ports** (→ p. 20)
3. LNK, ACT status-LEDs and IoT LED of the IoT interface (X23)
   → **IoT port** (→ p. 21)
4. IOL and DI status-LEDs of the IO-Link port (X01...X08)
   → **IO-Link Ports (Class A)** (→ p. 21)
5. PWR status LED of the voltage supply (X31)
   → **Voltage supply** (→ p. 21)

7.2 LED indicators

34047

The device only has the following LED indicators:

7.2.1 Status LEDs

34436

The RDY LED indicates the status of the gateway.
The RUN LED indicates the current state of the Modbus TCP state machine.
The ERR LED indicates occurring errors.

Status LED			Description
RDY	green	on	Status: OK
		flashes 5 Hz	Status: Error
		flashes (200 ms on, 800 ms off)	Status: Firmware update is running
		off	Status: Gateway not running or gateway booting
ERR	red	on	communication error
		flashes 10 Hz	boot error
		flashes (200 ms on, 200 ms off, 200 ms on, 1000 ms off)	watchdog error (Modbus TCP or process data)
		flashes (200 ms on, 1000 ms off)	local error
		flashes 2.5 Hz	invalid configuration
		off	no error
RUN	green	on	connection established
		flashes 1 Hz	ready, but not yet configured
		flashes 5 Hz	waiting for connection
		off	not ready

7.2.2 Ethernet ports

34348

Each Ethernet port has 2 LEDs (LNK and ACT). The LEDs indicate the status of the Ethernet connection.

Status LED			Description
LNK	green	on	Ethernet connection established
		off	No Ethernet connection
ACT	yellow	flashes	Data is transmitted via the Ethernet interface.
		off	No data transmission

7.2.3 IoT port

34043

The IoT port has the 3 LNK, ACT and IoT LEDs. The LEDs indicate the status of the Ethernet connection and the device identification.

Status LED			Description
LNK	green	on	Ethernet connection established
		off	No Ethernet connection
ACT	yellow	flashes	Data is transmitted via the Ethernet interface.
		off	No data transmission
IoT	green	flashes	Device identification active

7.2.4 Voltage supply

34191

The interface for voltage supply (X31) has the LED that is marked as US. The LED indicates the status of the voltage supply.

Status LED			Description
US	green	on	The supply voltage Us is applied.
		off	No supply voltage is applied or the applied supply voltage is too low.

7.2.5 IO-Link Ports (Class A)

34074

Each IO-Link Port Class A has 2 LEDs labelled IOL and DI. The LEDs indicate the status of the IO-Link port.

Status LED			Description
IOL	yellow	Off	Port configured as DI/DO: pin 4 (C/Q) = OFF
		on	Port configured as DI/DO: pin 4 (C/Q) =ON
	green	flashing 1 Hz	Port configured asIO-Link: no IO-Link device found
		Flashing with 2 Hz	Port configured asIO-Link: Status PREOPERATE
		on	Port configured asIO-Link: Status OPERATE
	red	Flashing with 2 Hz	Port configuration error or short circuit / overload on US
		on	Transmission Error
DI	yellow	Off	Digital input: pin 2 = OFF
		on	Digital input: pin 2 = ON

8 Set-up

52357

When the supply voltage is switched on, the AL1342 starts with the factory settings. The display elements signal the current operating mode (→ **Operating and display elements** (→ p. 19)).

To enable parameter setting of the AL1342, the IoT interface and / or the fieldbus interface must be configured according to the network environment.

► Configure IoT interface (LR DEVICE: → **IoT: Configure IP settings** (→ p. 26) or → **IoT: Configuring IP settings** (→ p. 47)).

► Configure fieldbus interface (LR DEVICE: → **Fieldbus: Configure IP settings** (→ p. 30) or IoT: → **Fieldbus: Configuring the IP settings** (→ p. 50)).

> IoT / fieldbus interface has valid IP settings.

> User can set the parameters of the AL1342.

Further steps:

- Optional: Update firmware of AL1342 (→ **Updating the firmware** (→ p. 93)).
- Set the parameters of the AL1342 (→ **Configuration** (→ p. 23)).

9 Configuration

Content

LR DEVICE....24
ifm IoT Core....38
Modbus TCP....81

33858

9.1 LR DEVICE

Content

Remarks 25
IoT: Configure IP settings 26
IoT: Configure security mode 27
IoT: Configuring access rights 28
IoT: Configure the interface to LR AGENT or LR SMARTOBSERVER 29
Fieldbus: Configure IP settings 30
Fieldbus: set the length of the process data 31
IO-Link ports: Activate data transfer to LR AGENT or LR SMARTOBSERVER 32
IO-Link ports: Configure operating mode 33
IO-Link ports: Set the device validation and data storage 34
IO-Link ports: set fail-safe values 35
Info: Show device information 35
Firmware: Reset device to factory settings 36
Firmware: Reboot the device 36
Configure IO-Link devices 37

33692

On delivery, the AL1342 is configured with the factory settings (→ **Factory settings** (→ p. 94)).

Required software: LR DEVICE (1.5.0.x or higher) (art.-no.: QA0011/QA0012)

9.1.1 Remarks

Content

Offline parameter setting 25
Parameter setting with LR DEVICE.....25

34180

Offline parameter setting

34060

The AL1342 supports the offline parameter setting. In this context, the user creates and stores a configuration for the IO-Link master and the connected IO-Link devices without being connected to the AL1342 (OFFLINE mode). The configuration created in this way can be stored as a file (*.lrp) and loaded to the AL1342 and activated at a later date.



Further information about offline parameter setting: → Operating instructions LR DEVICE

Parameter setting with LR DEVICE

34437

Parameter setting of the AL1342 with the LR DEVICE is only possible via the IoT interface X23.

9.1.2 IoT: Configure IP settings

34049

For access to the IO-Link master via the IT infrastructure the user has to set the IP settings of the IoT port.

To configure the IP settings with DHCP, a DHCP server has to be active in the IT network. If no DHCP server can be reached in the IT network, an IP address is automatically assigned to the IoT port with the Zeroconfig protocol (address range: → **Factory settings** (→ p. 94)).

To configure the IP settings of the IoT interface:

► Select [IoT] menu.

> The menu page shows the current settings.

► Set the following parameters as required:

Name	Description	Possible values	
[DHCP]	Activate/deactivate the DHCP client of the device	[Static IP]	IP settings were set by the user
		[DHCP]	IP settings are set by a DHCP server in the network.
[IP address]*	IP address of the IoT port	Factory setting: 169.254.X.X	
[Subnet mask]*	Subnet mask of the Ethernet network	Factory setting: 255.255.0.0	
[Default gateway IP address]*	IP address of the network gateway	Factory setting: 0.0.0.0	
[MAC address]	MAC address of the IoT port	The value is firmly set.	

* ... can only be edited if parameter [DHCP] = [Static IP]

► Save changed values on the device.

9.1.3 IoT: Configure security mode

54680

The IoT interface of the IO-Link offers a security mode. It enables secure data transmission via transport encryption and restriction of the access to IO-Link masters and IO-Link devices via user authentication.

To configure the security mode:

► Select [IoT] menu.

> The menu page shows the current settings.

► Set the following parameters as required:

Name	Description	Possible values	
[Security mode HTTPS]	Set the security mode	[Disabled]	Security mode disabled
		[Enabled]	Security mode enabled
[Security password]	Password Note: The set password is not displayed.		

► Save changed values on the device.

The security mode only protects the access to the device via the IoT interface.
The user name "administrator" cannot be changed.

The security mode can be enabled without setting the password. During the attempt to write to the device, LR DEVICE requires to enter and confirm the password.

After entering the password, the user has unrestricted access to IO-Link masters and connected IO-Link devices. The password will only be requested again if the current LR DEVICE session is over (e. g. after restarting the LR DEVICE).

To change the set password:

► Sign in with a valid password.

► Enter the new password in the field [Security password].

► Write changes to the device.

> The new password is set.

9.1.4 IoT: Configuring access rights

34046

The access rights define which instance may read and / or write the parameter data, process data and event/diagnostic messages.

In order to configure the access rights to the IO-Link master:

► Select [IoT] menu.

> The menu page shows the current settings.

► Set the following parameters as required:

Name	Description	Possible values	
[Access Rights]	The access rights to the parameter data, process data and the event/diagnostic messages of the IO-Link master as well as the connected IO-Link devices	[Modbus TCP + IoT]*	▪ Modbus TCP and IoT Core have read and write access rights to parameters and process data ▪ Modbus TCP and IoT Core have read access rights to events/alarms
		[Modbus TCP + IoT (read-only)]	▪ Modbus TCP has read and write access rights to parameters and process data ▪ Modbus TCP has read access rights to events/alarms ▪ IoT Core only has read access rights to parameters, process data and events/alarms
		[IoT only]	▪ IoT Core has read and write access rights to parameters and process data ▪ IoT has read access rights to events/alarms ▪ Modbus TCP has no access rights

* ... Factory setting

► Save changed values on the device.

If in LR DEVICE and Modbus TCP projection software the parameter [Access Rights] is = [Modbus TCP + IoT], the parameter values set in the Modbus TCP projection software will always apply.

If the parameter [Access Rights] in LR DEVICE is = [IoT only], set the parameter [Access Rights] = [Keep settings] in the Modbus TCP projection software.

If the parameter [Access Rigts] in LR DEVICE is = [<Fieldbus> + IoT (read-only)], write access to the device configuration via LR DEVICE and IoT core services is blocked. To enable write access again, set the parameter to [<Fieldbus> + IoT] via fieldbus configuration software.

Changes of the parameter [Access Rights] will only be effective after restarting the IO-Link master (→ **Firmware: Reboot the device** (→ p. 36)).

9.1.5 IoT: Configure the interface to LR AGENT or LR SMARTOBSERVER

34048

To enable transfer of process data from the IO-Link master to LR AGENT or LR SMARTOBSERVER, the interface has to be configured accordingly.

► Select [IoT] menu.

> The menu page shows the current settings.

► Set the following parameters as required:

Name	Description	Possible values	
[IP address LR Agent or SMARTOBSERVER]	IP address of LR AGENT or LR SMARTOBSERVER	Factory setting: 255.255.255.255	
[Port LR Agent or SMARTOBSERVER]	Port number that is used to send process data to LR AGENT or LR SMARTOBSERVER	0 ... 65535	Factory setting:: 35100
[Interval LR Agent or SMARTOBSERVER]	Cycle time for the transfer of the process data to LR AGENT or LR SMARTOBSERVER (value in milliseconds)	[Off]	no transfer
		500 ... 2147483647	500 ms ... 2147483647 ms
[Application Tag]	Source identifier of the IO-Link master in the structure of LR AGENT or LR SMARTOBSERVER (String32)	Factory setting: AL1342	

After changing the parameter [Port LR Agent or SMARTOBSERVER] or [Application Tag], it may take 120 seconds before the device establishes a new TCP connection.

To prevent the delay:

► Reboot the device after changing the the parameter.

► Save changed values on the device.

9.1.6 Fieldbus: Configure IP settings

54698

The configuration of the IP settings of the fieldbus port is only possible via LR DEVICE and IoT.

To configure the IP settings of the Modbus TCP interface:

► Select [Fieldbus] menu.

> The menu page shows the current settings.

► Set the following parameters as required:

Name	Description	Possible values	
[DHCP]	Activate / deactivate the DHCP client of the device	[Static IP]	IP parameters are set by the user
		[DHCP]	IP parameters are set by a DHCP server in the network.
		[BOOTP]	IP parameters are set via the Bootstrap Protocol (BOOTP)
[IP address]*	IP address of the Modbus TCP interface	Factory setting:: 192.168.1.250	
[Subnet mask]*	Subnet mask of the IP network	Factory setting: 255.255.255.0	
[Default gateway IP address]*	IP address of the gateway	Factory setting: 0.0.0.0	
[MAC address]	MAC address of the Modbus TCP interface	The value is firmly set.	
[Fieldbus firmware]	Firmware version of the Modbus TCP stack	e.g. 2.6.0.5	
[connectiontimeout]	Max. value for Connection Timeout (value in milliseconds)	1 ... 300000	

* ... Parameter nur editierbar, wenn Parameter [DHCP] = [Static IP]

► Save changed values on the device.

9.1.7 Fieldbus: set the length of the process data

54681

To set the length of the process data to be transmitted and the arrangement of the bytes:

► Select [Fieldbus] menu.

> The menu page shows the current settings.

► Set the following parameters as required:

Name	Description	Possible values	
[Process data length]	Length of the process input data and process output data per IO-Link port	2 bytes input 2 bytes output	2 bytes input data, 2 bytes output data
		4 bytes input 4 bytes output	4 bytes input data, 4 bytes output data
		8 bytes input 8 bytes output	8 bytes input data, 8 bytes output data
		16 bytes input 16 bytes output	16 bytes input data, 16 bytes output data
		32 bytes input 32 bytes output	32 bytes input data, 32 bytes output data
[Swap]	Arrangement of the bytes in process data	off	as Array of Bytes
		on	as Integer16 value; during an update of the process data, bytes are exchanged word by word (input data and output data)

► Save changed values on the device.

9.1.8 IO-Link ports: Activate data transfer to LR AGENT or LR SMARTOBSERVER

33690

The user can decide separately for each IO-Link port whether the process data of the connected IO-Link devices should be transferred to LR AGENT or LR SMARTOBSERVER.

To transfer process data the interface to the LR AGENT or LR SMARTOBSERVER has to be correctly configured (→ **IoT: Configure the interface to LR AGENT or LR SMARTOBSERVER** (→ p. 29)).

To activate / deactivate data transfer:

► Select [Port x] menu (x = 1...8).

> The menu page shows the current settings.

► Set the following parameters as required:

Name	Description	Possible values	
[Transmission to LR Agent or SMARTOBSERVER]	Transfer of process data of the connected IO-Link device to LR AGENT oder LR SMARTOBSERVER	[Disabled]	Transfer process data
		[Enabled]	Don't transfer process data

► Save changed values on the device.

9.1.9 IO-Link ports: Configure operating mode

33694

The IO-Link ports X01...X08 of the device support the following operating modes:

- Disabled: no data transfer at pin 4 (C/Q) of the IO-Link port
- Digital input (DI): binary input signal at pin 4 (C/Q) of the IO-Link port
- Digital output (DO): binary output signal at pin 4 (C/Q) of the IO-Link port
- IO-Link: IO-Link data transfer via pin 4 (C/Q) of the IO-Link port

The user can set the operating mode separately for each IO-Link port.
To set the operating mode of an IO-Link port:

► Select [Port x] menu (x = 1...8).

> The menu page shows the current settings.

► Set the following parameters as required:

Name	Description	Possible values	
[Mode Pin4 US]	Operating mode of pin 4 of the IO-Link port	[Disabled]	Port deactivated
		[DI]	Operation as digital input
		[DO]	Operation as digital output
		[IO-Link]	Operation as IO-Link interface
[Cycle time actual]**	Current cycle time of the data transfer between IO-Link master and IO-Link device on the port (value in microseconds)	Parameter can only be read	
[Cycle time preset]*	Cycle time of the data transfer between the IO-Link master and the IO-Link device at the port (value in microseconds)	0	The device automatically sets the fastest possible cycle time.
		1 ... 132800	1 microsecond ... 132800 microseconds
[Bitrate]**	Current transmission rate of the data transfer between the IO-Link master and the IO-Link device on the port	Parameter can only be read	

* ... Parameter only available if [Mode] = [IO-Link]
** ... Parameter only visible if the IO-Link device is connected to the IO-Link port.

► Save changed values on the device.

9.1.10 IO-Link ports: Set the device validation and data storage

33697

The user can choose how the IO-Link ports are to behave with regard to the device validation and the storage / recovery of parameter data of the connected IO-Link device.

The following options are available:

Option	Validation of the IO-Link device	Storage of the parameter values	Recovery of the parameter values
[No check and clear]	no	no	no
[Type compatible V1.0 device]	yes, test the compatibility with IO-Link standard V1.0	no	no
[Type compatible V1.1 device]	yes, test the compatibility with IO-Link standard V1.1	no	no
[Type compatible V1.1 device with Backup + Restore]	yes, test the compatibility with IO-Link standard V1.1 and identity of design (vendor ID and device ID)	yes, automatic storage of the parameter values; changes of the current parameter values will be stored	yes, recovery of the parameter values when connecting an identical IO-Link device with factory settings
[Type compatible V1.1 device with Restore]	yes, test the compatibility with IO-Link standard V1.1 and identity of design (vendor ID and device ID)	no, there is no automatic storage changes of the current parameter values will not be stored	yes, recovery of the parameter values when connecting an identical IO-Link device with factory settings

The options only apply if the IO-Link port is in the operating mode "IO-Link".

For options [Type compatible V1.1 device with Backup + Restore] and [Type compatible V1.1 device with Restore]: If the vendor ID and device ID are changed in the online mode, the data memory will be deleted and a new backup of the parameter values of the connected IO-Link device will be created in the IO-Link master.

To configure the device validation and the data storage:

► select [Port x] menu (x = 1...8).

> The menu page shows the current settings.

► Set the following parameters as required:

Name	Description	Possible values	
[Validation / Data Storage]	Supported IO-Link standard and behaviour of the IO-Link master when connecting a new IO-Link device at port x (x = 1...8)	[No check and clear]	
		[Type compatible V1.0 device]	
		[Type compatible V1.1 device]	
		[Type compatible V1.1 device with Backup + Restore]	
		[Type compatible V1.1 device with Restore]	
[Vendor ID]	ID of the manufacturer that is to be validated	0...65535	Factory setting: 0# ifm electronic: 310
[Device ID]	ID of the IO-Link device that is to be validated	0...16777215	Factory setting: 0

► Save changed values on the device.

9.1.11 IO-Link ports: set fail-safe values

34459

In case the Modbus TCP connection is interrupted, fail-safe values can be assigned to the outputs of the IO-Link ports.

To set the fail-safe values of the IO-Link ports:

► Select [Port x] menu (x = 1...8).

> The menu page shows the current settings.

► Set the following parameters as required:

Name	Description	Possible values	
[Fail-safe digital out]	Fail-safe values for output (operating mode "DO")	Reset	Reset value (LOW)
		Old	hold old value
		Set	Set value (HIGH)
[Fail-safe IO-Link]	Fail-safe value for output (operating mode "IO-Link")	Off	no fail-safe value
		Reset	reset value
		Old	hold old value
		Pattern	provide sample

► Save changed values on the device.

9.1.12 Info: Show device information

34065

To read the general information of the ifm IO-Link master:

► Select [Info] menu.

> The menu page shows the current settings.

Name	Description	Possible values
[Product code]	Article number of the IO-Link master	AL1342
[Device family]	Device family of the IO-Link master	IO-Link master
[Vendor]	Vendor	ifm electronic gmbh
[SW-Revision]	Firmware of the IO-Link master	
[HW revision]	Hardware version of the IO-Link master	
[Bootloader revision]	Bootloader version of the IO-Link master	
[Serial number]	Serial number	

9.1.13 Firmware: Reset device to factory settings

33838

When the IO-Link master is reset, all parameters are set to the factory settings:
To reset the device to factory settings:

► Select [Firmware] menu.
> The menu page shows the current settings.
► Click on [Factory Reset] to reset the device.
> LR DEVICE sets the device to the factory settings.

9.1.14 Firmware: Reboot the device

33832

When rebooting the device, all settings are kept.
To restart the AL1342:

► Select [Firmware] menu.
> The menu page shows the current settings.
► Click on [Reboot] to reboot the device.
> LR DEVICE reboots the ifm IO-Link master.

9.1.15 Configure IO-Link devices

33856

To configure the IO-Link devices connected to the device with the LR DEVICEparameter setting software:

Requirements:

> IO-Link master is correctly installed and connected to the LR DEVICE parameter setting software.

> The IO-Link device is connected correctly with the AL1342.

> Operating mode of the IO-Link port is "IO-Link" (→ **IO-Link ports: Configure operating mode** (→ p. 33)).

> IoT has write access rights to the IO-Link master (→ **IoT: Configuring access rights** (→ p. 28)).

1 Select IO-Link master

► Start LR DEVICE.

► Update IODD file library
OR:
Import IODD file of the IO-Link device manually.

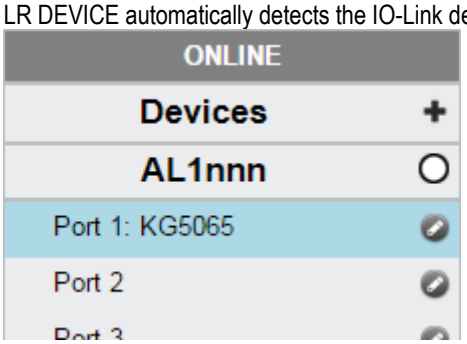
► Scan network for devices.

> LR DEVICE detects IO-Link master.

2 Add IO-Link device

► Under [ONLINE]: Click on the required IO-Link master.

> LR DEVICE automatically detects the IO-Link devices connected to the IO-Link master (e.g. ifm sensor KG5065).



3 Configure IO-Link device

► Mouse click on the port to which the IO-Link device is connected.

> LR DEVICE reads and shows the current parameter values of the IO-Link device.

► Configure IO-Link device.

Information about the available parameters of the IO-Link device: → IO Device Description (IODD) des IO-Link Devices

► Save the changed configuration on the IO-Link device.

9.2 ifm IoT Core

Content

Programmers' notes 39
First steps 43
General functions 43
IoT: Configuring access rights 47
IoT: Configuring IP settings 47
IoT: Configuring the LR AGENT or LR SMARTOBSERVER interface 48
IoT: Configuring security mode 48
Fieldbus: Configuring the IP settings 50
Fieldbus: Confguring the fieldbus interface 51
Fieldbus: Setting fail-safe values 52
IO-Link ports: Setting the operating mode of pin 4 (US) 52
IO-Link ports: Configuring device validation and data storage 53
IO-Link ports: Configuring data transfer to LR AGENT or LR SMARTOBSERVER 55
IO-Link ports: Reading / writing process data 55
IO-Link ports: Indicating port events 58
IO-Link devices: Accessing parameters 58
IO-Link devices: Reading an writing device information 59
IO-Link devices: Indicating IO-Link events 60
Gateway: Resetting, rebooting and localising the device 60
Gateway: Reading device information 60
Gateway: Reading status and diagnostic information 60
Gateway: Updating the firmware 61
Gateway: Setting the application tag 63
Subscribing to notifications 64
Using Web Socket 68
MQTT support 70
Using the IoT-Core Visualizer 74

52244

General notes on the ifm IoT Core: → **Programmers' notes** (→ p. 39)

9.2.1 Programmers' notes

Content

IoT Core: General information 39
Access the ifm IoT Core 40
IoT Core: Diagnostic codes 42

34229

IoT Core: General information

52256

The DataLine device family has an IoT Core. The IoT Core allows the user to address the AL1342 from IT networks via a REST API and to integrate it into Internet-of-Things applications.

