IO-Link Master ICE2 and ICE3

MQTT

Technical Update

Your automation, our passion.

PEPPERL+FUCHS

With regard to the supply of products, the current issue of the following document is applicable: The General Terms of Delivery for Products and Services of the Electrical Industry, published by the Central Association of the Electrical Industry (Zentralverband Elektrotechnik und Elektroindustrie (ZVEI) e.V.) in its most recent version as well as the supplementary clause: "Expanded reservation of proprietorship"

Table of Contents

1. MQTT

1.1 Document Overview

This document is a supplement to the ICE2 and ICE3 Manuals to discuss the MQTT Settings Configuration page, which is available starting with application base 1.5.42 for either PROFINET IO or EtherNet/IP.

The following topics are discussed.

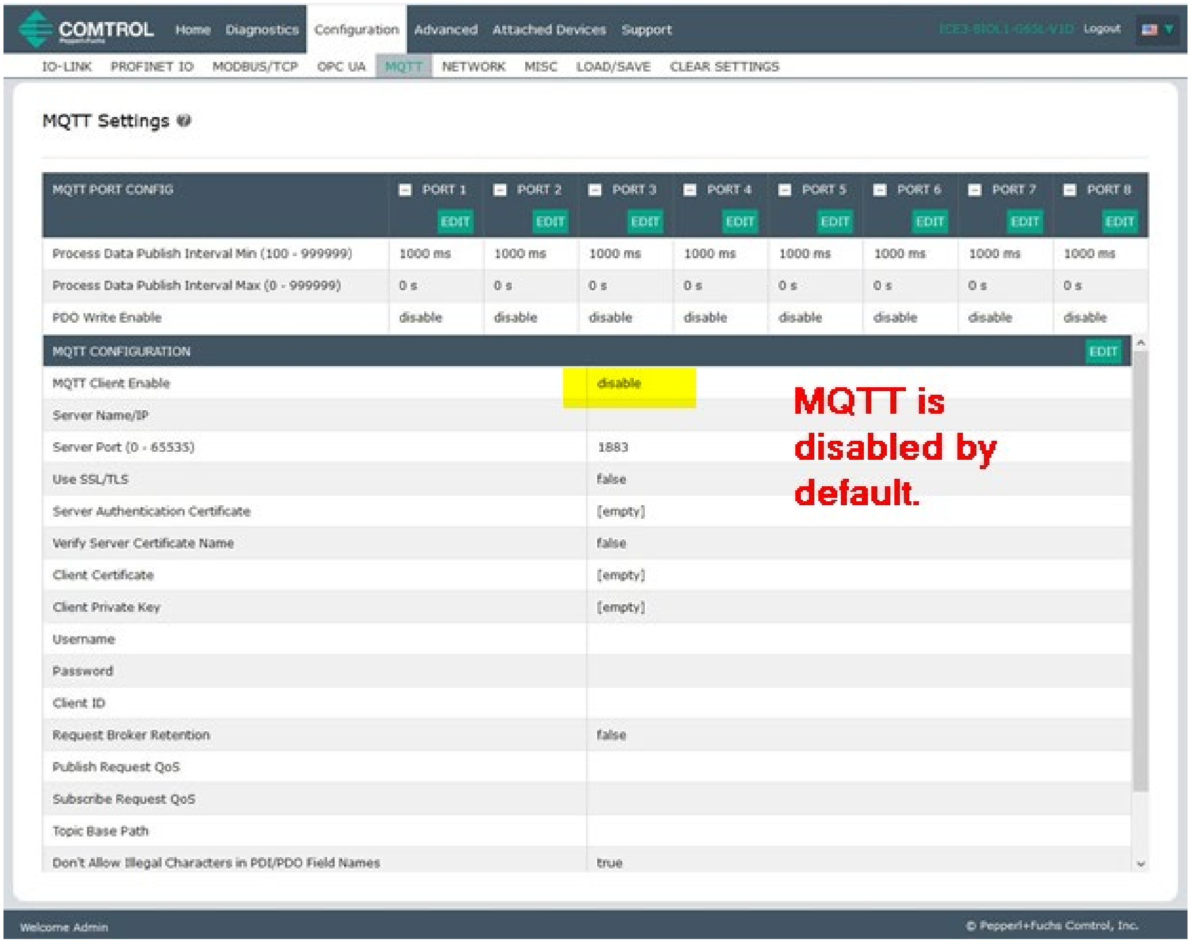
- Process Data Publish Interval
- Topics and Payloads
- ISDU Read/Write
- ISDU Request Payload
- ISDU Response Payload
- PDO Write
- MQTT Settings Parameters
- Configuring MQTT