A device description is stored on the AL1342. This device description is a structured, machine-readable data object in JSON format. All current values of parameters, process data, diagnostic data and device information are mapped in this data object. These data values can be read and changed by means of services.

Access the ifm IoT Core

52257

The user can access the ifm IoT Core via HTTP requests. The following request methods are available.

GET request

33804

Using the GET method the user has read access to a data point.

The syntax of the request to the IoT Core is:

```
http://ip/datapoint/service
```

Parameter	Description
ip	IP address of the IoT interface
data_point	Data point which is to be accessed
service	Service

The syntax of the return of the IoT Core is:

```
{
"cid":id,
"data":{"value":resp_data},
"code":diag_code
}
```

Parameter	Description
id	Correlation ID for the assignment of request and return
resp_data	Value of the data point; depending on the data type of the data point
diag_code	Diagnostic code (→ **IoT Core: Diagnostic codes** (→ p. 42))

Example: GET request

54033

Request (via browser):

```
http://192.168.0.250/devicetag/applicationtag/getdata
```

Response:

```
{
"cid":-1,
"data":{"value":"AL1342"},
"code":200
}
```

POST request

54700

Using a POST request the user has read and write access to a data point.
The syntax of the request to the IoT Core is:

```
{
"code":"code_id",
"cid":id,
"adr":"data_point/service",
"data":{req_data},
"auth":{"user":"usr_id","passwd":"password"}
}
```

Field	Parameter	Description	
code	code_id	Service class	
		▪ request	Request
		▪ transaction	Transaction
		▪ event	Event
cid	id	Correlation ID for the assignment of request and response; ID freely assignable by the user	
adr	data_point	Data point of the element tree which is to be accessed	
	service	Service to be performed (→ **Overview: IoT services** (→ p. 136))	
data*	req_data	Data to be transferred to the IoT Core (e.g. new values); syntax depending on the service	
auth**	usr_id	user name (base64 coded); default value: administrator	
	password	password (base64 coded)	

* = optional; only required for services, that submit data to the IoT core (e. g. setdata)
** = optional; only required, if security mode is activated

The syntax of the return of the IoT Core is:

```
{
"cid":id,
"data":{resp_data},
"code":diag_code
}
```

Field	Parameter	Description
cid	id	Correlation ID for the assignment of request and response (see request)
data*	resp_data	Value of the data point; syntax depending on the service
code	diag_code	Diagnostic code (→ **IoT Core: Diagnostic codes** (→ p. 42))

* = optional; only required for services, that receive data from the IoT core (e.g. getdata)

Example: POST request

54035

Request:

```
{
"code":"request",
"cid":4711,
"adr":"devicetag/applicationtag/getdata"
}
```

Response:


```
{
"cid":4711,
"data":{"value":"AL1342"},
"code":200
}
```


IoT Core: Diagnostic codes

54688

Code	Text	Description
200	OK	Request successfully processed
230	OK but needs reboot	Request successfully processed; IO-Link master must be restarted
231	OK but block request not finished	Request successfully processed; blockwise request, but not yet finished
232	Data has been accepted, but internally modified	New values have been accepted, but were adjusted by the IO-Link master (Master cycle time)
233	IP settings (of IoT-Port) have been updated. Application needs to reload device. Wait at least 1 second before reloading device.	IP settings have been successfully changed, IO-Link master will be reloaded; wait for at least 1 second
400	Bad request	Invalid request
401	Unauthorized	Non authorised request
403	Forbidden	Forbidden request
500	Internal Server Error	Internal fault
503	Service Unavailable	The service is not available (e. g. IO-Link port in wrong operating mode; no IO-Link device at IO-Link port)
530	The requested data is invalid	Invalid process data
531	IO-Link error	Error in IO-Link Master / device
532	PLC connected Error	Error while setting data, because IO-Link master is still connected to fieldbus PLC

9.2.2 First steps

52245

To read the device description of the AL1342:

► Send the following POST request to the AL1342:
`{"code":"request","cid":-1,"adr":"gettree"}`

> AL1342 returns the device description as structured JSON object.

► Identify all substructures and the data points contained therein in the tree structure of the JSON object.

► Identify the applicable services for the access to substructures and the data points contained therein.

9.2.3 General functions

61148

The AL1342 has the type `device` (→ **Overview: IoT types** (→ p. 135)).

The following services can be used on the root element of the type `device`:

Service	Description
../gettree	Provide the complete tree or subtree of the device description (JSON)
../getidentity	Reading device information
../getdatamulti	Reading several parameter values sequentially
../getelementinfo	Reading detailed information of an element
../getsubscriberlist	Print a list of all active notification subscriptions
../querytree	Search device description for specific elements

Depending on the read and write access rights, the following services can be applied to elements of type `data`:

Service	Description
../getdata	Reading the value of the element
../setdata	Write the value of the element

Example: Reading properties of an element

59782

Task: Determine the data type and value range of the accessrights parameter.

Solution: Read the properties of the element `iotsetup/accessrights` of the getelementinfo service. The fields type (data type) and valuation (range of values) contain the required information.

- Request:

```
{
"code":"request",
"cid":4711,
"adr":"getelementinfo",
"data":{"adr":"iotsetup/accessrights"}
}
```

- Response:

```
{
"cid":4711,
"data":{
"identifier":"accessrights",
```

```
"type":"data",
"uid":null,
"profiles":["parameter"],
"format":{
"type":"enum",
"namespace":"json",
"encoding":"integer",
"valuation":{
"valuelist":{
"0":"Fieldbus + IoT",
"1":"Fieldbus + IoT (read-only)",
"3":"IoT only"}}}},
"code":200
}
```


The accessrights parameter has the data type ENUM with the valid values "Fieldbus + IoT", "Fieldbus + IoT (read only)" and "IoT only".

Example: output subtree

61149

Task: Output all direct sub-elements of the node `firmware`.

Solution: Use the service gettree to output the required subtree (root node: firmware, sub-levels to be shown: 1)

- Request:


```
{
"code":"request",
"cid":4711,
"adr":"gettree",
"data":{
"adr":"firmware",
"level":1}
}
```


- Response:


```
{
"cid":4711,
"data":{
"identifier":"firmware",
"type":"structure",
"profiles":[
"software","software/uploadablesoftware"],
"subs":[
{
"identifier":"version","type":"data","profiles":["parameter"],
"format":{"type":"string","namespace":"json","encoding":"UTF-8"}},
{
"identifier":"type","type":"data",
"format":{"type":"string","namespace":"json","encoding":"UTF-8"}},
{
"identifier":"install","type":"service"},
{
"identifier":"factoryreset","type":"service"},
{
"identifier":"signal","type":"service"},
{
"identifier":"container","type":"data",
"format":{"type":"binary","namespace":"json","encoding":"base64"}},
```

```
{
"identifier":"reboot","type":"service"}]
},
"code":200
}
```

Example: Read several parameter values of the IO-Link master simultaneously

33840

Task: The following current values are to be read by the IO-Link master: temperature, serial number

Solution: Read the current parameter values using the getdatamulti service (data point temperature: /processdatamaster/temperature; data point serial number: /deviceinfo/serialnumber)

- Request:

```
{
"code":"request",
"cid":4711,
"adr":"/getdatamulti",
"data":{"datatosend":["/processdatamaster/temperature","/deviceinfo/serialnumber"]
}
}
```

- Response:

```
{
"cid":4711,
"data":{"processdatamaster/temperature":{"code":200,"data":44},
"deviceinfo/serialnumber":{"code":200,"data":"000174210147"}},
"code":200
}
```

Example: Browsing device description

61150

Task: List all elements with the designation "status" and the profile "runcontrol".

Solution: Use the service querytree to browse the device description with the parameters "status" (name) and "runcorntrol" (profile)

- Request:

```
{
"cid":4711,
"adr":"querytree",
"code":"request",
"data":{
"profile":"runcontrol",
"name":"status"}
}
```

- Response:

```
{
"cid":4711,
"data":{
"adrList":[
"device/connections/mqttConnection/status",
"device/connections/mqttConnection/mqttCmdChannel/status"]},
"code":200
}
```

Setting the storage duration

61153

The IoT Core offers the possibility to set the storage duration of data and notifications. The Services **Service: setdata** (→ p. 146) and **Service: subscribe** (→ p. 148) therefore have the parameter "duration".

Example: Subscribing to notifications

61154

Task: The current values of the following parameters are to be sent regularly to a network server with IP address 192.168.0.4:

- Product name of the IO-Link Devices an IO-Link port X02
- Cyclic input data of the IO-Link Devices an IO-Link port X02
- Operating temperature of the IO-Link master.

The subscription is only to be active until the next restart of the IO-Link master.

Solution: Subscribe to the required data using the subscribe service.

- Request:

```
{
"code":"request",
"cid":4711,
"adr":"/timer[1]/counter/datachanged/subscribe",
"data":{
"callback":"http://192.168.0.4:80/temp",
"datatosend":[
"/iolinkmaster/port[2]/iolinkdevice/productname",
"/iolinkmaster/port[2]/iolinkdevice/pdin",
"/processdatamaster/temperature"],
"duration":"uptime"}
}
```

- Response:

```
{
"cid":4711,
"code":200
}
```

9.2.4 IoT: Configuring access rights

59785

Substructure: `iotsetup`

Available data points:

Name	Description	Access
../accessrights	Access rights to the IO-Link master	rw

rw ... read and write

If in IoT and Modbus TCP projection software the parameter [Access Rights] is = [Modbus TCP + IoT], the parameter values set in the Modbus TCP projection software will always apply.

If in IoT the parameter [Access Rights] is = [IoT only], set the parameter [Access Rights] = [Keep settings] in the Modbus TCP projection software.

If in LR DEVICE the parameter [Access Rigts] is = [EtherCAT + IoT (read-only)], write access to the device configuration via LR DEVICE and IoT core services is blocked. To enable write access again, set the parameter to [EtherCAT + IoT] via fieldbus configuration software.

Changes of the parameter [Access Rights] will only be effective after restarting the IO-Link master (→ **Firmware: Reboot the device** (→ p. 36)).

9.2.5 IoT: Configuring IP settings

61155

Substructure: `iotsetup`

Available data points:

Name	Description	Access
../network/dhcp	Configuration of the IP settings of the IoT port	rw
../network/ipaddress	IP address of the IoT port	rw
../network/subnetmask	Subnet mask of the network segment	rw
../network/ipdefaultgateway	IP address of the network gateway	rw

rw ... read and write

Applicable services:

Name	Description
../network/setblock	Write all values of the substructure blockwise

Change the IP parameters in the substructure network only blockwise with the setblock service!

9.2.6 IoT: Configuring the LR AGENT or LR SMARTOBSERVER interface

59786

Substructure: `iotsetup`

Avalable data points:

Name	Description	Access
../smobip	IP address of the LR SMARTOBSERVER	rw
../smobport	Port number of the LR SMARTOBSERVER	rw
../smobinterval	Cycle time for data transmission to LR SMARTOBERVER (value in milliseconds)	rw

rw ... read and write

9.2.7 IoT: Configuring security mode

54683

The access to the IoT interface of the IO-Link master can be protected with a security mode:

Substructure: `iotsetup`

Available data points:

Name	Description	Access
../security/securitymode	active security mode	rw
../security/password	Password for authentication (Base64 coded)	w

rw ... read and write
w ... write only

Valid character set for the Base64 coding / decoding of the password: UTF-8
Online tool for coding / decoding: → www.base64encode.org)

Note: Security mode

54684

The security mode enables restricting access to the IO-Link master and the connected IO-Link devices from the IT network. In the activated security mode, the following restrictions apply:

- Access only with authentication (password-protected user account)
- Access only via secure https connection (Transport Layer Security - TLS)

The security mode only protects the access to the device via the IoT interface.
The standard value for users is: administrator
The set password cannot be read with getdata.

The current status of the security function can be read with the getidentity service (→ **Servicet: getidentity** (→ p. 139)).

For the authentication, the user must additionally provide the POST requests with a valid user name and password in the field "auth". The user name and the password will be shown as Base64-coded character strings (→ **Example: Request with authentication** (→ p. 50)).

The following requests can be done if the security mode is enabled, also without authentication:

- /getidentity
- /deviceinfo/vendor/getdata
- /deviceinfo/productcode/getdata

Example: Activate security mode

54701

Task: Activate the security mode of the IO-Link interface of the IO-Link master. Set the password "password" (Base64 coded: cGFzc3dvcmQ=)

Solution: The activation sonsists of 2 steps:

1 Activate security mode

Use service setdata with datapoint iotsetup/security/securitymode to activate the security mode.

- Request:

```
{
"code":"request",
"cid":-1,
"adr":"/iotsetup/security/securitymode/setdata",
"data":{"newvalue":"1"}
}
```

- Response:

```
{
"cid":-1,
"code":200
}
```

2 Set required password

Use service setdata with data point iotsetup/security/password to set the required password.

- Request:

```
{
"code":"request",
"cid":-1,
"adr":"/iotsetup/security/password/setdata",
```

```
"data":{"newvalue":"cGFzc3dvcmQ="}
}
```

- Response:

```
{
"cid":-1,
"code":200
}
```

Example: Request with authentication

54685

Task: The temperature of the IO-Link master is to be read. The security function is enabled (current password: password).

Solution: Read the data point processdatamaster/temperature with the getdata service. The request must be sent using https. The user name and the password are transferred as a Base64-coded character string ("administrator" = "YWRtaW5pc3RyYXRvcg==", "password" = "cGFzc3dvcmQ=")

- Request:

```
{
"code":"request",
"cid":-1,
"adr":"processdatamaster/temperature/getdata",
"auth":{"user":"YWRtaW5pc3RyYXRvcg==","passwd":"cGFzc3dvcmQ="}
}
```

- Response:

```
{
"cid":-1,
"data":{"value":37},
"code":200
}
```

Example: reset password

54686

Task: The existing password is to be reset.

Solution: To reset a password, disable the security mode. To disable it, enter the user name and the password (the fields "user" and "passwd").

- Request:

```
{
"code":"request",
"cid":-1,
"adr":"iotsetup/security/securitymode/setdata",
"data":{"newvalue":0},
"auth":{"user":"YWRtaW5pc3RyYXRvcg==","passwd":"SW9UNGlmbQ=="}
}
```

- Response:

```
{
"cid":-1,
"code":200
}
```

9.2.8 Fieldbus: Configuring the IP settings

61435

Substructure: `fieldbussetup`

Available data points:

Name	Description	Access
../fieldbusfirmware	Firmware version of the IO-Link master	r
../network/macaddress	MAC address of the fieldbus interface	r
../network/ipaddress	IP address of the fieldbus interface	rw*
../network/subnetmask	Subnet mask of the network segment	rw*
../network/ipdefaultgateway	IP address of the network gateway	rw*
../network/dhcp	Activate / deactivate the DHCP client	rw
../connectionstatus	Status of the connection to the Modbus TCP network	r

r ... read only
rw ... read and write
* ... only editable, if the Modbus TCP plc ist not in RUNNING state

Applicable services:

Name	Description
../network/setblock	write all values of substructur at once

Change the IP parameters in the substructure network only blockwise by using the service setblock!

9.2.9 Fieldbus: Confguring the fieldbus interface

61440

Substructure: `fieldbussetup/configuration`

Available data points:

Name	Description	Access
../processdataconfiguration	Length of the process data	rw*
../connectiontimeout	max. value for fieldbus connection timeout	rw*
../swap	Byte order of process data	rw*

rw ... read and write
* ... only editable, if the Modbus TCP plc ist not in RUNNING state

9.2.10 Fieldbus: Setting fail-safe values

61441

Substructure: `fieldbussetup/configuration/port[n]` (n = 1...8)

Available data points:

Name	Description	Access
../failsafedigitalout	Fail-safe value for the digital output - pin 4 (DO)	rw*
../failsafeiolink	Fail-safe value for output data IO-Link - pin 4 (IO-Link)	rw*

rw ... read and write
* ... only editable, if the Modbus TCP plc ist not in RUNNING state

9.2.11 IO-Link ports: Setting the operating mode of pin 4 (US)

59793

Substructure: `iolinkmaster/port[n]` (n = 1...8).

Available data points:

Name	Description	Access
../mode	Operating mode of the IO-Link port	rw*
../mastercycletime_preset	Cycle time of the data transfer at the IO-Link port (value in ms)	rw*
../mastercycletime_actual	Current cycle time of the data transfer at the IO-Link port (value in ms)	r
../comspeed	Data transfer rate of the IO-Link port	r

r ... read only
rw ... read and write
* ... only changeable, if the <Feldsbus> plc is not in RUNNING state

9.2.12 IO-Link ports: Configuring device validation and data storage

59792

Substructure: `iolinkmaster/port[n]` (n = 1...8).

Available data points:

Name	Description	Access
../validation_datastorage_mode	Response of the IO-Link port when a new IO-Link device is connected	rw*
../validation_vendorid	IO-Link ID of the manufacturer that is to be validated	rw*
../validation_deviceid	IO-Link ID of the device that is to be validated	rw*
../datastorage	Structure for port data storage	rw
../datastorage/maxsize	Maximum size of the data storage content (in bytes)	r
../datastorage/chunksize	Size of a data segment (in bytes)	r
../datastorage/size	Size of the data storage content (in bytes)	r

r ... read only
rw ... read and write
* ... can only be changed if the Modbus TCP PLC is not in RUNNING state

Applicable services:

Service	Description
../validation_useconnecteddevice	Validate the IO-Link device connected to the IO-Link port*
../datastorage/getblobdata	Reading the content of the data storage area
../datastorage/stream_set	Transfer an individual data segment*
../datastorage/start_stream_set	Start sequential transmission of several data segments*

* ... can only be changed if the Modbus TCP PLC is not in the RUNNING state

Example: Clone the Data Storage of an IO-Link port

52344

Task: Save the Data Storage of IO-Link port X02 of IO-Link master 1 and restore the data at IO-Link master 2.

Solution: The cloning process consists of 2 steps. In the first step, the Data Storage of the IO-Link port of IO-Link master 1 is saved. In the second step, the saved data is restored at the Data Storage of port IO-Link port of IO-Link master 2.

Save Data Storage:

1 Preparations

- ► Read size of segments of Data Storage (h = number of bytes):
  {"code":"request", "cid": -1,"adr":"/iolinkmaster/port[2]/datastorage/chunksize/getdata"}
  Example: h = 256
- ► Read total size of Data Storage area (g = number of bytes):
  {"code": "request", "cid": -1, "adr": "/iolinkmaster/port[2]/datastorage/size/getdata"}
  Example: g = 550
- ► Calcuate the number of reading steps n: n = first integer value to which the following applies: g < n*h
  Example: n= 3, because 550 < 3*256

2 Read Data Storage of IO-Link port

- ► Read Data Storage segment by segment ("pos" is the byte offset, at which the reading process with length "length" starts).
  {"code": "request", "cid": -1, "adr": "/iolinkmaster/port[2]/datastorage/getblobdata", "data": {"pos": 0, "length": h}}
  {"code": "request", "cid": -1, "adr": "/iolinkmaster/port[2]/datastorage/getblobdata", "data": {"pos": h, "length": h}}
  {"code": "request", "cid": -1, "adr": "/iolinkmaster/port[2]/datastorage/getblobdata", "data": {"pos": 2*h, "length": h}}
  ...

{"code": "request", "cid": -1, "adr": "/iolinkmaster/port[2]/datastorage/getblobdata", "data": {"pos": n*h, "length": h}}
Example:
1st read request: pos = 0, length = 256
2nd read resquest: pos = 256, length =256
3rd read request: pos = 512, length = 256

> Each segment value will be returned as BASE64 coded string.

► Join segments.

Restore Data Storage:

1 Preparations

► Determine the size of the saved Data Storage value (n = number of bytes).
Example: n = 550

► Read size of segments (s = number of bytes):
{"code":"request", "cid": -1,"adr":"/iolinkmaster/port[1]/datastorage/chunksize/getdata"}
Example: s = 256

2 Transfer Data Storage strings

► Start transfer of Data Storage string ("size" = size of Data Storage string):
{"code":"request", "cid": -1, "adr":"/iolinkmaster/port[1]/datastorage/start_stream_set", "data": {"size": n}}
Example: size = 550

► Transfer Data Storage string segment by segment ("value" = string value of length s):
{"code": "request", "cid": -1, "adr": "/iolinkmaster/port[1]/datastorage/stream_set", "data": {"value": "aWZtfgIAAABBTDF4NXhfY25faXRfdDIuMi43Nw..."}

9.2.13 IO-Link ports: Configuring data transfer to LR AGENT or LR SMARTOBSERVER

59795

Substructure: `iolinkmaster/port[n]` (n = 1...8).

Available data points:

Name	Description	Access
../senddatatosmob	Process data to LR AGENT or LR SMARTOBSERVER	rw

rw ... read and write

9.2.14 IO-Link ports: Reading / writing process data

61156

Substructure: `iolinkmaster/port[n]` (n = 1...8)

Available data points:

Name	Description	Access
../pin2in	Value of the digital input on pin 2 of the IO-Link port	r
../iolinkdevice/pdin	Value of the IO-Link input on pin 4 of the IO-Link port	r
../iolinkdevice/pdout	Value of the IO-Link output on pin 4 of the IO-Link port	rw*

r ... read only
rw ... read and write
*... can only be changed if the fieldbus PLC is not in RUNNING state

Example: Read IO-Link process data (operating mode "IO-Link")

33842

Task: Read the current measured value of the ifm temperature sensor TN2531 at IO-Link port X02
Solution: Read the data point for the process input data with the `getdata` service.

- Request:

```
{
"code":"request",
"cid":4711,
"adr":"/iolinkmaster/port[2]/iolinkdevice/pdin/getdata"
}
```

- Response:

```
{
"cid":4711,
"data":{"value": "03C9"},
"code":200
}
```

The return value is given in hexadecimal format. Besides the temperature value the return value comprises additional information (→ IO Device Description (IODD) of the sensor). The temperature value is shown in bits 2 to 15.

0x03C9 = 0b**1111001001**

Temperature value: 0b11110010 = 242

Therefore: The current temperature value is 24.2 °C.

Example: Writing IO-Link value (operating mode "IO-Link")

59804

Task: Switch on the buzzer of DV2500 at IO-Link Port X2. The DV2500 operates in On/Off mode.

Solution: The IODD of the DV2500 shows the structure of the IO-Link process value (→ e.g. LED activity). The buzzer will be switched using bit 40 of the process value (OFF = 0, ON = 1).

To switch the buzzer:

1. Read the current process value (→ **Example: Read IO-Link process data (operating mode "IO-Link")** (→ p. 55)).
2. Set bit 40 of the read value to 1.
3. Write the process value to the IO-Link device.

Example:

Read process value:

0x0000 0000 004D = 0b0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0100 1101

New process value:

0b0000 000**1** 0000 0000 0000 0000 0000 0000 0000 0000 0100 1101 = 0x0100 0000 004D

- Request:

```
{
"code":"request",
"cid":10,
"adr":"iolinkmaster/port[2]/iolinkdevice/pdout/setdata",
"data":{"newvalue":"01000000004D"}
}
```

- Response:

```
{
"cid":10,
"code":200
}
```

Example: Writing digital output (operating mode "DO")

59803

Task: Set the output value of the IO-Link devices at IO-Link Port X1 to "ON". The operating mode of the IO-Link port is "Digital Output (DO)".

Solution: Write the value 1 to data point pdout. The value has to be written as hexadecimal value with a length of 1 byte (OFF = "00", ON = "01").

- Request:

```
{
"code":"request",
"cid":10,
"adr":"iolinkmaster/port[1]/iolinkdevice/pdout/setdata",
"data":{"newvalue":"01"}
}
```

- Response:

```
{
"cid":10,
"code":200
}
```

Example: Reading digital input (operating mode "DI")

59802

Task: Read the current input value of the IO-Link device at IO-Link port X5. The operating mode of the IO-Link port is "Digital Intput (DI)".

Solution: Read the value of data point pdin. The value will be returned as hexadecimal value with a length of 1 byte (OFF = "00", ON = "01"").

- Request:


```
{
"code":"request",
"cid":10,
"adr":"iolinkmaster/port[5]/iolinkdevice/pdin/getdata"
}
```


- Response:


```
{
"cid":10,
"data":{"value":"00"},
"code":200
}
```

9.2.15 IO-Link ports: Indicating port events

59796

Substructure: iolinkmaster/port[n] (n = 1...8).

Available data points:

Name	Description	Access
../portevent	Indication of the following events at IO-Link port n: ▪ plugging IO-Link device ▪ pulling IO-Link device ▪ changing operating mode of IO-Link port	r

r ... read only



Subscribing events: → **Subscribing to notifications** (→ p. 64)

9.2.16 IO-Link devices: Accessing parameters

59800

The ifm IoT Core supports the configuration of the connected IO-Link devices. A parameter is accessed via IO-Link index and subindex (→ IO Device Description (IODD) of the device).

Substructure: `iolinkmaster/port[n]/iolinkdevice` (n = 1...8)

Applicable services:

Service	Description
../iolreadacyclic	Read a parameter of an IO-Link device (acyclic)
../iolwriteacyclic	Write a parameter of an IO-Link device (acyclic)

Example: Read the parameter value of an IO-Link device

33847

Task: Read the serial number of the ifm temperature sensor TN2531 at IO-Link port X02

Solution: Read the serial number with the `iolreadacyclic` service from the IO-Link device (index: 21, subindex: 0)

- **Request:**

```
{
"code":"request",
"cid":4711,
"adr":"/iolinkmaster/port[2]/iolinkdevice/iolreadacyclic",
"data":{"index":21,"subindex":0}
}
```

- **Return:**

```
{
"cid":4711,
"data":{"value":"4730323134323830373130"},
"code":200
}
```

The returned value is given in hexadecimal format. The conversion of the HEX value in a STRING value is: G0214280710

Example: Change the parameter value of an IO-Link device

33844

Task: Set the output configuration OUT1 of the ifm temperature sensor TN2531 at IO-Link port X02 to the value "Hnc / hysteresis function, normally closed".

Solution: Change the parameter [ou1] of the sensor to the value 4 using the iolwriteacyclicdata service. The parameter can be accessed via IO-Link index 580, subindex 0 (→ IO-Link description of the sensor).

- Request:

```
{
"code":"request",
"cid":4711,
"adr":"/iolinkmaster/port[2]/iolinkdevice/iolwriteacyclic",
"data":{"index":580,"subindex":0,"value":"34"}
}
```

The value has to be given in hexadecimal format. The conversion of the STRING value in a HEX value is: 34.

- Response:

```
{
"cid":4711,
"code":200
}
```

9.2.17 IO-Link devices: Reading an writing device information

59797

Substructure: `iolinkmaster/port[n]/iolinkdevice` (n = 1...8)

Available data points:

Name	Description	Access
../status	Status of the connected IO-Link device	r
../vendorid	IO-Link ID of the vendor	r
../deviceid	IO-Link ID of the IO-Link device	r
../productname	Product name of the IO-Link device	r
../serial	Serial number of the IO-Link device	r
../applicationspecifictag	Device-specific identification (application tag)	rw

r ... read only
rw ... read and write

9.2.18 IO-Link devices: Indicating IO-Link events

59798

Substructure: `iolinkmaster/port[n]/iolinkdevice` (n = 1...8).
Available data points:

Name	Description	Access
../iolinkevent	Indication of IO-Link events	r

r ... read only



Subscribing events: → **Subscribing to notifications** (→ p. 64)

9.2.19 Gateway: Resetting, rebooting and localising the device

59790

Substructure: `firmware`
Applicable services:

Name	Description
../factoryreset	Reset IO-Link master to factory settings
../reboot	Reboot IO-Link master
../signal	Trigger the flashing of the status LED

9.2.20 Gateway: Reading device information

52254

Substructure: `deviceinfo`
Available data points:

Name	Description	Access
../productcode	Article number	r
../vendor	Manufacturer	r
../devicefamily	Device family	r
../hwrevision	Hardware revision	r
../serialnumber	Serial number	r
../revision	Firmware version	r
../bootloaderrevision	Bootloader version	r
../extensionrevisions	Firmware and bootloader version	r
../fieldbustype	Fieldbus	r

r ... read only

Additional information about the AL1342 can be read with the service `getidentity` (→ **Servicet: getidentity** (→ p. 139)).

9.2.21 Gateway: Reading status and diagnostic information

61157

Substructure: `processdatamaster`
Available data points:

Name	Description	Access
../temperature	Temperature of the IO-Link master (value in °C)	r
../voltage	Present voltage value of the supply voltage US (value in mV)	r
../current	Present current value of the sensor supply US (value in mA)	r
../supervisionstatus	Status of the device supply US	r

r ... read only

9.2.22 Gateway: Updating the firmware

59789

Substructure: `firmware`

Available data points:

Name	Description	Access
../version	Software version	r
../type	Software type	r
../container	Structure for updating the firmware	w
../container/maxsize	Maximum size of the container structure (in bytes)	r
../container/chunksize	Size of a data segment (in bytes)	r
../container/size	Size of the container content (in bytes)	r

r = only read
w = write only

Applicable services:

Name	Description
../install	Install firmware transferred to the IO-Link master
../container/stream_set	Transfer an individual data segment
../container/start_stream_set	Start sequential transmission of several data segments

Example: Update firmware

52252

Task:

Update the firmware of the device; size of the firmware file: 356676 bytes

Solution:

The firmware is transferred to the device in fragments (chunks). The size of the fragments depends on the size of the flash memory of the IO-Link master. To transfer the firmware, the firmware file must be converted into a character string using BASE64.

1 Preparations

- ► Determine the size of the fragments (g = number of bytes):
  {"code":"request", "cid": -1, "adr":"/firmware/container/chunksize/getdata"}
- ► Convert the firmware file into a BASE64 string.

2 Start the transfer of the firmware

- ► Start the transfer of the firmware via the service start_stream_set (parameter "size": size of the firmware file):
  {"code":"request", "cid": -1, "adr":"/firmware/container/start_stream_set", "data":{"size":356676}}

3 Load the firmware into the flash memory of the IO-Link master

► Send the BASE64 string of the firmware file to the IO-Link master fragment by fragment (value = string value with length g).
{"code": "request", "cid": -1, "adr": "/firmware/container/stream_set", "cid": -1, "data": {"value": "aWZtfgIAAABBTDF4NXhfY25faXRfdDIuMi43Nw..."}

► Repeat step 3 until all fragments of the firmware file have been sent to the IO-Link master.

> IO-Link master stores the segments received in the container area.

4 Install firmware

► Start the installation of the transmitted firmware.
{"code": "request", "cid": -1, "adr": "/firmware/install", "data": {}}

9.2.23 Gateway: Setting the application tag

59791

Substructure: `devicetag`

Available data points:

Name	Description	Access
../applicationtag	Name of the IO-Link master (application tag)	rw

rw ... read and write

For the storage of the applicationtag 32 bytes are available on the IO-Link master. If the memory area is exceeded during writing with setdata, the IoT core aborts the write process and returns the diagnostics code 400.

When writing the application tag, note the different memory requirements of the individual UTF-8 characters:

- characters 0-127: 1 byte per character
- characters >127: more than 1 byte per character

Example: Change name of the IO-Link master

a33823

Task: Set the name of the IO-Link master to AL1342 for the representation in the LR SMARTOBSERVER.

Solution: Change the parameter [Application Tag] with the setdata service to the value [AL1342].

The data point of the parameter [Application Tag] in the device description object is /devicetag/applicationtag.

- Request:

```
{
"code":"request",
"cid":4711,
"adr":"/devicetag/applicationtag/setdata",
"data":{"newvalue":"AL1342"}
}
```

- Response:

```
{"cid":4711,"code":200}
```

9.2.24 Subscribing to notifications

61159

If a data point has the sub-element `datachanged`, the user can subscribe to notifications on value and condition changes. Notifications can be triggered by the expiration of a timer or an event. The IoT Core supports the output of notifications in CSV or JSON format.

Available data points:

Name	Description	Access
timer[x]/counter	Timer for triggering a notification	rw
timer[x]/interval	Cycle time of the update of the subscribed values	rw
iolinkmaster/port[n]/portevent	Display of the following events on IO-Link port n: ▪ IO-Link device connected ▪ IO-Link device disconnected ▪ Operating mode of the IO-Link port changed	rw
iolinkmaster/port[n]/iolinkdevice/iolinkevent	Display of IO-Link events	rw

r ... read only
rw ... read and write
x = [1,2]
n = 1...8

Applicable services:

Name	Description
../datachanged/subscribe	Subsrscibe to notification
../datachanged/unsubsribe	Unsubscribe notification
../datachanged/getsubscriptioninfo	Show information about notifications

Additionally, the user can use **Service: getsubscriberlist** (→ p. 140) show all active subscriptions.

Example: Subscribing to notifications

61160

Task: The current values of the following parameters are to be sent regularly to a network server with IP address 192.168.0.4:

- cyclic input data of the IO-Link Devices an IO-Link port X02
- Operating temperature of the IO-Link master.

Solution: Subscribe to the required data using the subscribe service.



The following options are additionally available:

- via WebSockets (ws://): **Example: Subscribing notifications via WebSocket** (→ p. 68)
- via MQTT (mqtt://): **Example: Configuring the MQTT command channel** (→ p. 71)

- Request:

```
{
"code":"request",
"cid":4711,
"adr":"/timer[1]/counter/datachanged/subscribe",
"data":
{
```

```
"callback":"http://192.168.0.4:80/temp",
"datatosend":[
"/iolinkmaster/port[2]/iolinkdevice/pdin",
"/processdatamaster/temperature"]
}
}
```

In addition, the time interval of the timer[1] must be set to a value between 500 ms and 2147483647 ms.

- Request:

```
{
"code":"request",
"cid":4712,
"adr":"/timer[1]/interval/setdata",
"data":{"newvalue":500}
}
```

- Response:

```
{
"cid":4712,
"code":200
}
```

- Notification (JSON)

```
{
"code":"event",
"cid":4711,
"adr":"",
"data":{
"eventno":"6317",
"srcurl":"/timer[1]/counter/datachanged",
"payload":{
"/timer[1]/counter":{"code":200,"data":1},
"/processdatamaster/temperature":{"code":200,"data":39},
"/iolinkmaster/port[2]/iolinkdevice/pdin":{"code":200,"data":"03B0"}}}
}
```

Example: Changing a subscription

61161

Task: The existing subscription (**Example: Subscribing to notifications** (→ p. 64)) is to be changed. Instead of the temperature of the IO-Link master, the operating voltage applied is to be transmitted.

Solution: Overwrite the existing subscription. For this purpose, the parameter values for "cid" and "callback" in the request must be the same as those of the existing subscription.

- Request:

```
{
"code":"request",
"cid":4711,
"adr":"/timer[1]/counter/datachanged/subscribe",
"data":{
"callback":"http://192.168.0.4:80/temp",
"datatosend":[
"/iolinkmaster/port[2]/iolinkdevice/pdin",
"/processdatamaster/voltage"]}
}
```

Example: Subscribing to notifications in CSV format

61162

Task: Every 2 seconds, the current values of the following parameters are to be sent to a network server with the IP address 192.168.0.4

- cyclic IO-Link input data of the IO-Link device at port X02
- Operating temperature of the IO-Link master.

The data should be transmitted in CSV format (comma separator).

Solution:

► Use the subscribe service to subscribe to the required data and set the output format to "csv0".



Data in CSV format can only be sent via TCP protocol.

- Request:

```
{
"cid": 1,
"adr":"/timer[1]/counter/datachanged/subscribe",
"code":"request",
"callback":"tcp://192.168.50.59:1883/topic",
"codec":"csv0",
"data":{
"datatosend":[
"/iolinkmaster/port[2]/iolinkdevice/pdin",
"/processdatamaster/temperature"]}
}
```

► Set the interval of the timer to 2 seconds:

- Request:

```
{
"code":"request",
"cid":4712,
"adr":"/timer[1]/interval/setdata",
"data":{"newvalue":2000}
}
```

The cyclically sent notification has the following structure:

```
/timer[1]/counter/datachanged,6317,200,1,200,39,200,03B0
```

Example: Unsubscribing from notifications

61163

Task: The existing subscription (**Example: Subscribing to notifications** (→ p. 64)) is to be deleted.

Solution: Use the unsubscribe service to delete the subscription. For this purpose, the value of the parameter "callback" in the request must be equal to the value of the existing subscription.

```
{
"code":"request",
"cid":4711,
"adr":"/timer[1]/counter/datachanged/unsubscribe",
"data":{
"callback":"http://192.168.0.4:80/temp"
}
```

Example: Checking subscriptions

61164

Task: Information about the existing subscription (**Example: Subscribing to notifications** (→ p. 64)Show **Example: Subscribing to notifications** (→ p. 64)).

Solution: Use the service getsubscriptioninfo and the parameter values cid, "adr" and "callback" of the existing subscription to retrieve the information.

- Request:

```
{
"code":"request",
"cid":4711,
"adr":"/timer[1]/counter/datachanged/getsubscriptioninfo",
"data":{
"callback":"http://192.168.0.4:80/temp"}
}
```

- Response:

```
{
"cid." 4711,
"data":{
"callback":"http://192.168.0.4:80/temp",
"datatosend":[
"/iolinkmaster/port[2]/iolinkdevice/productname",
"/iolinkmaster/port[2]/iolinkdevice/pdin",
"/processdatamaster/temperature"]},
"code":200
}
```

9.2.25 Using Web Socket

61165

The IoT Core supports communication via WebSocket protocol. With Web Sockets, the user can establish a full-duplex communication channel via a TCP connection.

WebSockets can be used for the following services:

- subscribe / unsubscribe

Maximum number of WebSocket connections: 8
Fail-safe WebSocket connections (wss://) are not supported.

To transmit notifications via a WebSockets connection:

► Establish the WebSocket connection (e.g. "`ws://192.168.0.55:80/websocket`")

- Option 1: without parameter "callback"

► make subscribe/unsubscribe request without parameter "callback".

> IoT-Core sends notifications about existing WebSocket connections.

- Option 2. with parameter "callback"

► make subscribe/unsubscribe requests with parameter "callback" ("`ws:///myTopic`").

> IoT-Core sends notifications about existing WebSocket connections to the topic `myTopic`.

Example: Subscribing notifications via WebSocket

61166

Task: The current values of the following parameters are to be sent regularly to the data sink `myTopic` via an existing WebSocket connection:

- Product name of the IO-Link Devices an IO-Link port X02
- cyclic input data of the IO-Link Devices an IO-Link port X02
- Operating temperature of the IO-Link master.

Solution: Subscribe to the required data using the subscribe service.

- Request:

```
{
"code":"request",
"cid":4711,
"adr":"/timer[1]/counter/datachanged/subscribe",
"data":{
"callback":"ws:///myTopic",
"datatosend":[
"/iolinkmaster/port[2]/iolinkdevice/productname",
"/iolinkmaster/port[2]/iolinkdevice/pdin",
"/processdatamaster/temperature"]}
}
```

If the notifications are to be transmitted via the existing WebSocket connection, but without a special data sink, the callback parameter is not required.

- Request:

```
{
"code":"request",
"cid":4711,
"adr":"/timer[1]/counter/datachanged/subscribe",
"data":{
"datatosend":[
```

```
"/iolinkmaster/port[2]/iolinkdevice/productname",
"/iolinkmaster/port[2]/iolinkdevice/pdin",
"/processdatamaster/temperature"]}
}
```

9.2.26 MQTT support

61168

The IoT Core supports the MQTT protocol. The protocol allows an MQTT client to communicate with the IoT Core via an MQTT broker to request and receive data. The IoT Core can publish data via the MQTT connection.

Configuring the MQTT command channel

61169

To enable MQTT communication, the user needs to activate and configure an MQTT command channel.

Substructure: `connections/mqttConnection`

Name	Description	Access
../type	Type of the connection (MQTT)	r
../status	Global MQTT status	r
../status/preset	Presetting of the MQTT status; Basic settings: running	r
../MQTTSetup	Substructure for general MQTT settings	w
../MQTTSetup/QoS	Quality of Service of the MQTT communication ▪ 0: QoS Level 0 - PUBLISH (without confirmation) ▪ 1: QoS Level 1 - PUBLISH > PUBREC (one-time confirmation) ▪ 2: QoS Level 2 - PUBLISH > PUBREC > PUBREL > PUBCOMP (double confirmation)	rw
../MQTTSetup/version	MQTT version	r
../mqttCmdChannel	Substructure of the MQTT command channel	w
../mqttCmdChannel/type	Type of the MQTT command channel	r
../mqttCmdChannel/status	Status of the MQTT command channel	r
../mqttCmdChannel/status/preset	Presetting of the MQTT status; Basic setting: stopped	r
../mqttCmdChannel/mqttCmdChannelSetup	Structure for settings of the command channel	w
../mqttCmdChannel/mqttCmdChannelSetup/brokerIP	IP address of the MQTT broker	rw
../mqttCmdChannel/mqttCmdChannelSetup/brokerPort	Port number of the MQTT broker	rw
../mqttCmdChannel/mqttCmdChannelSetup/cmdTopic	Designation of the MQTT topic	rw
../mqttCmdChannel/mqttCmdChannelSetup/defaultReplyTopic	Standard response topic	rw

Applicable services:

Name	Description
../status/start	Enable MQTT
../status/stop	Deactivate MQTT
../status/reset	Reset MQTT
../mqttCmdChannel/status/start	Activate MQTT command channel
../mqttCmdChannel/status/stop	Deactivate MQTT command channel
../mqttCmdChannel/status/reset	Reset MQTT command channel

Notes on the states of an MQTT connection: **Note: Connection states** (→ p. 71)

To create an MQTT connection, perform the following steps in sequence:

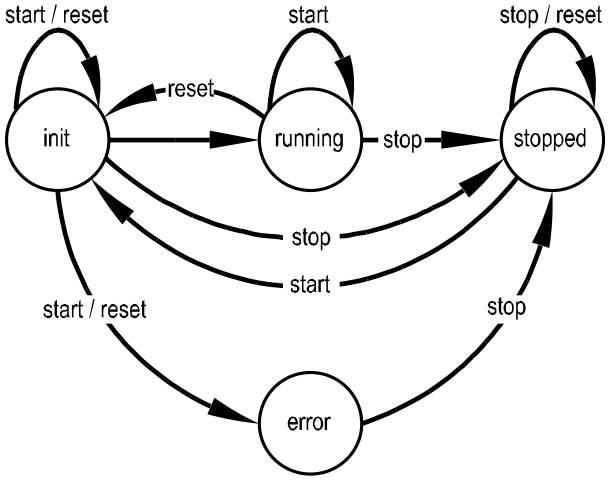
Ensure that the MQTT broker can be reached and that the selected port of the MQTT broker is enabled for data transmission.
Max. number of simultaneous MQTT connections: 10
Wildcards "+" and "#" in topics are not supported.

► Activate MQTT command channel.
► Set the IP address of the MQTT.
► Set the port number of the MQTT broker.
► Set topic.
► Set standard response topic.
> The command channel is created with the selected properties.
> The user can publish on the topic with the IoT Core.
> MQTT clients can subscribe to the topic.

Note: Connection states

61170

The following status diagram shows the influence of the services "start", "stop" and "reset" on the status of an MQTT connection:



After the initialisation in the "init" state has been completed, the connection automatically changes to the "running" state.

The connection automatically switches to the "error" state if at least one of the following events occurs:

- no MQTT broker available

Example: Configuring the MQTT command channel

61171

Task: Configuring and activating the MQTT command channel (IP address MQTT broker: 192.168.82.100, port: 1883, topic: abc).

Solution:

► Check whether MQTT broker can be reached and the port has been released.
► Activate command channel

- Request:

```
{
"code":"request",
"cid":4711,
"adr":"/connections/mqttConnection/MQTTSetup/mqttCmdChannel/status/start"
}
```

► Set the IP address of the MQTT broker/server.