1.2 MQTT Settings Configuration Page

MQTT support provides a way to publish various data to an MQTT broker. MQTT is a simple publish-subscribe messaging protocol that is becoming popular for use in Internet of Things (IoT) type applications:

- http://mqtt.org/
- https://en.wikipedia.org/wiki/MQTT/

The MQTT standard does not define any format for the published messages, but JSON has been almost universally adopted by the MQTT implementation in the IoT area, so JSON is the format chosen for use by the IO-Link Master MQTT implementation.



MQTT Settings

MQTT PORT CONFIG	PORT 1 EDIT	PORT 2 EDIT	PORT 3 EDIT	PORT 4 EDIT	PORT 5 EDIT	PORT 6 EDIT	PORT 7 EDIT	PORT 8 EDIT
Process Data Publish Interval Min (100 - 999999)	1000 ms	1000 ms	1000 ms	1000 ms	1000 ms	1000 ms	1000 ms	1000 ms
Process Data Publish Interval Max (0 - 999999)	0 s	0 s	0 s	0 s	0 s	0 s	0 s	0 s
PDO Write Enable	disable	disable	disable	disable	disable	disable	disable	disable

MQTT CONFIGURATION	EDIT
MQTT Client Enable	disable
Server Name/IP	
Server Port (0 - 65535)	1883
Use SSL/TLS	false
Server Authentication Certificate	[empty]
Verify Server Certificate Name	false
Client Certificate	[empty]
Client Private Key	[empty]
Username	
Password	
Client ID	
Request Broker Retention	false
Publish Request QoS	
Subscribe Request QoS	
Topic Base Path	
Don't Allow Illegal Characters in PDI/PDO Field Names	true

Figure 1

1.3 Process Data Publish Interval

The PD Interval Min/Max configuration values control how often a PDI/PDO message is published. With the default configuration (min=1000ms max=0s), a value will be published when it has changed and at least 1 second has elapsed since the previous message was published. With max=0, a value will never be published unless it changes (except for once on startup).

If Interval Max is configured to a non-zero value, then a PD messages will always be published after the configured interval even when the data is unchanged.

For example, with a configuration of min=5000ms max=60s, a changing value will only be published once every 5 seconds regardless of how often it changes, and a non-changing value will be published once every 60 seconds even when it has not changed.

1.4 Topics and Payloads

All published payloads are JSON objects. The initial set of published paths and data are listed below.

1.4.1 MqttTopicBase/clientinfo

A summary of information about the IO-Link Master is published once each time the MQTT client starts. Example payload:

```
{
"hostname" : "grant-ice2",
"manufacturer" : "Pepperl-Fuchs Comtrol, Inc.",
"model" : "ICE2-8IOL-K45P-DIN",
"serial" : "9710-000064",
"version" : "EtherNet/IP 1.5.0.201",
"numdioports" : 0,
"numiolinkports" : 8
}
```

1.4.2 MqttTopicBase/clientstatus

A periodic message containing IO-Link Master status is published under the "clientstatus" topic at a user- configured interval. The publishing period is user-configurable and can be disabled completely. An example payload is shown below.

```
{
"uptime": 3464,
"ports": [
{
"port": 1,
"mode": "SIOInput",
"status": "Operational",
"state": "None",
"pd_retries": 0,
"pd_errors": 0,
"pdi_valid": true,
"pdo_valid": false
},
{
"port": 2,
"mode": "IOLinkInput",
"status": "Operational",
"state": "Operate",
"pd_retries": 1,
"pd_errors": 0,
"pdi_valid": false,
"pdo_valid": false
},
{
"port": 3,
"mode": "Reset",
"status": "Inactive",
"state": "None",
"pd_retries": 0,
"pd_errors": 0,
"pdi_valid": true,
"pdo_valid": false
},
...
{
"port": 8,
"mode": "IOLinkInput",
"status": "Inactive",
"state": "Init",
"pd_retries": 0,
"pd_errors": 0,
"pdi_valid": true,
"pdo_valid": false
}
]
}
```