- Request:

```
{
"code":"request",
"cid":4712,
"adr":"/connections/mqttConnection/mqttCmdChannel/mqttCmdChannelSetup/brokerIP/set
data"
"data":{"192.168.82.100"}
}
```

► Set the port number of the MQTT broker/server.

- Request:

```
{
"code":"request",
"cid":4713,
"adr":"/connections/mqttConnection/mqttCmdChannel/mqttCmdChannelSetup/brokerPort/s
etdata"
"data":{"1883"}
}
```

► Set topic.

- Request:

```
{
"code":"request",
"cid":4714,
"adr":"/connections/mqttConnection/mqttCmdChannel/mqttCmdChannelSetup/cmdTopic/set
data"
"data":{"abc"}
}
```

► Set standard response topic.

- Request:

```
{
"code":"request",
"cid":4715,
"adr":"/connections/mqttConnection/mqttCmdChannel/mqttCmdChannelSetup/defaultReply
Topic/setdata"
"data":{"xyz"}
}
```

► Set the QoS.

- Request:

```
{
"code":"request",
"cid":4716,
"adr":"/connections/mqttConnection/MQTTSetup/QoS/setdata",
"data":{"QoS2"}
}
```

Example: Publish the temperature to an MQTT broker

54687

Task: Publish the temperature of the IO-Link master to an MQTT broker (IP address MQTT broker: 192.168.82.100, port: 1883, topic: abc

Solution:

- Request:

```
{
"code":"request",
"cid":-1,
"adr":"/timer[1]/counter/datachanged/subscribe",
"data":{
"callback":"mqtt://192.168.82.100:1883/abc",
"datatosend":["processdatamaster/temperature"}
}
```

- Response:

```
{
"cid":-1,
"code":200
}
```

9.2.27 Using the IoT-Core Visualizer

Content

Managing notifications 75
Searching for elements in the device tree 77
Configuring IO-Link the master 78
Reading and writing process data 79
Updating the firmware 80

61173

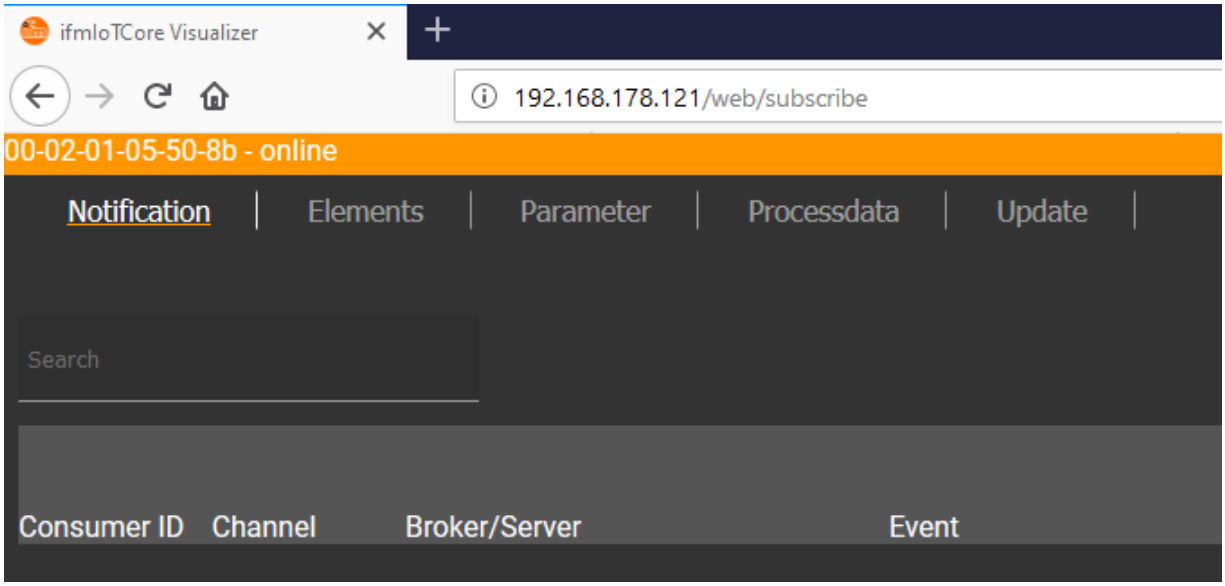
The ifm-IoT Core Visualizer of the IO-Link master provides a graphical user interface for accessing functions of the ifm-IoT Core.

To start the IoT Core Visualizer:

► Start web browser.

► Call the following address: http://ipaddress/web/subscribe

> Browser shows IoT Core Visualizer:



The navigation menu gives the user access to the following functions:

- [Notification]: Creating and managing notifications (subscribe / unsubscribe)
- [Elements]: Searching for elements in device description
- [Parameter]: Configuring IO-Link master
- [Processdata]: Reading and writing process data
- [Update]: Updating the firmware of the IO-Link master

Managing notifications

61174

The menu page allows you to perform the following functions

- Creating notifications
- Showing active notifications
- Deleting notifications (single, all)

Requirements:

- Iot-Core Visualizer has been started.

► Click on [Notification].

> The menu page for managing notifications appears.

> The menu page shows all registered notifications in a table

Creating a new notification

61175

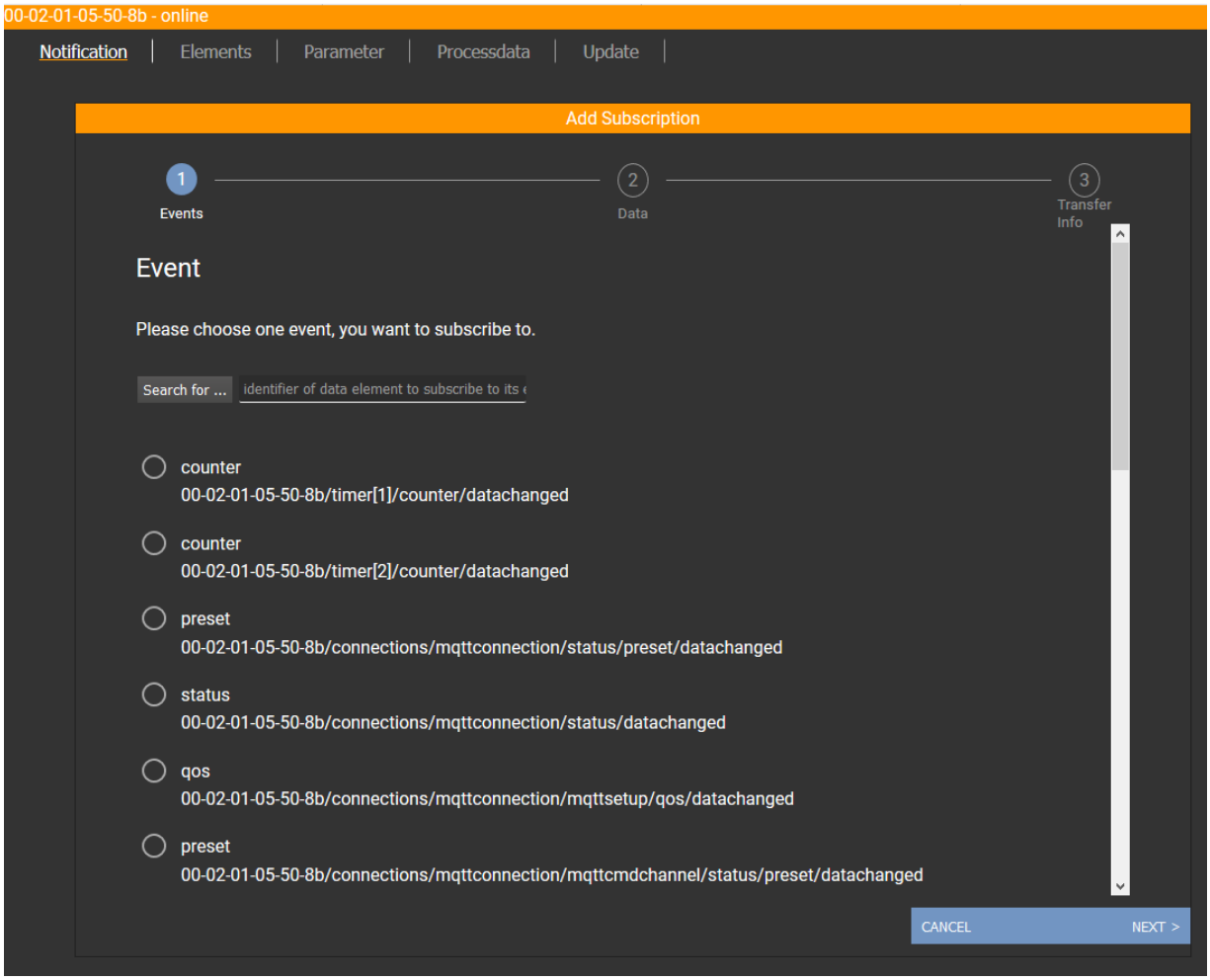
A wizard is used to register new notifications.

Requirements:

- The [Notification] menu page is open.

► Click on [+] on the right side of the table.

> The wizard for the creation of notifications appears.



► Use the wizard to enter the required notification parameters step by step.
> Created notification subscription is displayed in the table.

For cyclical notifications via timer[1] or timer[2], the user also needs to set the interval time of the timer in question.

Deleting a notification

61176

Requirements:

- The [Notification] menu page is open.
- At least one notification is active.

► Click on [x] in the column [Unsubscribe].
> The selected notification will be deleted (unsubscribe).

Searching for elements in the device tree

61177

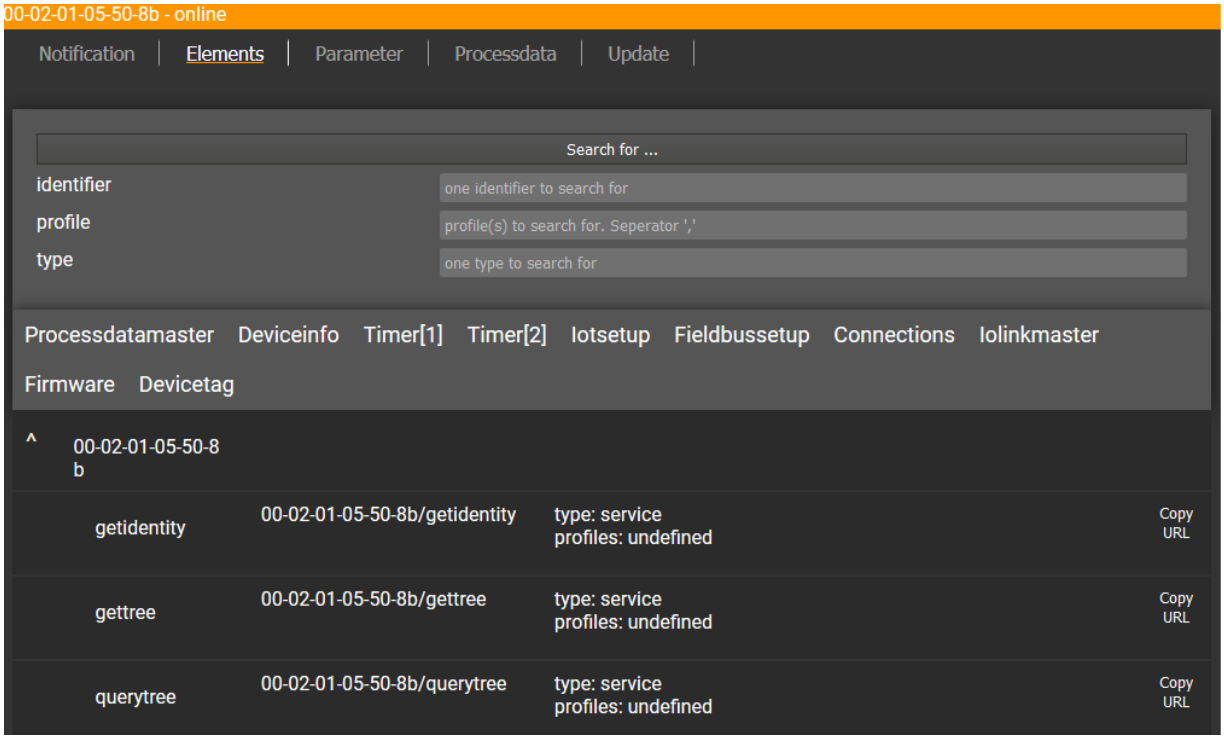
The [Elements] menu page allows you to search the device description for elements with specific properties (status, profile, name) and to output the results.

Requirements:

- Iot-Core Visualizer has been started.

► Click on [Elements].

> The input mask appears.



► Enter the search criteria of the required item in the [identifier], [profile] and [type] boxes.

► Click on [Search for ...].

> IoT-Core Visualizer searches device description for elements with selected search criteria.

> The result list shows all elements found.

Configuring IO-Link the master

61178

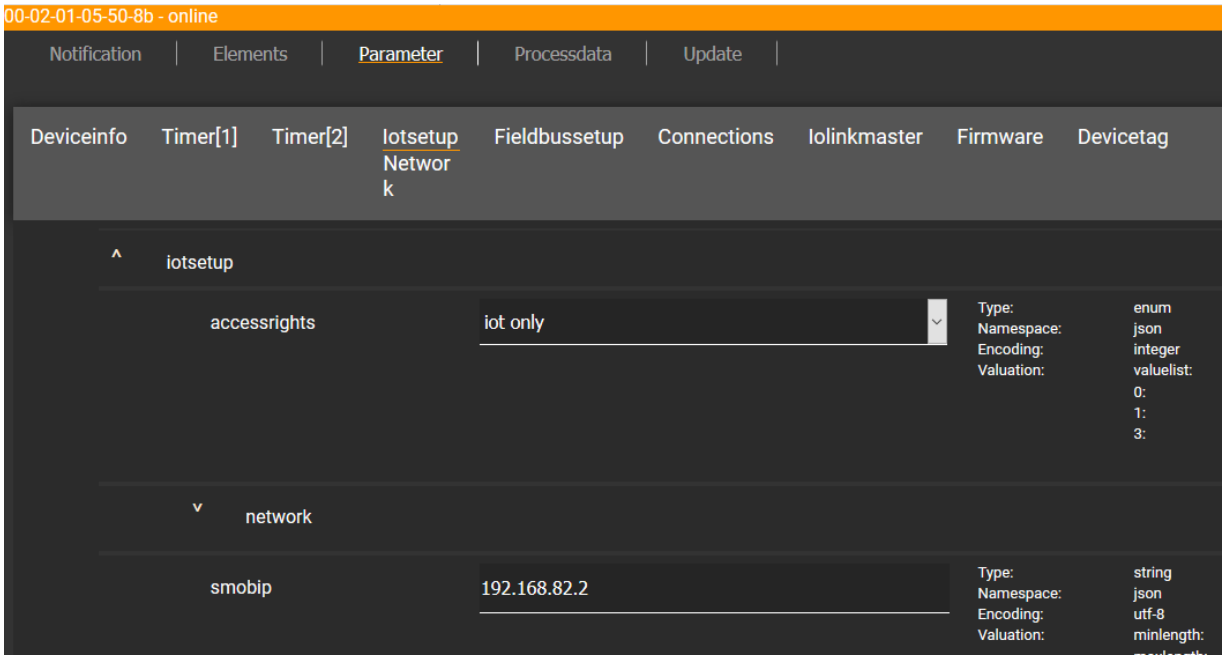
The [Parameter] menu page allows you to configure the IO-Link master.
Available options:

- Reading and writing individual parameters
- Backup and restore the current configuration of the machine.

Requirements:

- Iot-Core Visualizer has been started.

► Click on [Parameter].
> The menu page shows the available parameters of the IO-Link master.
> Current parameter values are displayed.
> Editable parameters can be changed.



To change a parameter:

► Navigate to the desired parameter in the device description.
► Changing the parameter value
► Click on the pencil icon to save the change on the IO-Link master.
> The changed parameter value is active.
► Optional: Repeat the procedure to change further parameter values.

Reading and writing process data

61179

The menu page allows the process data of the IO-Link master and the connected IO-Link devices to be read and written.

Requirements:

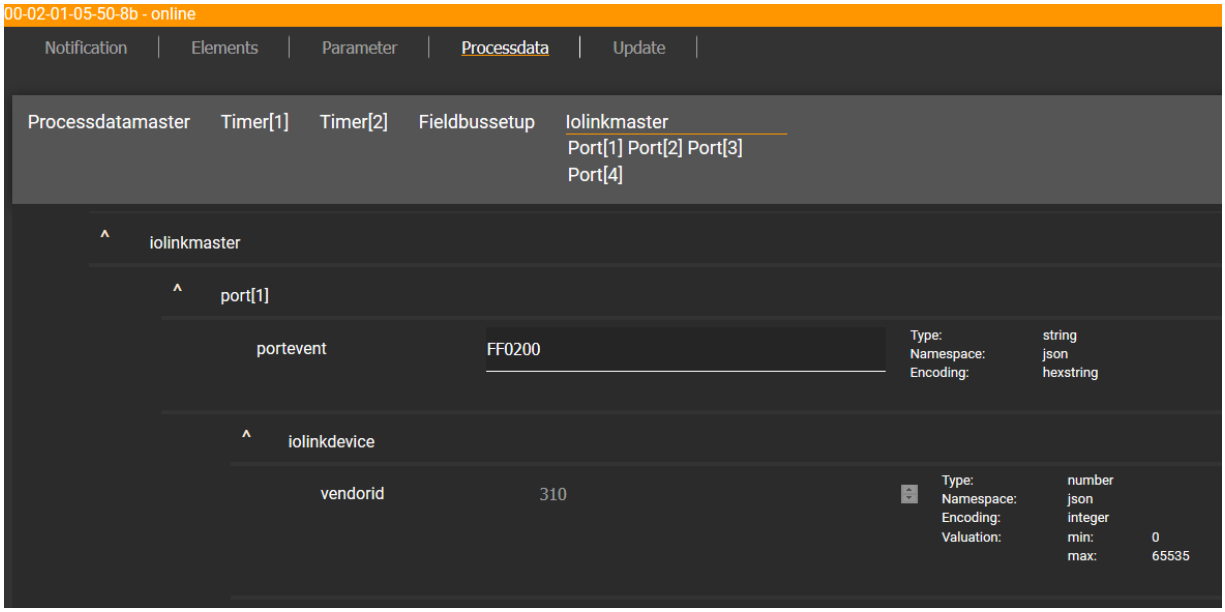
- Iot-Core Visualizer has been started.

► Click on [Processdata].

> Menu page shows the substructures of the device description that contain process data and events.

> The current process values are displayed.

> Editable process data can be changed.



To change the value of a process date:

► Navigate to the required process date in the device description.

► Change the process value.

► Click on the pencil icon to save the change on the IO-Link master.

> The changed process value is active.

► Optional: Repeat the procedure to change further process values.

Updating the firmware

61180

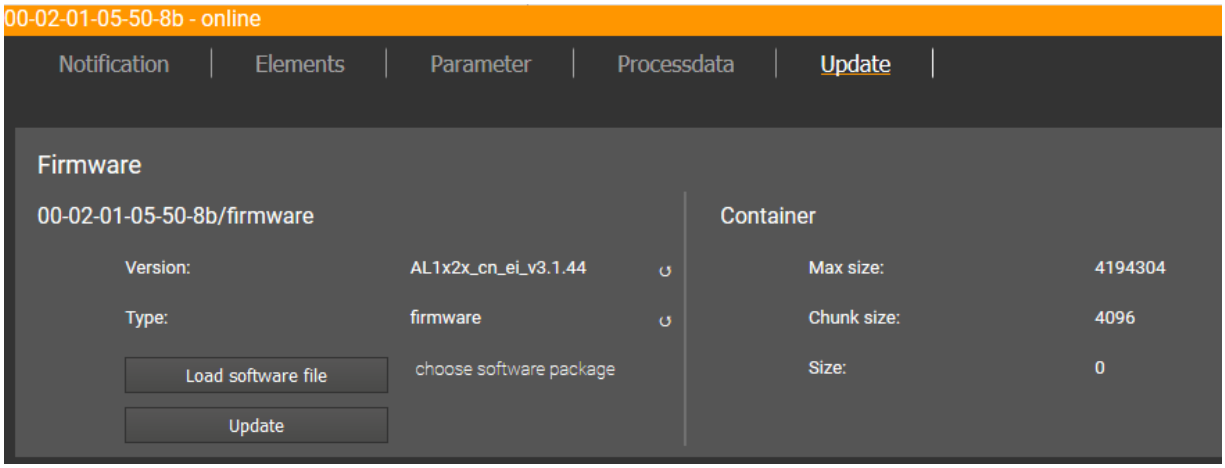
The [Update] menu page allows you to update the firmware of the IO-Link master:

Requirements:

- Iot-Core Visualizer has been started.

► Click on [Update].

> Menu page displays information about the current firmware version.



► Click on [Load software file] and select a new firmware file (*.bin).

► Click on [Update] to start the update process.

> The firmware of the IO-Link master will be updated.

> The area shows the progress bar.

> If the update process has been successful, the IO-Link master will restart automatically.

9.3 Modbus TCP

Content

Integrate the AL1342 into the Modbus project 81
Set IO-Link master....82
Set IO-Link ports....83
Read input data of several IO-Link ports....84
Read input data of individual IO-Link ports 85
Write output data of several IO-Link ports....86
Write output data of individual IO-Link ports 87
Read diagnostic information and events 88
Read device information....89
Control IO-Link master 89
Configure IO-Link devices 89
Modbus TCP: Programmers' notes 90

34455

On the field bus side, the device can be configured with any Modbus TCP compatible projection software.

9.3.1 Integrate the AL1342 into the Modbus project

34456

The AL1342 provides the functionality of a Modbus-TCP slave. The user can integrate the IO-Link master via the profile of a generic Modbus-TCP slave to a fieldbus project.

The IO-Link master, the IO-Link Ports and the process data are configured via the Modbus register of the AL1342.

Example: Integrate IO-Link master in a CODESYS project

34474

Familiarise yourself with the following CODESYS functions!

- Modbus master:
  → Online help > Fieldbus support >Modbus configurator >Modbus master
- Modbus slave device:
  → Online help > Fieldbus support > Modbus configurator > Modbus slave device

Task: Integrate IO-Link master in a CODESYS project

Hardware:

- AC14 DL as Modbus-TCP master
- AL1342 as Modbus-TCP slave

Solution:

Preparation:

► Create CODESYS project with AC14 DL.

1 Create Modbus-TCP master

► In the device tree: Right-click on [X8] node

> Context menu appears.

► In the context menu: Select [Add Device...].

> Dialogue window appears.

► Select the following settings:
1. [Vendor]: Select [ifm electronic].
2. [Device]: Select [Modbus_TCP_Master].
3. [Name]: Enter a unique name.

► Click on [Add Device].

> Device tree shows Modbus-TCP master as sub-node of the interface X8.

2 Create Modbus-TCP slave (AL1342)

► In the device tree: Right-click on the node of the added Modbus-TCP master

► In the context menu: Select [Add Device...].

> Dialogue window appears.

► Select the following settings:
1. [Vendor]: Select [ifm electronic].
2. [Device]: Select [Modbus_TCP_Slave].
3. [Name]: Enter a unique name

► Click on [Add Device].

> Device tree shows AL1342 as sub-node of the Modbus-TCP master.

3 Configure Modbus-TCP slave

► In the following tabs, set the parameters as required:
1. [General]: Set IP address and Unit ID
2. [Modbus Slave Channel]: Add Modbus register
2. [ModbusTCPSlave I/O Mapping]: Map Modbus register values to variables

9.3.2 Set IO-Link master

54624

Register area for the access to the configuration of the IO-Link master: →**Configuration Area** (→ p. 103)

The area contains the following data:

- Access rights to the IO-Link master
- Data length of the IO-Link input and output data of all IO-Link ports
- Alignment of the bytes in a data word

► Observe the general rules for access to the Modbus registers (→**Rules for accessing the Modbus registers** (→ p. 90))!

► When writing several registers at conce, ensure that the transferred parameter data has the correct length!

Register	Content	Access
8998	Access Rights; Process Data Length	r/w
8999	Byte Swap	r/w

r/w ... read and write

9.3.3 Set IO-Link ports

34461

Register area for the access to the configuration of the IO-Link ports: →**Configuration Area** (→ p. 103)

The area contains the following data:

- Operating mode of the IO-Link ports
- Device validation and Data storage settings
- Failsafe values of outputs

► Observe the general rules for access to the Modbus registers (→**Rules for accessing the Modbus registers** (→ p. 90))!

► When writing several registers at once, ensure that the transferred parameter data has the correct length!

Register	Contents	Access
9000	Port X01: Port Configuration	r/w
9006	Port X02: Port Configuration	r/w
9012	Port X03: Port Configuration	r/w
9018	Port X04: Port Configuration	r/w
9024	Port X05: Port Configuration	r/w
9030	Port X06: Port Configuration	r/w
9036	Port X07: Port Configuration	r/w
9042	Port X08: Port Configuration	r/w

r/w ... read and write

In addition, the user can set the IO-Link ports of the AL1342 via the following acyclic commands:

- "Set Mode": →**Command 0x10 – Set mode** (→ p. 119)
- "Set Validation ID / Data Storage": →**Command 0x20 – Set validation ID / data storage** (→ p. 121)
- "Set Fail-safe Data Pattern": →**Command 0x30 – Set fail-safe data pattern** (→ p. 123)

The commands use the process mechanisms of the acyclic command channel (→**Use acyclic services** (→ p. 91)).

9.3.4 Read input data of several IO-Link ports

34464

Register area for compact access to the input data of the IO-Link ports X01…X04 and X05...X08: →**Input Data** (→ p. 108)

The area contains the following data:

- Combined digital inputs - pin 2 / pin 4 (DI)
- Status information of the IO-Link ports
- Status information of the IO-Link devices
- Combined IO-Link input data of the IO-Link ports

Observe the general rules for access to the Modbus registers (→**Rules for accessing the Modbus registers** (→ p. 90))!

The parameter "Invalid Data" indicates whether the read IO-Link input data is valid.

► Together with the input data, also read and evaluate the corresponding status information of the IO-Link devices!

Register	Contents	Access
197	Port X01…X04: Digital Input - pin 2 / pin 4 (DI)	r
198	Port X01…X04: Status Information IO-Link Ports	r
199	Port X01…X04: Status Information IO-Link Devices	r
200	Port X01…X04: Compact Input Data - IO-Link (4n bytes)	r
297	Port X05…X08: Digital Input - pin 2 / pin 4 (DI)	r
298	Port X05…X08: Status Information IO-Link Ports	r
299	Port X05...X08: Status Information IO-Link Devices	r
300	Port X05…X08: Compact Input Data - IO-Link (4n bytes)	r

r ... read only
n = [2,4,8,16,32]; is determined by parameters [Process Data Length] (→**Configuration Area** (→ p. 103)

9.3.5 Read input data of individual IO-Link ports

34463

Register area for separate access to input data of the individual IO-Link ports: →**Single Port Access** (→ p. 113)

The area contains the following data for each IO-Link port X01...X08:

- Digital input data at pin 2 / pin 4 (DI)
- Status information of IO-Link port
- Diagnostic and status information of the connected IO-Link device
- IO-Link input data

Observe the general rules for access to the Modbus registers (→**Rules for accessing the Modbus registers** (→ p. 90))!

The parameter "Invalid Data" indicates whether the read IO-Link input data is valid.

► Also read and evaluate the corresponding diagnostic information!

Register	Contents	Access
1000	Port X01: Digital Input - Pin 2 / Pin 4 (DI)	r
1001	Port X01: Diagnostic + Status Data	r
1002	Port X01: Input Data - IO-Link (n bytes)	r
2000	Port X02: Digital Input - Pin 2 / Pin 4 (DI)	r
2001	Port X02: Diagnostic + Status Data	r
2002	Port X02: Input Data - IO-Link (n bytes)	r
3000	Port X03: Digital Input - Pin 2 / Pin 4 (DI)	r
3001	Port X03: Diagnostic + Status Data	r
3002	Port X03: Input Data - IO-Link (n bytes)	r
4000	Port X04: Digital Input - Pin 2 / Pin 4 (DI)	r
4001	Port X04: Diagnostic + Status Data	r
4002	Port X04: Input Data - IO-Link (n bytes)	r
5000	Port X05: Digital Input - Pin 2 / Pin 4 (DI)	r
5001	Port X05: Diagnostic + Status Data	r
5002	Port X05: Input Data - IO-Link (n bytes)	r
6000	Port X06: Digital Input - Pin 2 / Pin 4 (DI)	r
6001	Port X06: Diagnostic + Status Data	r
6002	Port X06: Input Data - IO-Link (n bytes)	r
7000	Port X07: Digital Input - Pin 2 / Pin 4 (DI)	r
7001	Port X07: Diagnostic + Status Data	r
7002	Port X07: Input Data - IO-Link (n bytes)	r
8000	Port X08: Digital Input - Pin 2 / Pin 4 (DI)	r
8001	Port X08: Diagnostic + Status Data	r
8002	Port X08: Input Data - IO-Link (n bytes)	r

r .. read only
n = [2,4,8,16,32]; is determined by parameters [Process Data Length] (→**Configuration Area** (→ p. 103)

9.3.6 Write output data of several IO-Link ports

34473

Register area for compact access to the output data of the IO-Link ports X01…X04 and X05...X08: →**Output Data** (→ p. 111)

The area contains the following data:

- Digital output data at pin 4 (DO)
- IO-Link output data of the IO-Link ports

► Observe the general rules for access to the Modbus registers (→**Rules for accessing the Modbus registers** (→ p. 90))!

Several connected register areas can be written with one write command.

The IO-Link master writes only the outputs in "Compact Output Data" that are completely covered by the transferred output data.

Example: The configured process data length is 4 bytes. If all in all 5 words have been transferred to register 600, the IO-Link master writes the outputs X01 (words 1+2) and X02 (words 3+4). Output X03 is not written.

► When writing the IO-Link outputs, ensure that the output data has the correct length!

The output data is invalid in the following situations:

- no Ethernet cable connected
- PLC has terminated the connection
- connection to the PLC has a timeout

Register	Contents	Access
599	Port X01...X04: Digital Output - Pin 4 (DO)	r/w
600	Port X01...X04: Compact Output Data IO-Link (4n bytes)	r/w
699	Port X05...X08: Digital Output - pin 4 (DO)	r/w
700	Port X05...X08: Compact Output Data IO-Link (4n bytes)	r/w

r/w = read and write
n = [2,4,8,16,32]; is determined by parameters [Process Data Length] (→**Configuration Area** (→ p. 103)

9.3.7 Write output data of individual IO-Link ports

34462

Register area for separate access to output data of individual IO-Link ports: →**Single Port Access** (→ p. 113)

The area contains the following data for each IO-Link port X01...X08:

- Digital output data at pin 4 (DO)
- IO-Link output data

► Observe the general rules for access to the Modbus registers (→**Rules for accessing the Modbus registers** (→ p. 90))!

If the user tries to write more than 34 bytes of output data to a single port, the IO-Link master cancels the execution of the command and returns the error code "2 ILLIGEAL ADDRESS".

► When writing outputs, ensure that the length of the transferred output data corresponds with the configured process data length.

The output data is invalid in the following situations:

- no Ethernet cable connected
- PLC has terminated the connection
- Connection to the PLC has a timeout

Register	Contents	Access
1100	Port X01: Digital Output - Pin 4 (DO)	r/w
1101	Port X01: Output Data IO-Link (n bytes)	r/w
2100	Port X02: Digital Output - Pin 4 (DO)	r/w
2101	Port X02: Output Data IO-Link (n bytes)	r/w
3100	Port X03: Digital Output - Pin 4 (DO)	r/w
3101	Port X03: Output Data IO-Link (n bytes)	r/w
4100	Port X04: Digital Output - Pin 4 (DO)	r/w
4101	Port X04: Output Data IO-Link (n bytes)	r/w
5100	Port X05: Digital Output - Pin 4 (DO)	r/w
5101	Port X05: Output Data IO-Link (n bytes)	r/w
6100	Port X06: Digital Output - Pin 4 (DO)	r/w
6101	Port X06: Output Data IO-Link (n bytes)	r/w
7100	Port X07: Digital Output - Pin 4 (DO)	r/w
7101	Port X07: Output Data IO-Link (n bytes)	r/w
8100	Port X08: Digital Output - Pin 4 (DO)	r/w
8101	Port X08: Output Data IO-Link (n bytes)	r/w

r/w = read and write
n = [2,4,8,16,32]; is determined by parameters [Process Data Length] (→**Configuration Area** (→ p. 103)

9.3.8 Read diagnostic information and events

34467

Register area for the access to diagnostic information of the IO-Link ports X01…X08:→**Diagnostic data** (→ p. 105)

The area contains the following data:

- Status/error flags for port configuration
- Vendor ID / device ID of the connected IO-Link devices
- Events and corresponding event codes

► Observe the general rules for access to the Modbus registers (→**Rules for accessing the Modbus registers** (→ p. 90))!

A maximum of 3 events per IO-Link port are displayed.

One-time events will be deleted after a minimum of 10 s (Event Single Shot).

Occurring events indicate the time of the error occurrence (event appears). If the error cause disappears, this is indicated by a further event (event disappears). Both event type always occur in pairs.

Register	Contents	Access
30	Port X01: Diagnostic Data	r
40	Port X02: Diagnostic Data	r
50	Port X03: Diagnostic Data	r
60	Port X04: Diagnostic Data	r
70	Port X05: Diagnostic Data	r
80	Port X06: Diagnostic Data	r
90	Port X07: Diagnostic Data	r
100	Port X08: Diagnostic Data	r

r ... read only

Additional diagnostic and status details are provided in the following register areas:

- "Input Data" area: →**Read input data of several IO-Link ports** (→ p. 84)
- "Single Port Access" area: →**Read input data of several IO-Link ports** (→ p. 84)

9.3.9 Read device information

34451

The user can read device information using the FC43.
The AL1342 supports the following data records ("Read Device ID code"):

- Basic Device Identification (0x01): contained data objects: → Modbus TCP specification
- Regular Device Identification (0x02): contained data objects: → Modbus TCP specification
- Specific Device Identification (0x04): contained data objects:

Object ID	Object name / description	Data type	Possible values
0x00	VendorName	ASCII string	ifm electronic
0x01	ProductCode	ASCII string	1342
0x02	MajorMinorRevision	ASCII string	e.g. V1.001
0x03	VendorURL	ASCII string	www.ifm.com
0x04	ProductName	ASCII string	IO-Link Master DL MOD 8P IP67
0x05	ModelName	ASCII string	1342
0x06	UserApplicationName	ASCII string	MODBUS IO-Link master

9.3.10 Control IO-Link master

34458

The user can control the IO-Link master using the following acyclic commands:

- "Reboot": →**Command 0x40 – Reboot** (→ p. 125)
- "Factory Reset": →**Command 0x50 – Factory Reset** (→ p. 126)

The commands use the process mechanisms of the acyclic command channel (→**Use acyclic services** (→ p. 91)).

9.3.11 Configure IO-Link devices

33881

The IO-Link master supports the configuration of the connected IO-Link devices from the Modbus TCP projection software. The parameters of an IO-Link device are set via IO-Link index and subindex. The number of the configurable parameters depends on the connected IO-Link device.

Available parameters of the IO-Link devices: → IO Device Description (IODD) of the IO-Link device

The user can read and write IO-Link index and subindex using the following methods:

- Acyclic communication: → **Use acyclic services** (→ p. 91)

9.3.12 Modbus TCP: Programmers' notes

Content

Rules for accessing the Modbus registers 90
Supported function codes....90
Note: Exception Codes....90
Use acyclic services 91

33665

Rules for accessing the Modbus registers

54703

The following general rules apply for access to the Modbus registers:

- ► Only use the valid function codes to read or write Modbus registers (→ **Supported function codes** (→ p. 90)).
- ► After every read or write access check the validity of the transmitted data (→ **Mapping: PQI** (→ p. 114)).

Supported function codes

34440

The AL1342 supports the following function codes for read and/or write access to the Modbus register:

Function code	Function name / description
03 (0x03)	Read Multiple Registers
04 (0x04)	Read Input Register
06 (0x06)	Write Single Register
16 (0x10)	Write Multiple Registers
23 (0x17)	Read/Write Multiple Registers
43 (0x2B)	Read Device Identification

Detailed information about the function codes: → MODBUS-TCP specification

Note: Exception Codes

54689

If processing the function code request was without error, the response message will have the following content:

- Response Function Code: Request Function Code
- Response Data: Requested data

If an error occurs while accessing the registers, the AL1342 replies with an error code instead of the function code. The response message has the following content:

- Response Function Code: Error Code (= Request Function Code + 0x80)
- Response Data: Exception Code

The following exception codes are available:

Exception Code	Name	Description
0x1	ILLEGAL FUNCTION	Invalid function code (Modbus function not implemented)

0x2	ILLEGAL DATA ADDRESS	Invalid data address (wrong address or length)
0x3	ILLEGAL DATA VALUE	Invalid data value (invalid parameter; wrong number of register)
0x4	SERVER DEVICE FAILURE	Unrecoverable error in Modbus server while processing

Use acyclic services

34471

The AL1342 has a command interface to execute acyclic commands. A cyclic command consists of a request and a response.

Register	Contents	Access
500	Command Request Channel (Fieldbus PLC >>> IO-Link Master)	r/w
0	Command Response Channel (IO-Link Master >>> Fieldbus PLC)	r

Structure of the acyclic command channel: →**Acyclic Command Channel** (→ p. 114)

General procedure of the acyclic communication:

1 Write Command Request

► In the request channel: Write required data (except for [User ID]).

> Write required [User ID].

> Changed [User ID] signals a new command.

> In the response channel: registers are reset to 0.

> Acyclic command channel is blocked.

> Processing of the command is started.

2 Check status

► In the response channel: Check [Command Status] register.

> If [Command Status] <> 0: continue with step 3

> If [Command Status] == 0: repeat step 2.

3 Read Command Response

► In the response channel: read returned user data.

> Acyclic command channel is released.

10 Operation

Content

Using web-based management....92

34061

10.1 Using web-based management

61181

The device has an integrated web server The web server generates a website with the following data:

- Status information of the ports
- Access to product page of connected IO-Link devices (only ifm devices)
- Diagnostic information of the device
- Version information of the installed firmware components

To access the web interface of the IO-Link master:

► Connect the IO-Link master to the laptop / PC via the IoT port.
► Optional: Check the IP settings of the IoT interface.
► Start web browser.
► In the address field of the web browser, enter the IP address of the IoT interface and confirm with [ENTER].
> The web browser shows the website with the status and diagnostic information of the device.

11 Maintenance, repair and disposal

Content

Cleaning process....93
Updating the firmware....93
Replacing IO-Link device....93

51990

The operation of the unit is maintenance-free.

► Dispose of the unit in an environmentally friendly way in accordance with the applicable national regulations when it is no longer used.

11.1 Cleaning process

51991

► Clean the surface of the unit when necessary.
► Do not use any caustic cleaning agents for this!
► In case of severe soiling, use a damp cloth.
► Do not use any caustic cleaning agents for this!

11.2 Updating the firmware

61183

The firmware of the IO-Link master can be updated via the IoT Core Visualizer↘→ **Updating the firmware** (→ p. 80, → p. 93)).

11.3 Replacing IO-Link device

34182

To replace an IO-Link device:

Requirement:

> New IO-Link device is with factory settings.
> New IO-Link device supports IO-Link standard 1.1 or higher.

1 Set data storage

► Set the following parameters of the IO-Link port
▪ Set Validation and Data Storage to [Type compatible V1.1 device with Restore] or [Type compatible V1.1. device with Backup + Restore]
▪ Set correct values to [Vendor ID] and [Device ID] according to properties of the IO-Link device.
► Save changes.

2 Replace IO-Link device

► Disconnect old IO-Link device from IO-Link master.
► Connect new IO-Link device with the same IO-Link port of the AL1342.
> IO-Link master copies parameter values from the data memory to the new IO-Link device.

12 Factory settings

34509

In the factory settings, the device has the following parameter settings:

Parameter	Factory setting
[IP address] (Modbus TCP)	192.168.1.250
[Subnet mask] (Modbus TCP)	255.255.255.0
[IP gateway address] (Modbus TCP)	0.0.0.0
[IP address] (IoT interface)	169.254.X.X
[Subnet mask] (IoT interface)	255.255.0.0
[IP gateway address] (IoT interface)	0.0.0.0
[Modbus TCP name]	blank
Data memory (Data Storage)	empty

13 Accessories

33870

List of accessories of AL1342: → www.ifm.com > Product page > Accessories

14 Appendix

Content

Technical data 97
Modbus TCP 101
ifm IoT Core 127

33879

14.1 Technical data

Content

Application 97
Electrical data 97
Inputs / outputs 97
Inputs 98
Outputs 98
Interfaces 98
Operating conditions 99
Approvals / tests 99
Mechanical data 99
Electrical connection 100

34188

14.1.1 Application

33878

Application	
Application	I/O modules for field applications
Daisy-chain function	Fieldbus interface

14.1.2 Electrical data

33808

Electrical data	
Operating voltage [V]	20...30 DC; (US; to SELV/PELV)
Current Consumption [mA]	300...3900; (US)
Protection class	III
Sensor supply US	
Max. current load total [A]	3.6

14.1.3 Inputs / outputs

34068

Inputs / outputs	
Total number of inputs and outputs	16; (configurable)
Number of Inputs and Outputs	Number of digital inputs: 16; Number of digital outputs: 8

14.1.4 Inputs

34069

Inputs	
Number of digital inputs	16; (IO-Link Port Class A: 8 x 2)
Switching level high [V]	11...30
Switching level low [V]	0...5
Digital inputs protected against short circuits	yes

14.1.5 Outputs

34053

Outputs	
Number of digital outputs	8; (IO-Link Port Class A: 8 x 1)
Max. current load per output [mA]	300
Short-circuit protection	yes

14.1.6 Interfaces

34389

Interfaces	
Communication interface	Ethernet; IO-Link
Communication interface	IO-Link; TCP/IP; Modbus TCP
Ethernet	
Transmission standard	10Base-T; 100Base-TX
Transmission rate [MBit/s]	10; 100
Protocol	TCP/IP; Modbus TCP
Factory settings	▪ IP address: 192.168.1.250 ▪ Subnet mask: 255.255.255.0 ▪ Gateway IP address: 0.0.0.0 ▪ MAC address: see type label
IO-Link master	
Type of transmission	COM 1 / COM 2 / COM 3
IO-Link revision	V1.1
Number of ports Class A	8
IoT interface	
Transmission standard	10Base-T; 100Base-TX
Transmission rate [Mbits/s]	10; 100
Protocol	DCP, DCHP, Auto IP
Factory settings	▪ IP address: 169.254.X.X ▪ Subnet mask: 255.255.0.0 ▪ Gateway IP address: 0.0.0.0 ▪ MAC address: see type label

14.1.7 Operating conditions

34062

Operating conditions	
Applications	Indoor use
Ambient temperature [°C]	-25...60
Storage temperature [°C]	-25...85
Max. perm. relative air humidity [%]	90
Max. height above sea level [m]	2000
Protection rating	IP 65; IP 66; IP 67
Pollution Degree	2

14.1.8 Approvals / tests

33877

Approval / tests	
EMC	▪ EN 61000-6-2 ▪ EN 61000-6-4
MTTF [Years]	90

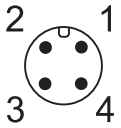
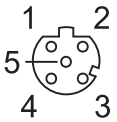
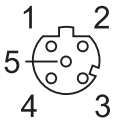
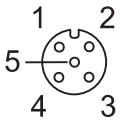
14.1.9 Mechanical data

34050

Mechanical data	
Weight [g]	403
Materials	Housing: PA; socket: brass nickel-plated

14.1.10 Electrical connection

33806

Voltage supply IN X31		
Plug and socket connection	M12	
Wiring		1: + 24 V DC (US) 2: - 3: GND (US) 4: -
Ethernet IN / OUT X21, X22		
Plug and socket connection	M12	
Wiring		1: TX + 2: RX + 3: TX - 4: RX - 5: -
IoT X23		
Plug and socket connection	M12	
Wiring		1: TX + 2: RX + 3: TX - 4: RX - 5: -
Process connection IO-Link ports Class A X01...X08		
Plug and socket connection	M12	
Wiring		1: Sensor supply (US) L+ 2: DI 3: Sensor supply (US) L- 4: C/Q IO-Link 5: -

14.2 Modbus TCP

Content

Register 102
Acyclic commands 118

33674

14.2.1 Register

Content

Configuration Area....103
Diagnostic data....105
Input Data....108
Output Data....111
Single Port Access....113
Acyclic Command Channel....114

34442

The AL1342 saves the configuration data, process data and status/diagnostic data in Modbus registers.

Configuration Area

34475

Register	Content	
	Bits 8-15	Bits 0-7
8998	Access Rights	Process Data Length
8999	reserved	Byte Swap
9000	Port X01: Port Configuration (→ **Mapping: port configuration** (→ p. 104))	
9006	Port X02: Port Configuration (→ **Mapping: port configuration** (→ p. 104))	
9012	Port X03: Port Configuration (→ **Mapping: port configuration** (→ p. 104))	
9018	Port X04: Port Configuration (→ **Mapping: port configuration** (→ p. 104))	
9024	Port X05: Port Configuration (→ **Mapping: port configuration** (→ p. 104))	
9030	Port X06: Port Configuration (→ **Mapping: port configuration** (→ p. 104))	
9036	Port X07: Port Configuration (→ **Mapping: port configuration** (→ p. 104))	
9042	Port X08: Port Configuration (→ **Mapping: port configuration** (→ p. 104))	

Legend:

▪ [Access Rights]	The access rights to the parameter data, process data and the events/diagnostic messages of the IO-Link master as well as the connected IO-Link devices.	1 byte	0x00	Modbus TCP + IoT
			0x01	Modbus TCP + IoT (ro)
			0x02	Modbus TCP only
			0x03	Keep setting (default)
▪ [Process Data Length]	Length of the process input data and process output data	1 byte	0x00	2 bytes input / 2 bytes output data ▪ Input Data: 14 Words ▪ Output Data: 10 Words
			0x01	4 bytes input / 4 bytes output data ▪ Input Data: 22 Words ▪ Output Data: 18 Words
			0x02	8 bytes input / 8 bytes output data ▪ Input Data: 38 Words ▪ Output Data: 34 Words
			0x03	16 bytes input / 16 bytes output data ▪ Input Data: 70 Words ▪ Output Data: 66 Words
			0x04	32 bytes input / 32 bytes output data ▪ Input Data: 134 Words ▪ Output Data: 130 Words
▪ [Byte Swap]	Sequence of bytes in the data word	1 byte	0x00	as Array of Bytes
			0x01	as integer16 value; when process data is updated, bytes will be exchanged (input data and uouput data)
▪ [Port Configuration]	Configuration of the IO-Link port	12 bytes	→ **Mapping: port configuration** (→ p. 104)	

Mapping: port configuration

34478

Bits 8-15	Bits 0-7
Port Mode	Master Cycle Time
reserved	Validation ID
Vendor ID	
reserved	Device ID (MSB)
Device ID	Device ID (LSB)
Failsafe Mode -- IO-Link	Failsafe Mode -- Pin 4 (DO)

Legend:

▪ [Port Mode]	Operating mode of the IO-Link port	1 byte	0x00	deactivated
			0x01	Digital input (DI)
			0x02	Digital output (DO)
			0x03	IO-Link
▪ [Master Cycle Time]	Cycle time of the data transmission between the IO-Link master and the IO-Link device	1 byte	0x00	As fast as possible
			0x01	2 milliseconds
			0x02	4 milliseconds
			0x03	8 milliseconds
			0x04	16 milliseconds
			0x05	32 milliseconds
			0x06	64 milliseconds
			0x07	128 milliseconds
▪ [Validation ID]	Supported IO-Link standard and behaviour of the IO-Link master when connecting new IO-Link devices to the IO-Link port	1 byte	0x00	No validation
			0x01	V1.0 device
			0x02	V1.1 device
			0x03	V1.1 device with Backup + Restore
			0x04	V1.1 device with Backup
▪ [Vendor ID]	Vendor ID of the manufacturer of the device on the IO-Link port	2 bytes	0x0000...0xFFFF	
▪ [Device ID]	Device ID of the device on the IO-Link port Device ID = 0x123456 ▪ Device ID (MSB) = 0x12 ▪ Device ID = 0x34 ▪ Device ID (LSB) = 0x56	3 bytes	pro Byte: 0x00...0xFF	
▪ [Failsafe Mode -- IO-Link]	Fail-safe mode for output data when the Modbus TCP connection is interrupted	1 byte	0x00	No Failsafe
			0x01	Failsafe Reset Value
			0x02	Failsafe Old Value
			0x03	Failsafe with Pattern
▪ [Failsafe Mode -- Pin 4 (DO)]	Fail-safe value for the operating mode "digital output (DO)	1 byte	0x00	Failsafe Reset Value
			0x01	Failsafe Old Value
			0x02	Failsafe Set Value

Diagnostic data

34468

Register	Contents	
	Bits 8-15	Bits 0-7
30	reserved	Port X01: → **Mapping: Diagnostics** (→ p. 106)
31	Port X01: Vendor ID	
32	reserved	Port X01: Device ID (MSB)
33	Port X01: Device ID	Port X01: Device ID (LSB)
34...39	Port X01: Events (→**Mapping: events** (→ p. 107))	
40	Port X02: Diagnostic data (Mapping: → Port X01 - register 30...39)	
50	Port X03: Diagnostic data (Mapping: → Port X01 - register 30...39)	
60	Port X04: Diagnostic data (Mapping: → Port X01 - register 30...39)	
70	Port X05: Diagnostic data (Mapping: → Port X01 - register 30...39)	
80	Port X06: Diagnostic data (Mapping: → Port X01 - register 30...39)	
90	Port X07: Diagnostic data (Mapping: → Port X01 - register 30...39)	
100	Port X08: Diagnostic data (Mapping: → Port X01 - register 30...39)	

Legend:

- [Vendor ID] Vendor ID of the manufacturer of the device on the IO-Link port — 2 bytes — 0x0000...0xFFFF
- [Device ID] Device ID of the device on the IO-Link port — 3 bytes — per byte: 0x00...0xFF

  Device ID = 0x123456
  - Device ID (MSB) = 0x12
  - Device ID = 0x34
  - Device ID (LSB) = 0x56

Mapping: Diagnostics

34504

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
reserved	Wrong Length PD OUT	Wrong Length PD IN	Cycle time	Wrong Vendor ID/ Device ID	reserved	reserved	IOL mode

Legend:

▪ [IOL Mode]	Operating type of the IO-Link port	1 bit	0x0	Other
			0x1	IO-Link
▪ [Wrong Vendor ID/ Device ID]	Verification if the current and configured vendor ID and device ID are identical	1 bit	0x0	OK
			0x1	No match
▪ [Wrong Cycle Time]	Verification if the current and configured cycle time are identical	1 bit	0x0	OK
			0x1	No match
▪ [Wrong Length PD IN]	Verification if the size of the received input data are identical with the configured size	1 bit	0x0	OK
			0x1	Configured size too small
▪ [Wrong Length PD OUT]	Verification if the size of the sent output data is identical with the size expected by the IO-Link device	1 bit	0x0	OK
			0x1	Configured size too small

Mapping: events

34481

Bit															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
reserved								Event 1: Mode		Event 1: Type		Event 1: Src	Event 1: Instance		
Event 1: Code															
reserved								Event 2: Mode		Event 2: Type		Event 2: Src	Event 2: Instance		
Event 2: Code															
reserved								Event 3: Mode		Event 3: Type		Event 3: Src	Event 3: Instance		
Event 3: Code															

Legend:

▪ [Event m: Mode]	Mode: mode of the event	2 bits	0x0	reserved
			0x1	One-time event
			0x2	Event has disappeared
			0x3	Event has appeared
▪ [Event m: Type]	Type: category of the event	2 bits	0x0	reserved
			0x1	Notification
			0x2	Warning
			0x3	Error
▪ [Event m: Src]	Source: source of the event	1 bit	0x0	IO-Link Device
			0x1	IO-Link Master
▪ [Event m: Instance]	Type: trigger of the event	2 bits	0x0	unknown
			0x1...0x3	reserved
			0x4	Application
			0x5...0x7	reserved
▪ [Event m: Code]	Code: event code; depends on the device	2 bytes	→ IODD description of the IO-Link device	

Input Data

34448

Register	Contents	
	Bits 8-15	Bits 0-7
197	Port X01...X04: Digital Input - Pin 2 / Pin 4 (DI) (→ **Mapping: digital input data** (→ p. 109))	
198	Port X01...X04: Diagnostic Information (→ **Mapping: diagnostic information** (→ p. 109))	
199	Port X01...X04: Status Information IO-Link Ports (→**Mapping: Status information IO-Link ports** (→ p. 110))	
200	Port X01...X04: Compact Input Block (4n bytes) (→ **Mapping: Compact Input Block** (→ p. 110))	
297	Port X05...X08: Digital Input - Pin 2 / Pin 4 (DI) (→ **Mapping: digital input data** (→ p. 109))	
298	Port X05...X08: Diagnostic Information (→ **Mapping: diagnostic information** (→ p. 109))	
299	Port X05...X08: Status Information IO-Link Ports (→**Mapping: Status information IO-Link ports** (→ p. 110))	
300	Port X05...X08: Compact Input Block (4n bytes) (→ **Mapping: Compact Input Block** (→ p. 110))	

Legend:

▪ [Digital Input - Pin 2 / Pin 4 (DI)]	Digital input data Pin 2 / Pin 4 (operating mode DI) of 4 IO-Link ports	2 bytes	
▪ [Diagnostic Information]	Diagnostic information	2 bytes	
▪ [Status Information IO-Link Ports]	Status information of the IO-Link ports	2 bytes	
▪ [Compact Input Block (4n Bytes)]	Input data (operating mode IO-Link) of 4 IO-Link ports n = [2,4,8,16,32]; is determined by parameter [Process Data Length] (→**Configuration Area** (→ p. 103))	4n byte	per byte: 0x00...0xFF

Mapping: digital input data

34483

Register 197:

Bit															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
res.	res.	res.	res.	X04: pin 2	X03: pin 2	X02: pin 2	X01: pin 2	res.	res.	res.	res.	X04: pin 4	X03: pin 4	X02: pin 4	X01: pin 4

Register 297:

Bit															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
res.	res.	res.	res.	X08: pin 2	X07: pin 2	X06: pin 2	X05: pin 2	res.	res.	res.	res.	X08: pin 4	X07: pin 4	X06: pin 4	X05: pin 4

Legend:

▪ [pin 4]	Signal level on pin 4 of the IO-Link port (DI)	1 bit	0x0	LOW
			0x1	HIGH
▪ [pin 2]	Signal level on pin 2 of the IO-Link ports (if used)	1 bit	0x0	LOW
			0x1	HIGH

Mapping: diagnostic information

34502

Register 198:

Bit															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
res.	res.	res.	res.	X04: SC/ OL	X03: SC/ OL	X02: SC/ OL	X01: SC/ OL	res.	res.	res.	res.	res.	res.	SENS PWR	AUX PWR

Register 298:

Bit															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
res.	res.	res.	res.	X08: SC/ OL	X07: SC/ OL	X06: SC/ OL	X05: SC/ OL	res.	res.	res.	res.	res.	res.	SENS PWR	AUX PWR

Legend:

▪ [SC/OL]	Short Circuit / Overload: indicates the occurrence of a short-circuit or overvoltage on the IO-Link port	1 bit	0x0	error-free
			0x1	Short-circuit or overvoltage
▪ [SENS PWR]	Sensor Power: indicates the status of the supply voltage US	1 bit	0x0	US not available
			0x1	US available
▪ [AUX PWR]	Auxiliary Power: indicates the supply voltage UA	1 bit	0x0	UA not available
			0x1	UA available

Mapping: Status information IO-Link ports

34490

Register 199:

Bit															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
res.	res.	res.	res.	X04: Data invalid	X03: Data invalid	X02: Data invalid	X01: Data invalid	res.	res.	res.	res.	X04: Dev Not Conn	X03: Dev Not Conn	X02: Dev Not Conn	X01: Dev Not Conn

Register 299:

Bit															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
res.	res.	res.	res.	X08: Data invalid	X07: Data invalid	X06: Data Invalid	X05: Data invalid	res.	res.	res.	sres.	X08: Dev Not Conn	X07: Dev Not Conn	X06: Dev Not Conn	X05: Dev Not Conn

Legend:

▪ [Data invalid]	indicates the status of the process input data on the IO-Link port	1 bit	0x0	data valid
			0x1	data invalid
▪ [Dev Not Conn]	Device Connected: indicates the connection to the device on the IO-Link port	1 bit	0x0	available
			0x1	not available

Mapping: Compact Input Block

54566

IO-Link port	Register area				
	2 bytes/port (n = 2)	**4 bytes/port (n = 4)**	**8 bytes/port (n = 8)**	**16 bytes/port (n = 16)**	**32 bytes/port (n = 32)**
Port X01	200	200...201	200...203	200...207	200...215
Port X02	201	202...203	204...207	208...215	216...231
Port X03	202	204...205	208...211	216...223	232...247
Port X04	203	206...207	212...215	224...231	248...263
Port X05	300	300...301	300...303	300...307	300...315
Port X06	301	302...303	304...307	308...315	316...331
Port X07	302	304...305	308...311	316...323	332...347
Port X08	303	306...307	312...315	324...331	348...363

Output Data

34446

Register	Contents	
	Bits 8-15	Bits 0-7
599	Port X01...X04: Digital Output - pin 4 (DO) (→ **Mapping: Digital output data** (→ p. 112))	
600	Port X01...X04: Compact Output Block (4n bytes) (→ **Mapping: Compact Output Block** (→ p. 112))	
699	Port X05...X08: Digital Output - pin 4 (DO) (→ **Mapping: Digital output data** (→ p. 112))	
700	Port X05...X08: Compact Output Block (4n bytes) (→ **Mapping: Compact Output Block** (→ p. 112))	

Legend:

▪ [Digital Output - pin 4 (DO)]	Digital output data - pin 4 (operating mode DO) of 4 IO-Link ports	2 bytes	
▪ [Compact Output Block (4n Bytes)	Output data (operating mode IO-Link) of 4 IO-Link ports n = [2,4,8,16,32]; is determined by parameters [Process Data Length] (→**Configuration Area** (→ p. 103))	4n bytes	per byte: 0x00...0xFF

Mapping: Digital output data

34482

Register 599:

Bit															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
res.	res.	res.	res.	res.	res.	res.	res.	res.	res.	res.	res.	X04: pin 4	X03: pin 4	X02: pin 4	X01: pin 4

Register 699:

Bit															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
res.	res.	res.	res.	res.	res.	res.	res.	res.	res.	res.	res.	X08: pin 4	X07: pin 4	X06: pin 4	X05: pin 4

Legend:

- [pin 4] Signal level on pin 4 of the IO-Link port (DO) 1 bit 0x0 LOW
  0x1 HIGH

Mapping: Compact Output Block

54569

IO-Link port	Register area				
	2 bytes/port (n = 2)	**4 bytes/port (n = 4)**	**8 bytes/port (n = 8)**	**16 bytes/port (n = 16)**	**32 bytes/port (n = 32)**
Port X01	600	600...601	600...603	600...607	600...615
Port X02	601	602...603	604...607	608...615	616...631
Port X03	602	604...605	608...611	616...623	632...647
Port X04	603	606...607	612...615	624...631	648...663
Port X05	700	700...701	700...703	700...707	700...715
Port X06	701	702...703	704...707	708...715	716...731
Port X07	702	704...705	708...711	716...723	732...747
Port X08	703	706...707	712...715	724...731	748...763

Single Port Access

34445

Register	Contents	
	Bits 8-15	**Bits 0-7**
1000	Port X01: Digital Data - pin 2	Port X01: Digital Input - pin 4 (DI)
1001	Port X01: → **Mapping: Status information** (→ p. 114)	Port X01: → **Mapping: PQI** (→ p. 114)
1002	Port X01: Input Data IO-Link (n Bytes)	
1100	reserved	Port X01: Digital Output - pin 4 (DO)
1101	Port X01: Output Data IO-Link (n Bytes)	
2000	Port X02: Single Port Access (Mapping: → Port X01 - Register 1000...1101)	
3000	Port X03: Single Port Access (Mapping: → Port X01 - Register 1000...1101)	
4000	Port X04: Single Port Access (Mapping: → Port X01 - Register 1000...1101)	
5000	Port X05: Single Port Access (Mapping: → Port X01 - Register 1000...1101)	
6000	Port X06: Single Port Access (Mapping: → Port X01 - Register 1000...1101)	
7000	Port X07: Single Port Access (Mapping: → Port X01 - Register 1000...1101)	
8000	Port X08: Single Port Access (Mapping: → Port X01 - Register 1000...1101)	

Legend:

▪ [Digital Input - pin 2]	Signal level pin 2	1 byte	0x00	LOW
			0x01	HIGH
▪ [Digital Input - pin 4 (DI)]	Signal level pin 4 (operating mode DI)	1 byte	0x00	LOW
			0x01	HIGH
▪ [Input Data IO-Link (n Bytes)]	Input data (operating mode IO-Link) (n bytes) n = [2,4,8,16,32]; is determined by the parameter [Process Data Length] (→ **Configuration Area** (→ p. 103))	n bytes	per byte: 0x00...0xFF	
▪ [Digital Output - pin 4(DO)]	Signal level pin4 (operating mode DO)	1 byte	0x00	LOW
			0x01	HIGH
▪ [Output Data IO-Link (n Bytes)	Output data (operating mode IO-Link) (n bytes) n = [2,4,8,16,32]; is determined by the parameter [Process Data Length] (→ **Configuration Area** (→ p. 103))	n bytes	per byte: 0x00...0xFF	

Mapping: Status information

34480

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
reserved	reserved	reserved	reserved	reserved	SC / OL	SENS PWR	AUX PWR

Legend:

- [SC / OL] Short Circuit / Overload: indicates the occurrence of a short-circuit or overvoltage on the IO-Link port — 1 bit — 0x0 No error; 0x1 Short-circuit or overvoltage
- [SENS PWR] Sensor Power: indicates the status of the supply voltage US — 1 bit — 0x0 No error; 0x1 Error
- [AUX PWR] Auxiliary Power: indicates the supply voltage UA — 1 bit — 0x0 UA present and no error; 0x1 Error

Mapping: PQI

34479

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
reserved	Wrong Length PD OUT	Wrong Length PD IN	Wrong Cycle Time	Wrong VID/DID	Invalid Data	Dev Not Conn	IOL Mode

Legend:

- [IOL Mode] Operating type of the IO-Link port — 1 bit — 0x0 Other; 0x1 IO-Link
- [Dev Not Conn] Connection between IO-Link Device and IO-Link port — 1 bit — 0x0 connected.; 0x1 not connected
- [Invalid Data] Status of the process input data on the IO-Link port — 1 bit — 0x0 valid data; 0x1 invalid data
- [Wrong VID/DID] Verification whether the current and configured vendor ID and device ID are identical — 1 bit — 0x0 OK; 0x1 no match
- [Wrong Cycle Time] Verification whether the current and configured cycle time are identical — 1 bit — 0x0 OK; 0x1 no match
- [Wrong Length PD IN] Verification whether the size of the received input data are identical with the configured size — 1 bit — 0x0 OK; 0x1 Configured size too small
- [Wrong Length PD OUT] Verification whether the size of the sent output data is identical with the size expected by the IO-Link device — 1 bit — 0x0 OK; 0x1 Configured size too small

Acyclic Command Channel

34449

The following Modbus registers are available for acyclic data transmission:

Register	Contents	
	Bits 8-15	Bits 0-7
500	Command Request Channel (→ **Request channel** (→ p. 115))	
0	Command Response Channel (→ **Response channel** (→ p. 116))	

Legend:

- [Command Request Channel] Area for transmission of command request (fieldbus PLC >>> IO-Link master) 44 bytes
- [Command Response Channel] Area for transmission of command response (IO-Link master >>> fieldbus PLC) 44 bytes

Request channel

34450

Register	Contents	
	Bits 8-15	Bits 0-7
500	Port No.	
501	Index	
502	Subindex	
503	Command	User ID
504	Data Length (Number of Bytes)	
505	Data (byte 0)	Data (byte 1)
...	...	...
521	Data (byte 32)	Data (byte 33)

Legend:

▪ [Port No.]	Number of the IO-Link port	1 Word	0x0001	Port X01
			0x0002	Port X02
			...	...
			0x0008	Port X08
▪ [Index]	Index of the IO-Link object	1 Word	0x0000...0xFFFF	
▪ [Subindex]	Subindex of the IO-Link object	1 Word	0x0000...0xFFFF	
▪ [Command]	Command number	1 byte	0x01	Read
			0x02	Write
▪ [User ID]	ID to identify the command	1 byte	0x00...0xFF	
▪ [Data Length (Number of Bytes)]	Number of bytes that contain relevant user data (is only evaluated for Command = 0x02)	1 Word	0x0000	0 bytes
			...	...
			0x0022	34 bytes
▪ [Data (Byte n)]	user data	n bytes	per byte: 0x00 ... 0xFF	

Response channel

34453

Register	Contents	
	Bits 8-15	Bits 0-7
0	Port No.	
1	Index	
2	Subindex	
3	Command	User ID
4	Result	
5	Data Length (Number of Bytes)	
6	Data (byte 0) / Error Code	Data (byte 1) / Additional Code
...	...	...
21	Data (byte 30)	Data (byte 31)

Legend:

▪ [Port No.]	Number of the IO-Link port	1 Word	0x0001	Port X01
			0x0002	Port X02
			...	...
			0x0008	Port X08
▪ [Index]	Index of the IO-Link object	1 Word	0x0000...0xFFFF	
▪ [Subindex]	Subindex of the IO-Link object	1 Word	0x0000...0xFFFF	
▪ [Command]	Command number	1 byte	0x01	Read
			0x02	Write
▪ [User ID]	reflected User ID from request channel	1 byte	0x00...0xFF	
▪ [Result]	Status of the command processing	1 Word	0x0000	OK
			0x000F	OK, but data length too long (only with [Command] = 0x02)
			0x00FF	Error
▪ [Data Length (Number of Bytes)]	Number of bytes that contain relevant user data	1 Word	0x0000	0 bytes
			...	....
			0x0020	32 bytes
▪ [Error Code	Error Code	1 byte	→ **Error codes** (→ p. 117)	
▪ [Additional Code]	Additional error codes	1 byte	→ **Additional Codes** (→ p. 117)	
▪ [Data (Byte n)]	User data (byte n)	n bytes	per byte: 0x00...0xFF	

Error codes

34342

Error code	Description
0x71	Service not available (unknown command has been sent to the IO-Link port)
0x72	Port blocked (another cyclic process accesses the IO-Link port)
0x73	Forbidden (access rights don't allow command processing)
0x74	Invalid data (wrong parameter has been sent in the command)
0x76	Wrong port (wrong port number)
0x77	Wrong port function (wrong port function or wrong parameter has been sent to the device)
0x78	Invalid length (set length is > 0x20)
0x80	Error in the device application; observe additional code (→ **Additional Codes** (→ p. 117))

Additional Codes

54584

Code	Name	Description
0x00	APP_DEV	Device application error - no details
0x11	IDX_NOTAVAIL	Index not available
0x12	SUBIDX_NOTAVAIL	Subindex not available
0x20	SERV_NOTAVAIL	Service temporarily not available
0x21	SERV_NOTAVAIL_LOCCTRL	Service temporarily not available - local control
0x22	SERV_NOTAVAIL_DEVCTRL	Service temporarily not available - device control
0x23	IDX_NOT_WRITEABLE	Access denied
0x30	PAR_VALOUTOFRNG	Parameter value out of range
0x31	PAR_VALGTLIM	Parameter value above limit
0x32	PAR_VALLTLIM	Parameter value below limit
0x33	VAL_LENOVRRUN	Parameter length overrun
0x34	VAL_LENUNDRUN	Parameter length underrun
0x35	FUNC_NOTAVAIL	Function not available
0x36	FUNC_UNAVAILTEMP	Function temporarily not available
0x40	PAR_SETINVALID	Invalid parameter set
0x41	PAR_SETINCONSIST	Inconsistent parameter set
0x82	APP_DEVNOTRDY	Application not ready

Additional Codes are only available, if Error Code = 0x80 (→ **Error codes** (→ p. 117))

14.2.2 Acyclic commands

Content

Command 0x10 – Set mode....119
Command 0x20 – Set validation ID / data storage....121
Command 0x30 – Set fail-safe data pattern....123
Command 0x40 – Reboot....125
Command 0x50 – Factory Reset....126

34331

Command 0x10 – Set mode

34322

The command changes the operating mode of an IO-Link port of the AL1342.

Corresponding parameter: [Port Mode] (→ **Mapping: port configuration** (→ p. 104))

Command request

34500

Register	Contents	
	Bits 8-15	**Bits 0-7**
500	Port No.	
501	reserved	
502	reserved	
503	0x10	User ID
504	reserved	
505	Target Mode	
506 ... 521	reserved	

Legend:

- [Port No.] Number of the IO-Link port — 1 word — 0x0001 Port X01; 0x0002 Port X02; ... ...; 0x0008 Port X08
- [User ID] ID to identify the command — 1 byte — 0x00 .. 0xFF
- [Target Mode] Operating type of the IO-Link port — 1 word — 0x0000 deactivated; 0x0001 digital input (DI); 0x0002 digital output (DO); 0x0003 IO-Link

Command response

34506

Register	Contents	
	Bits 8-15	Bits 0-7
0	Port No.	
1	reserved	
2	reserved	
3	0x10	User ID
4	Result	
5	Data Length (Number of Bytes)	
6	reserved / Error Code	Target Mode / Additional Code
7 ... 21	reserved	

Legend:

▪ [Port No.]	Number of the IO-Link port	1 word	0x0001	Port X01
			0x0002	Port X02
			...	...
			0x0008	Port X08
▪ [User ID]	reflected User ID from request channel	1 byte	0x00 .. 0xFF	
▪ [Result]	Status of the command processing	1 byte	0x00	OK
			0xFF	Error
▪ [Data Length (Number of Bytes)]	Number of bytes that contain relevant user data	1 word	0x0001	1 byte
			0x0002	2 bytes
▪ [Target Mode]	Operating type of the IO-Link port	1 byte	0x00	deactivated
			0x01	digital input (DI)
			0x02	digital output (DO)
			0x03	IO-Link
▪ [Error Code]	Error ID	1 byte	→ **Error codes** (→ p. 117)	
▪ [Additional Code]	Additional error codes	1 byte	→ **Additional Codes** (→ p. 117)	

Command 0x20 – Set validation ID / data storage

34321

The command sets the behaviour of the IO-Link master when connecting a new IO-Link device to an IO-Linkport of the device.

Corresponding parameter: [Validation ID] (→ **Mapping: port configuration** (→ p. 104))

Command request

34495

Register	Contents	
	Bits 8-15	**Bits 0-7**
500	Port No.	
501	reserved	
502	reserved	
503	0x20	User ID
504	reserved	
505	Validation ID	
506 ... 521	reserved	

Legend:

▪ [Port No.]	Number of the IO-Link port	1 word	0x0001	Port X01
			0x0002	Port X02
			...	...
			0x0008	Port X08
▪ [User ID]	ID to identify the command	1 byte	0x00 .. 0xFF	
▪ [Validation ID]	Supported IO-Link standard and behaviour of the IO-Link master when connecting new IO-Link devices to the IO-Link port	1 word	0x0000	no validation
			0x0001	V1.0 device
			0x0002	V1.1 device
			0x0003	V1.1 device, backup + restore
			0x0004	V1.1 device, backup

Command response

34497

Register	Contents	
	Bits 8-15	Bits 0-7
0	Port No.	
1	reserved	
2	reserved	
3	0x10	User ID
4	Result	
5	Data Length (Number of Bytes)	
6	reserved / Error Code	Validation ID / Additional Code
7 ... 21	reserved	

Legend:

▪	[Port No.]	Number of the IO-Link port	1 word	0x0001	Port X01
				0x0002	Port X02
				...	...
				0x0008	Port X08
▪	[User ID]	reflected user ID from request channel	1 byte	0x00 .. 0xFF	
▪	[Result]	Status of the command processing	1 byte	0x00	OK
				0xFF	Error
▪	[Data Length (Number of Bytes)]	Number of bytes that contain relevant user data	1 word	0x0001	1 byte
				0x0002	2 bytes
▪	[Validation ID]	Supported IO-Link standard and behaviour of the IO-Link master when connecting new IO-Link devices to the IO-Link port	1 byte	0x00	no validation
				0x01	V1.0 device
				0x02	V1.1 device
				0x03	V1.1 device, backup + restore
				0x04	V1.1 device, backup
▪	[Error Code]	Error ID	1 byte	→ **Error codes** (→ p. 117)	
▪	[Additional Code]	additional error codes	1 byte	→ **Additional Codes** (→ p. 117)	

Command 0x30 – Set fail-safe data pattern

34379

The command sets the behaviour of the outputs when the Modbus TCP connection and the corresponding fail-safe values are interrupted.

Corresponding parameter: [Fail-safe Mode] (→ **Mapping: port configuration** (→ p. 104)
The number of the required fail-safe values results from the size of the output data (→ **Configuration Area** (→ p. 103)).

Command request

34498

Register	Contents	
	Bits 8-15	**Bits 0-7**
500	Port No.	
501	reserved	
502	reserved	
503	0x30	User ID
504	Byte Length N	
505	Failsafe Mode	
506	reserved / Failsafe Data (Byte 1)	reserved / Failsafe Data (Byte 0)
...	...	...
521	reserved / Failsafe Data (Byte 31)	reserved / Failsafe data (Byte 30)

Legend:

▪ [Port No.]	Number of the IO-Link port	1 word	0x0001	Port X01
			0x0002	Port X02
			...	...
			0x0008	Port X08
▪ [User ID]	ID to identify the command	1 byte	0x00 .. 0xFF	
▪ [Data Length (Number of Bytes)]	Number of bytes that contain relevant user data (is only evaluated for Command = 0x02)	1 word	0x0002	2 bytes
			...	...
			0x0022	34 bytes
▪ [Failsafe Mode]	Fail-safe mode for the outputs of the IO-Link ports in case of an interruption of the Modbus TCP connection	1 word	0x0000	No Failsafe
			0x0001	Failsafe: Reset Value
			0x0002	Failsafe: Old Value
			0x0003	Failsafe: with Pattern
▪ [Failsafe Data (Byte n)]	Fail-safe values for the outputs (only with fail-safe mode = 0x0003)	1 byte	0x00 .. 0xFF	

Command response

34496

Register	Contents	
	Bits 8-15	Bits 0-7
0	Port No.	
1	reserved	
2	reserved	
3	0x30	User ID
4	Result	
5	Data Length (Number of Bytes)	
6	reserved / Error Code	Failsafe Mode / Additional Code
7 ... 21	reserved	

Legend:

▪ [Port No.]	Number of the IO-Link port	1 Word	0x0001	Port X01
			0x0002	Port X02
			...	...
			0x0008	Port X08
▪ [User ID]	reflected User ID from request channel	1 byte	0x00...0xFF	
▪ [Result]	Status of the command processing	1 Word	0x0000	OK
			0x00FF	Error
▪ [Data Length (Number of Bytes)]	Number of bytes that contain relevant user data	1 word	0x0001	1 byte
			0x0002	2 bytes
▪ [Failsafe Mode]	Fail-safe mode for the outputs of the IO-Link ports in case of an interruption of the Modbus TCP connection	1 byte	0x00	No Failsafe
			0x01	Failsafe: Reset Value
			0x02	Failsafe: Old Value
			0x03	Failsafe: with Pattern
▪ [Error Code]	Error ID	1 byte	→ **Error codes** (→ p. 117)	
▪ [Additional Code]	additional error codes	1 byte	→ **Additional Codes** (→ p. 117)	

Command 0x40 – Reboot

34457

The command reboots the AL1342.

Command request

34494

Register	Contents	
	Bits 8-15	Bits 0-7
500	reserved	
501	reserved	
502	reserved	
503	0x40	User ID
504	reserved	
505	0x00AA	
506 ... 521	reserved	

Legend:

- [User ID] ID to identify the command 1 byte 0x00 .. 0xFF

Command response

34508

Register	Contents	
	Bits 8-15	Bits 0-7
0	reserved	
1	reserved	
2	reserved	
3	0x40	User ID
4	Result	
5	Data Length (Number of Bytes)	
6	reserved / Error Code	0xAA / Additional Code
7 ... 21	reserved	

Legend:

- [User ID] reflected User ID from request channel 1 byte 0x00 .. 0xFF
- [Result] Status of the command processing 1 word 0x0000 OK
  0x00FF Error
- [Data Length (Number of Bytes)] Number of bytes that contain relevant user data 1 word 0x0001 1 byte
  0x0002 2 bytes
- [Error Code] Error ID 1 byte → **Error codes** (→ p. 117)
- [Additional Code] Additional error codes 1 byte → **Additional Codes** (→ p. 117)

Command 0x50 – Factory Reset

34499

The command resets all parameters to the factory settings (→ **Factory settings** (→ p. 94)).

Command request

34501

Register	Contents	
	Bits 8-15	Bits 0-7
500	reserved	
501	reserved	
502	reserved	
503	0x50	User ID
504	reserved	
505	0x0055	
506 ... 520	reserved	

Legend:

- [User ID] ID to identify the command 1 byte 0x00 .. 0xFF

Command response

34507

Register	Contents	
	Bits 8-15	Bits 0-7
0	reserved	
1	reserved	
2	reserved	
3	0x50	User ID
4	Result	
5	Data Length (Number of Bytes)	
6	reserved / Error Code	0x55 / Additional Code
7 ... 21	reserved	

Legend:

- [User ID] reflected User ID from request channel 1 byte 0x00 .. 0xFF
- [Result] Status of the command processing 1 word 0x0000 OK
  0x00FF Error
- [Data Length (Number of Bytes)] Number of bytes that contain relevant user data 1 word 0x0001 1 byte
  0x0002 2 bytes
- [Error Code] Error ID 1 byte → **Error codes** (→ p. 117)
- [Additional Code] additional error codes 1 byte → **Additional Codes** (→ p. 117)

14.3 ifm IoT Core

Content

Overview: IoT profile....128
Overview: IoT types....135
Overview: IoT services....136

33803

14.3.1 Overview: IoT profile

Content

Profile: blob 128
Profile: deviceinfo 129
Profile: devicetag 129
Profile: iolinkdevice_full 130
Profile: iolinkmaster 130
Profile: mqttCmdChannel 131
Profile: mqttCmdChannelSetup 131
Profile: mqttConnection 131
Profile: mqttSetup 132
Profile: network 132
Profile: parameter 133
Profile: processdata 133
Profile: runcontrol 133
Profile: service 133
Profile: software 133
Profile: software/uploadedablesoftware 134
Profile: Timer 134

34054

Profile: blob

52264

Element (identifier)	Properties	Mandatory	Comment
blobname	▪ type = data ▪ profiles = blob		labels element as device information
../size	type = data	mandatory	
../chunksize	type = data	mandatory	
../setblobdata	type = service	optional	
../getblobdata	type = service	optional	
../start_stream_set	type = service	optional	
../stream_set	type = service	optional	
../clear	type = service	optional	
../getcrc	type = service	optional	
../getmd5	type = service	optional	
../getdata	type = service	optional	
../setdata	type = service	optional	

Profile: deviceinfo

34207

Element (identifier)	Properties	mandatory	Comments
deviceinfo	▪ type = structure ▪ profile = deviceinfo		characterises the element as device information
../devicename	type = data	optional	
../devicefamiliy	type = data	optional	
../devicevariant	type = data	optional	
../devicesymbol	type = data	optional	
../deviceicon	type = data	optional	
../serialnumber	type = data	mandatory	
../productid	type = data	optional	
../productname	type = data	optional	
../productcode	type = data	mandatory	
../producttext	type = data	optional	
../ordernumber	type = data	optional	
../productiondate	type = data	optional	
../productioncode	type = data	optional	
../hwrevision	type = data	mandatory	
../swrevision	type = data	mandatory	
../bootloaderrevision	type = data	optional	
../vendor	type = data	optional	
../vendortext	type = data	optional	
../vendorurl	type = data	optional	
../vendorlogo	type = data	optional	
../productwebsite	type = data	optional	
../supportcontact	type = data	optional	
../icon	type = data	optional	
../image	type = data	optional	
../standards	type = data	optional	

Profile: devicetag

34206

Element (identifier)	Properties	mandatory	Comments
devicetag	▪ type = structure ▪ profile = devicetag		
../applicationtag	type = data	mandatory	
../applicationgroup	type = data	optional	
../machinecode	type = data	optional	
../tenant	type = data	optional	

Profile: iolinkdevice_full

52265

Element (identifier)	Characteristics	Mandatory	Comments
iolinkdevice	▪ type = structure ▪ profile = iolinkdevice_full		Structure of an IO-Link device
../vendorid	type = data	mandatory	
../deviceid	type = data	mandatory	
../productname	type = data	mandatory	
../serial	type = data	mandatory	
../applicationspecifictag	type = data	mandatory	
../pdin	type = data	mandatory	
../pdout	type = data	mandatory	
../status	type = data	mandatory	
../iolreadacyclic	type = data	mandatory	
../iolwriteacyclic	type = data	mandatory	
../iolinkevent	type = data	mandatory	

Profile: iolinkmaster

34205

Element (identifier)	Properties	Mandatory	Comments
masterport	▪ type = structure ▪ profile = iolinkmaster		Executable service
../mode	▪ type = data ▪ profile = parameter	mandatory	
../comspeed	▪ type = data ▪ profile = parameter	mandatory	
../mastercycletime_actual	▪ type = data ▪ profile = parameter	mandatory	
../mastercycletime_preset	▪ type = data ▪ profile = parameter	mandatory	
../validation_datastorage_mode	▪ type = data ▪ profile = parameter	mandatory	
../validation_vendorid	▪ type = data ▪ profile = parameter	mandatory	
../validation_deviceid	▪ type = data ▪ profile = parameter	mandatory	
../additionalpins_in	▪ type = data ▪ profile = processdata	optional	
../additionalpins_out	▪ type = data ▪ profile = processdata	optional	
../portevent	▪ type = data	mandatory	
../iolinkdevice	▪ type = structure ▪ profile = iolinkdevice_full	mandatory	

Profile: mqttCmdChannel

61186

Element (identifier)	Properties	Mandatory	Comment
mqttCmdChannel	▪ type = structure ▪ profile = commChannel		Profile of the MQTT command channel
../type	▪ type = data ▪ data type = STRING	mandatory	Protocol type of the interface
../status	▪ type = data ▪ data type = STRING	mandatory	Status of the MQTT command channel (possible values: init, running, stopped, error)
../mqttCmdChannelSetup	type = profile		Sub-profile: **Profile: mqttCmdChannelSetup** (→ p. 131)

Profile: mqttCmdChannelSetup

61187

Element (identifier)	Properties	Mandatory	Comment
mqttCmdChannelSetup	▪ type = structure ▪ profile = mqttCmdChannelSetup		Settings of the MQTT command channel
../brokerIP	▪ type = datat ▪ data type = STRING	optional	
../brokerPort	▪ type = data ▪ data type = STRING	optional	
../cmdTopic	▪ type = data ▪ data type = STRING	optional	
../defaultReplyTopic	▪ type = data ▪ data type = STRING	optional	

Profile: mqttConnection

61188

Element (identifier)	Properties	Mandatory	Comment
mqttConnection	▪ type = structure ▪ profile = commInterface		MQTT connection in the IoT Core
../type	▪ type = data ▪ data type = STRING	mandatory	Protocol type of the interface
../status	▪ type = data ▪ data type = STRING	mandatory	global status of the MQTT (possible values: init, running, stopped, error)
../mqttSetup	type = profile		Sub-profile: **Profile: mqttSetup** (→ p. 132)
../mqttCmdChannel	type = profile		Sub-profile: **Profile: mqttCmdChannel** (→ p. 131)

Profile: mqttSetup

61189

Element (identifier)	Properties	Mandatory	Comment
mqttSetup	▪ type = structure ▪ profile = mqttSetup		Settings of the MQTT command channel
../QoS	▪ type = data ▪ data type = Number	mandatory	Quality of Service of the MQTT connection
../version	▪ type = data ▪ data type = STRING	mandatory	

Profile: network

52266

Element (identifier)	Characteristics	Mandatory	Comments
network	▪ type = structure ▪ profiles = deviceinfo		Characterises the element as device information
../macaddress	▪ type = data ▪ profile = parameter	mandatory	
../ipaddress	▪ type = data ▪ profile = parameter	optional	
../ipv6address	▪ type = data ▪ profile = parameter	mandatory	
../subnetmask	▪ type = data ▪ profile = parameter	mandatory	
../ipdefaultgateway	▪ type = data ▪ profile = parameter	mandatory	
../dhcp	▪ type = data ▪ profile = parameter	optional	
../ipversion	▪ type = data ▪ profile = parameter	optional	
../hostname	▪ type = data ▪ profile = parameter	optional	
../autonegotiation	▪ type = data ▪ profile = parameter	optional	
../portspeed	▪ type = data ▪ profile = parameter	optional	
../enablenetwork	type = service	optional	
../disablenetwork	type = service	optional	

Profile: parameter

34215

The profile is used to mark the elements of type `data` as parameters (acyclic data). The profile defines no substructure.

Profile: processdata

34225

The profile is used to mark the elements of type `data` as process data (cyclic data). The profile does not define a substructure.

Profile: runcontrol

61190

Element (identifier)	Properties	Mandatory	Comment
runcontrol	▪ type = profile ▪ profile = runcontrol		Control of the MQTT command channel
../start	type = service	mandatory	**Service: start** (→ p. 147)
../stop	type = service	mandatory	**Service: stop** (→ p. 147)
../reset	type = service	mandatory	**Service: Reset** (→ p. 144)

Profile: service

34224

Element (identifier)	Properties	mandatory	Comments
service	▪ type = service ▪ profile = service		Executable service

Profile: software

34223

Element (identifier)	Properties	mandatory	Comments
software	▪ type = structure ▪ profile = software		characterises the element as software
../version	type = data	mandatory	
../type	type = data	mandatory	
../status	type = structure	optional	
../diag	type = structure	optional	

Profile: software/uploadedablesoftware

52267

Element (identifier)	Characteristics	Mandatory	Comments
software	▪ type = structure ▪ profiles = software/uploadablesoftware		Software that can be loaded to the device via the IoT Core
../lastinstall	type = data	optional	
../installhistory	type = data	optional	
../container	▪ type = data ▪ profile = blob	mandatory	
../preinstall	type = service	optional	
../install	type = service	mandatory	
../postinstall	type = service	optional	
../abortinstall	type = service	optional	
../installstatus	type = data	optional	

Profile: Timer

34226

Element (identifier)	Properties	Mandatory	Comment
timer	▪ type = structure ▪ profile = timer		
../counter	▪ type = data ▪ profile = parameter	mandatory	
../interval	▪ type = data ▪ profile = parameter	optional	
../start	type = service	optional	
../stop	type = service	optional	

14.3.2 Overview: IoT types

34055

The ifm IoT Core uses the following element types:

Name	Description
structure	Element is a structure element (like a folder in a file system)
service	Element is a service that can be addressed from the network
event	Element is an event that can be started by the firmware and sends messages.
data	Element is a data point
device	Root element a device represents

14.3.3 Overview: IoT services

Content

Service: factoryreset....136
Service: getblobdata....137
Service: getdata....137
Service: getdatamulti....138
Service: getelementinfo....138
Servicet: getidentity....139
Service: getsubscriberlist....140
Service: getsubscriptioninfo....141
Service: gettree....142
Service: install....143
Service: iolreadacyclic....143
Service: iolwriteacyclic....143
Service: querytree....144
Service: reboot....144
Service: Reset....144
Service: setblock....145
Service: setdata....146
Service: signal....146
Service: start....147
Service: start_stream_set....147
Service: stop....147
Service: stream_set....148
Service: subscribe....148
Service: unsubscribe....149
Service: validation_useconnecteddevice....149

34056

Service: factoryreset

34184

Name: factoryreset

Description: The service sets the parameters of the device to the factory settings.

Request data (field "data"): none

Response data (field "data"): none

Example:

```
{
"code": "request",
"cid": 4711,
"adr": "/firmware/factoryreset"
}
```

Service: getblobdata

52345

Name: getblobdata
Description: The service reads a binary large object (blob).
Applicable to: datastorage
Request data (field "data"):

Data field	Required field	Data type	Default	Description
pos	mandatory	number	0	Byte position
length	mandatory	number	-	Size of the object (number of bytes)

Return data (field "data"):

Data field	Required field	Data type	Default	Description
data	mandatory	STRING	0	Data to be decoded (BASE64 coded)
crc	optional	HEX STRING		CRC of the data after decoding
md5	optional	HEX STRING		MD5 checksum of the data after decoding

Service: getdata

34183

Name: getdata
Description: Service reads the value of a data point and provides it.
Request data (field "data"): none
Return data (field "data"):

Data field	Required field	Data type	Description
value	mandatory	STRING	Value of the element/data point

Example:

```
{
"code":"request",
"cid":4711,
"adr":"devicetag/applicationtag/getdata"
}
```

Service: getdatamulti

34174

Name: getdatamulti

Description: The service sequentially reads the values of several data points and provides them. The value and the diagnostic code are provided for each data point.

Request data (field "data"):

Data field	Required field	Data type	Description
datatosend	mandatory	ARRAY OF STRINGS	List of data points to be requested; data points must support the service getdata ("datatosend":["url1","url2",...,"urlx"])

Response data (field "data"): for each requested data point

Data field	Required field	Data type	Description
url	mandatory	STRING	Data point request
code	mandatory	INT	Diagnostic code of the request
data	mandatory	STRING	Value of the data point

Service: getelementinfo

52269

Name: getelementinfo

Description: The service reads the properties of an element of the IoT tree.

Applicable to: Objects of the type `device`

Request data (field "data"):

Data field	Required field	Data type	Default	Description
adr	mandatory	STRING		URL of the element, which properties to be changed

Return data (field "data"):

Data field	Required field	Data type	Default	Description
identifier	mandatory	STRING		Identifier of the element
type	mandatory	STRING		Type of the element
format	optional	JSON object	blank	Format of the data or the service content
uid	optional	STRING	blank	
profiles	optional	JSON array	blank	
hash	optional	STRING	--	

Servicet: getidentity

54690

Name: getidentity
Description: The service reads the device information of the AL1342 and issues it.
Request data ("data" field): none
Return data ("data" field):

Data field	Required field	Data type	Description	
iot		Device	Device description as JSON object	
iot.name	mandatory	STRING		
iot.uid	optional	STRING		
iot.version	mandatory	STRING		
iot.catalogue	optional	ARRAY OF OBJECTS		
iot.deviceclass	optional	ARRAY OF STRING		
iot.serverlist	optional	ARRAY OF OBJECTS		
device	optional		AL1342	
device.serialnumber	optional		Serial number	
device.hwrevision	optional		Hardware version	
device.swrevision	optional		Software version	
device.custom	optional			
Security	optional		Security options	
security.securitymode	optional	ENUM	shows if the security mode is activated	
security.authscheme	optional	ENUM	shows the active authentication scheme	
security.ispasswordset	optional	BOOL	shows whether a password has been set	
security.activeconnection	optional	ENUM	shows the currently used communication interface	
			▪ tcp_if	unencrypted http connection at the IoT interface, port 80
			▪ tls_if	encrypted https connection at the IoT interface, port 443
			▪ fb_if	unencrypted http connection at the fieldbus interface, port 80

Service: getsubscriberlist

61191

Name: getsubscriberlist
Description: The service provides a list of all active subscriptions.
Request data ("data" field): none
Return data ("data" field): Array with the following data

Data field	Mandatory field	Data type	Description
adr	mandatory	STRING	Data source
datatosend	mandatory	ARRAY OF STRINGS	List with URLs of the subscribed data points
cid	mandatory	NUMBER	ID of the subscription
callbackurl	mandatory	STRING	Address to which IoT Core event notifications are to be sent;
duration	mandatory	STRING	Storage duration of the value

Example:

- **Request object:**

```
{
"code":"request",
"cid":4711,
"adr":"/getsubscriberlist"
}
```

- **Return object:**

```
{
"cid":4711,
"data":[
{
"adr":"/timer[1]/counter/datachanged/subscribe",
"datatosend":["/iolinkmaster/port[2]/iolinkdevice/pdin"],
"cid":1,
"callbackurl":"http://192.168.0.45:80/temp",
"duration":"lifetime"},
{
"adr":"/timer[1]/counter/datachanged/subscribe",
"datatosend":["/processdatamaster/temperature","/processdatamaster/voltage"],
"cid":2,
"callbackurl":"http://192.168.0.44:80/temp",
"duration":"lifetime"}
]
"code":200
}
```

Service: getsubscriptioninfo

61192

Name: getsubscriptioninfo
Description: The service provides information about an existing subscription (subscribe).

The following parameters of the existing subscription are to be used for the query:

- Value of the identifier cid (e.g. 4711)
- Number of the timer (e.g. timer[1])
- Name of the callback topic (e.g. B. temp)

Request data ("data" field):

Data field	Mandatory field	Data type	Description
callback	mandatory	STRING	Address to which IoT Core event notifications are to be sent; complete URL: http://ipaddress:port/path

Return data ("data" field):

Data field	Mandatory field	Data type	Description
subscription	mandatory	BOOL	Status of the transferred subscription parameter
datatosend	mandatory	ARRAY OF STRINGS	List with subscribed data points
cid	mandatory	NUMBER	ID of the subscribe request
callbackurl	mandatory	STRING	Address to which IoT Core event notifications are to be sent; complete URL: http://ipaddress:port/path

Example:

- **Request object:**

```
{
"code":"request",
"cid":4711,
"adr":"/timer[1]/counter/datachanged/getsubscriptioninfo",
"data":{
"callback":"http://192.168.0.44:80/temp"}
}
```

- **Return object:**

```
{
"cid": 4711,
"data":{
"subscription": true,
"datatosend":[
"/iolinkmaster/port[2]/iolinkdevice/productname",
"/iolinkmaster/port[2]/iolinkdevice/pdin",
"/processdatamaster/temperature"],
"callbackurl":"http://192.168.0.44:80/temp",
"duration":"lifetime"},
"code":200
}
```

Service: gettree

61193

Name: gettree

Description: The service reads the device description of the IO-Link master and outputs it as a JSON object. The output can be limited to a subtree of the device description.

Request data ("data" field):

Data field	Mandatory field	Data type	Description
adr	optional	STRING	Root element of the subtree
level	optional	STRING	max. level up to which the subtree is output ▪ no entry: all levels will be displayed ▪ 0: do not display sub-elements ("subs") ▪ 1: display sub-elements ▪ 2: display sub-elements up to the 2nd level ▪ 3: display sub-elements up to the 3rd level ... ▪ 20: display sub-elements up to the 20th level

Return data ("data" field):

Data field	Mandatory field	Data type	Description
identifier	mandatory	STRING	Identifier of the root element
type	mandatory	STRING	Type of the element
format	optional	JSON Object	Format of the data content
uid	optional	STRING	
profiles	optional	JSON-Array	
subs	mandatory	JSON-Array	Sub-elements
hash	optional	STRING	

Examples:

- output the complete device description

```
{
"code":"request",
"cid":4,
"adr":"/gettree"
}
```

- output the subtree counter[2] of the device description up to the 2nd level

```
{
"code":"request",
"cid":4,
"adr":"/gettree"
"data":{
"adr":"counter[2]",
"level":2}
}
```

Service: install

52343

Name: install
Description: The service installs the firmware stored in the container area of the device.
Applicable to: container
Request data (data): none
Return data (data): none

Service: iolreadacyclic

34178

Name: iolreadacyclic
Description: The service acyclically reads the parameter value of an IO-Link device. The parameter is accessed via IO-Link index and subindex.
Request data (field "data"):

Data field	Required field	Data type	Description
index	mandatory	NUMBER	IO-Link index of the parameter
subindex	mandatory	NUMBER	IO-Link subindex of the parameter

Response data (field "data"):

Data field	Required field	Data type	Description
value	mandatory	STRING	Value of the parameter; Value in hexadecimal format

Service: iolwriteacyclic

34177

Name: iolwriteacyclic
Description: The service acyclically writes the parameter value of an IO-Link device. The parameter is accessed via IO-Link index and subindex.
Request data (field "data"):

Data field	Required field	Data type	Description
index	mandatory	NUMBER	IO-Link index of the parameter
subindex	mandatory	NUMBER	IO-Link subindex of the parameter
value	mandatory	STRING	New value of the parameter; Value in hexadecimal format

Response data (field "data"): none

Service: querytree

61194

Name: querytree

Description: The service searches a device tree for the criteria profile, type and name and outputs a list with the URLs of the elements found. At least one of the search criteria must be specified. The service can only be executed on the root node of the machine.

Return data ("data" field):

Data field	Mandatory field	Data type	Description
profile	optional	STRING	Profile of the searched element
type	optional	STRING	Type of the searched element
name	optional	STRING	Type of the searched element

Return ("data" field):

Data field	Mandatory field	Data type	Description
urlList	mandatory	Array	Array with URLs of the found elements; URLs are separated by commas

Service: reboot

34176

Name: reboot

Description: The service reboots the device.

Request data (field "data"): none

Return data (field "data"): none

Example:

```
{
"code":"request",
"cid":4,
"adr":"firmware/reboot"
}
```

Service: Reset

61195

Name: Reset

Description: The service resets a connection to the initialisation state.

Request data ("data" field): none

Return data ("data" field): none

Example:

```
{
"code":"request",
"cid":4711,
"adr":"/connections/mqttConnection/MQTTSetup/mqttCmdChannel/status/reset"
}
```

Service: setblock

34186

Name: setblock

Description: The service simultaneously sets the values of several data points of a structure.

Request data (field "data"):

Data field	Required field	Data type	Description
datatoset	mandatory	ARRAY OF OBJECTS	List of data points and their new values; data points must support the service setdata
consistent	optional	BOOL	

Response data (field "data"): none

Example:

Request:

```
{
"code":"request",
"cid":4711,
"adr":"iotsetup/network/setblock",
"data":{
"datatoset":{
"ipaddress":"192.168.0.6",
"subnetmask":"255.255.255.0",
"ipdefaultgateway":"192.168.0.250",
"dhcp":0}
}
}
```

Response:

```
{
"cid":4711,
"code":233
}
```

Service: setdata

34195

Name: setdata
Description: The service sets the value of the data point.
Request data ("data" field):

Data field	Mandatory field	Data type	Description
newvalue	mandatory	STRING	New value of the element/data point
duration	mandatory	STRING	Duration of value storage ▪ lifetime: Value is saved with IoT Core; Value remains valid even after restart of the device ▪ uptime: Value is saved until the next restart of the device

Return data ("data" field): none

Example:

```
{"code": "request",
"cid":4711,
"adr": "devicetag/applicationtag/setdata",
"data":{
"newvalue": "ifm IO-Link master
"duration": "lifetime"}
}
```

Service: signal

33819

Name: signal
Description: The service starts the flashing of the status LEDs of the AL1342.
Request data (field "data"): none
Return data (field "data"): none

Example:

```
{
"code":"request",
"cid":4711,
"adr":"firmware/signal"
}
```

Service: start

61196

Name: start
Description: The service starts a connection.
Request data ("data" field): none
Return data ("data" field): none

Example:

```
{
"code":"request",
"cid":4711,
"adr":"/connections/mqttConnection/MQTTSetup/mqttCmdChannel/status/start"
}
```

Service: start_stream_set

52342

Name: start_stream_set
Description: The service starts the sequential transfer of multiple data segments.
Applicable to: Objects of type data
Request data (data):

Data field	Required field	Data type	Default	Description
size	mandatory	STRING		Total size of data to be transfered (number of bytes)

Return data (data): none

Service: stop

61197

Name: stop
Description: The servicestops a connection.
Request data ("data" field): none
Return data ("data" field): none

Example:

```
{
"code":"request",
"cid":4711,
"adr":"/connections/mqttConnection/MQTTSetup/mqttCmdChannel/status/stop"
}
```

Service: stream_set

52341

Name: stream_set
Description: The service transfers a data segment.
Applicable to: Objects of type data
Request data (data):

Data field	Required field	Data type	Default	Description
value	mandatory	BIN (BASE64)	*	Segment of binary data (BASE64 coded)

Return data (data): none

Service: subscribe

61198

Name: subscribe
Description: The service subscribes to the values of data points. The data points to be subscribed are transferred as a list. The IoT Core sends changes to the data sink defined in callback.

CSV formatted notifications can only be transmitted using the TCP protocol via an activated and configured MQTT channel.

Request data ("data" field):

Data field	Mandatory field	Data type	Description
callback	mandatory	STRING	Address to which IoT Core event notifications are to be sent; URL format: ▪ JSON: http://ipaddress:port/path ▪ JSON: ws:///path ▪ JSON: mqtt://ipadress:port/topic ▪ CSV: tcp://ipaddress:port/path
datatosend	mandatory	ARRAY OF STRINGS	List from URLs of data elements; Elements must support getdata
codec	optional	STRING	Format of the returned data ▪ json: JSON formatted ▪ csv: CSV with standard separator (,) ▪ csv0: CSV formatted with comma separator (,) ▪ csv1: CSV formatted with semicolon separator (;)
DURATION	mandatory	STRING	Duration of value storage ▪ lifetime: Value is saved with IoT Core; Value remains valid even after restart of the device ▪ uptime: Value is saved until the next restart of the device ▪ once: send only one notification, user must unsubscribe immediately

Return data ("data" field): none
Notification: JSON

```
{
"code":"event",
"cid":4711,
"adr":"",
"data":{
```

```
"eventno":"EventNo",
"srcurl":"SrcURL",
"payload":{
"eventurl":{"code":EventStatus,"data":EventData},
"datapointurl_1":{"code":DataStatus_1,"data":DataValue_1},
"datapointurl_2":{"code":DataStatus_2,"data":DataValue_2},
...}}
}
```

Notification: CSV

```
SrcURL,EventNo,EventStatus,EventData,DataStatus_1,DataValue_1,DataStatus_2,DataVal
ue_2,...
```

- `SrcURL`: Source of the event (data point on which subscribe command was listed)
- `EventNo`: Event number
- `EventStatus`: Status code of the event
- `EventData`: Event data
- DataStatus_1: Status code of the 1st element in list datatosend
- `DataValue_1`: Value of the 1st element in list datatosend
- `DataStatus_2`: Status code of the 2nd element in list datatosend
- `DataValue_2`: Value of the 2nd element in list datatosend
- ...

Service: unsubscribe

34197

Name: unsubscribe

Description: The service deletes an existing subscription. The service unsubcribe is successful if cid and the callback address are registered for an active subscription (subscribe). If the STRING "DELETE" is provided in callback, the IO-Link master deletes all active subscriptions.

Request data (field "data"):

Data field	Required field	Data type	Description
callback	mandatory	STRING	Address to which IoT Core event notifications are to be sent; complete URL: http://ipaddress:port/path

Response data (field "data"): none

Service: validation_useconnecteddevice

52340

Name: validation_connecteddevice

Description: The service checks, whether Device ID and Vendor ID of the connected IO-Link device match with the values of the datapoints `../validation_vendorid` and `../validation_deviceid`.

Applicable to: Objects of type `stucture`

Request data (data): none

Return data (data): none

15 Index

A

Access the ifm IoT Core 40
Accessories 95
Acyclic Command Channel 115
Acyclic commands 118
Additional Codes 117
Appendix 96
Application 97
Approvals / tests 99

C

Change history 6
Cleaning process 93
Command 0x10 – Set mode 119
Command 0x20 – Set validation ID / data storage 121
Command 0x30 – Set fail-safe data pattern 123
Command 0x40 – Reboot 125
Command 0x50 – Factory Reset 126
Command request 119, 121, 123, 125, 126
Command response 120, 122, 124, 125, 126
Communication, parameter setting, evaluation 11
Configuration 23
Configuration Area 103
Configure IO-Link devices 37, 89
Configuring IO-Link the master 78
Configuring the MQTT command channel 70
Connecting IO-Link devices for Class A operation 16
Connecting IO-Link devices for Class B operation 17
Connecting the device 18
Connecting the IoT port 15
Connecting the Modbus TCP ports 15
Control IO-Link master 89
Creating a new notification 75

D

Deleting a notification 76
Diagnostic data 105
Digital inputs 12

E

Electrical connection 14, 100
Electrical data 97
Error codes 117
Ethernet ports 20
Example
  Activate security mode 49
  Browsing device description 45
  Change name of the IO-Link master 63
  Change the parameter value of an IO-Link device 59
  Changing a subscription 65
  Checking subscriptions 67
  Clone the Data Storage of an IO-Link port 53
  Configuring the MQTT command channel 71
  GET request 40
  Integrate IO-Link master in a CODESYS project 81
  output subtree 44
  POST request 41
  Publish the temperature to an MQTT broker 73
  Read IO-Link process data (operating mode 55
  Read several parameter values of the IO-Link master simultaneously 45
  Read the parameter value of an IO-Link device 58
  Reading digital input (operating mode 56
  Reading properties of an element 43
  Request with authentication 50
  reset password 50
  Subscribing notifications via WebSocket 68
  Subscribing to notifications 46, 64
  Subscribing to notifications in CSV format 66
  Unsubscribing from notifications 66
  Update firmware 61
  Writing digital output (operating mode 56
  Writing IO-Link value (operating mode 55
Explanation of Symbols 5

F

Factory settings 94
Fieldbus
  Confguring the fieldbus interface 51
  Configure IP settings 30
  Configuring the IP settings 50
  set the length of the process data 31
  Setting fail-safe values 52
Firmware
  Reboot the device 36
  Reset device to factory settings 36
First steps 43
Function 10

G

Gateway
  Reading device information 60
  Reading status and diagnostic information 60
  Resetting, rebooting and localising the device 60
  Setting the application tag 63
  Updating the firmware 61
General 7
General functions 43
GET request 40

I

ifm IoT Core 38, 127
Info
  Show device information 35
Input Data 108
Inputs 98
Inputs / outputs 97
Integrate the AL1342 into the Modbus project 81
Intended use 9
Interfaces 98
Internet of Things (IoT) 11
IO-Link 11
IO-Link devices
  Accessing parameters 58
  Indicating IO-Link events 60
  Reading an writing device information 59
IO-Link ports 16
  Activate data transfer to LR AGENT or LR SMARTOBSERVER 32
  Configure operating mode 33
  Configuring data transfer to LR AGENT or LR SMARTOBSERVER 55
  Configuring device validation and data storage 53
  Indicating port events 58
  Reading / writing process data 55

set fail-safe values ... 35
Set the device validation and data storage ... 34
Setting the operating mode of pin 4 (US) ... 52
IO-Link Ports (Class A) ... 21
IO-Link supply ... 12
IoT
Configure IP settings ... 26
Configure security mode ... 27
Configure the interface to LR AGENT or LR SMARTOBSERVER ... 29
Configuring access rights ... 28, 47
Configuring IP settings ... 47
Configuring security mode ... 48
Configuring the LR AGENT or LR SMARTOBSERVER interface ... 48
IoT Core
Diagnostic codes ... 42
General information ... 39
IoT port ... 21
IT security ... 8

L

LED indicators ... 20
Legal and copyright information ... 5
LR DEVICE ... 24

M

Maintenance, repair and disposal ... 93
Managing notifications ... 75
Mapping
Compact Input Block ... 110
Compact Output Block ... 112
diagnostic information ... 109
Diagnostics ... 106
digital input data ... 109
Digital output data ... 112
events ... 107
port configuration ... 104
PQI ... 114
Status information ... 114
Status information IO-Link ports ... 110
Mechanical data ... 99
Modbus TCP ... 11, 81, 101
Programmers' notes ... 90
Mount the device ... 13
Mounting ... 13
MQTT support ... 70

N

Note
Connection states ... 71
Exception Codes ... 90
Security mode ... 49
Notes ... 14

O

Offline parameter setting ... 25
Operating and display elements ... 19
Operating conditions ... 99
Operation ... 92
Output Data ... 111
Outputs ... 98
Overview ... 19
IoT profile ... 128
IoT services ... 136
IoT types ... 135

P

Parameter setting ... 11
Parameter setting with LR DEVICE ... 25
POST request ... 41
Preliminary note ... 5
Profile
blob ... 128
deviceinfo ... 129
devicetag ... 129
iolinkdevice_full ... 130
iolinkmaster ... 130
mqttCmdChannel ... 131
mqttCmdChannelSetup ... 131
mqttConnection ... 131
mqttSetup ... 132
network ... 132
parameter ... 133
processdata ... 133
runcontrol ... 133
service ... 133
software ... 133
software/uploadedablesoftware ... 134
Timer ... 134
Programmers' notes ... 39
Purpose of the document ... 5

R

Read device information ... 89
Read diagnostic information and events ... 88
Read input data of individual IO-Link ports ... 85
Read input data of several IO-Link ports ... 84
Reading and writing process data ... 79
Register ... 102
Remarks ... 25
Replacing IO-Link device ... 93
Request channel ... 115
Required background knowledge ... 7
Response channel ... 116
Rules for accessing the Modbus registers ... 90

S

Safety instructions ... 7
Safety symbols on the device ... 7
Searching for elements in the device tree ... 77
Security mode ... 11
Service
factoryreset ... 136
getblobdata ... 137
getdata ... 137
getdatamulti ... 138
getelementinfo ... 138
getsubscriberlist ... 140
getsubscriptioninfo ... 141
gettree ... 142
install ... 143
iolreadacyclic ... 143
iolwriteacyclic ... 143
querytree ... 144
reboot ... 144
Reset ... 144
setblock ... 145
setdata ... 146

signal ... 146
start ... 147
start_stream_set ... 147
stop ... 147
stream_set ... 148
subscribe ... 148
unsubscribe ... 149
validation_useconnecteddevice ... 149
Servicet
getidentity ... 139
Set IO-Link master ... 82
Set IO-Link ports ... 83
Setting the storage duration ... 46
Set-up ... 22
Single Port Access ... 113
Status LEDs ... 20
Subscribing to notifications ... 64
Supported function codes ... 90

T

Technical data ... 97

U

Updating the firmware ... 80, 93
Use acyclic services ... 91
Using the IoT-Core Visualizer ... 74
Using Web Socket ... 68
Using web-based management ... 92

V

Visual indication ... 12
Voltage supply ... 21

W

Write output data of individual IO-Link ports ... 87
Write output data of several IO-Link ports ... 86