1.4.3 MqttTopicBase/port/n/status

A port status object is published on startup and each time the port changes state. Example payloads:

```
{
"port":2,
"mode":"IOLinkInput",
"status":"Operational, PDI Valid",
 "state":"Operate"
}

{
"port":5,
"mode":"IOLinkInput",
"status":"Inactive",
"state":"Init"
}

{
"port":3,
"mode":"Reset",
"status":"Inactive",
"state":"None"
}

{
"port":1,
"mode":"SIOOutput",
"status":"Operational",
"state":"Reset"
}
```

1.4.4 MqttTopicBase/port/n/deviceinfo

A port deviceinfo object is published each time the communication to an IO-Link device is established successfully. Example payloads:

```
{
"port": 4,
"vendorid": 1,
"deviceid": 1120516,
"functionid": 0,
"vendname": "Pepperl+Fuchs",
"vendtext": "www.pepperl-fuchs.com/io-link",
"prodname": "OMT300-R200-2EP-IO-V1",
"prodid": "295670-100140",
"prodtext": "Distance sensor",
"serial": "40000077249691",
"hwvers": "HW01.00",
"fwvers": "FW01.02",
"apptag": "Your automation, our passion.",
"functag": "R200 series",
"loctag": "***",
"pdibytes": 4,
"pdobytes": 1,
"isducapable": true,
"dscapable": true,
"dslength": 213,
"iolinkvers": "11"
}
```

1.4.5 MqttTopicBase/port/n/event

An event data object is published each time an event occurs for any port. Example payload:

```
{
"port" : 1,
"instance" : 3,
"mode" : 1,
"type" : 1,
"pdvalid" : 0,
"local" : 1,
"code" : 36,
"description" : "inst=AL mode=SINGLE type=MESSAGE
pd=INVALID local=ff code=0x0024:M_PREOPERATE"
}
```

1.4.6 MqttTopicBase/port/n/pdi

PDI values are published when they change. Raw byte array data is always present. If PDI length is 4 or less, an unsigned integer version is also present. If an IODD file is present, dissected field values will be present for the configured process data group as defined by the IODD. If enabled, the process data field names will be "sanitized" so that they are legal JavaScript identifiers by replacing illegal characters with underscores.

Example payloads:

```
{
"port" : 2,
"valid" : 1,
"uint" : 366,
"raw" : [1,137]
}


{
"port" : 4,
"valid" : 1,
"raw" : [9,124,9,79,9,127,0,0]
}
```

```
{
"port": 1,
"valid": 1,
"V_PdT": {
   "Temperature": 0,
   "Switch status [OUT1].": 0
       },
"raw": [9,140,9,136,9,90,0,0]
}


{
"port": 1,
"valid": 1,
"V_PdT": {
   "Temperature": 0,
    "Switch_status_OUT1_": 0
},
"raw": [9,140,9,136,9,90,0,0]
}
```

1.4.7 MqttTopicBase/port/n/pdo

PDO values are published when they change (subject to the minimum publish interval setting) or periodically according to the maximum publish interval setting. Raw byte array data is always present. If PDO length is 4 or less, an unsigned integer version is also present. If an IODD file is present, dissected field values will be present for the configured process data group as defined by the IODD. If enabled, the process data field names will be "sanitized" so that they are legal JavaScript identifiers by replacing illegal characters with underscores.

```
{
"port" : 1,
"valid" : 1,
"uint" : 252,
"raw" : [252]
}
```

1.4.8 MqttTopicBase/port/n/auxin

IO-Link port DI (auxiliary input) pin values will be published when they change (subject to minimum publish interval setting) or periodically according to the maximum publish interval setting. The payload comprises the port number and a single "value" field having an integer value of 0 or 1.

```
{
"port": 2,
"value": 0
}
```

1.4.9 ISDU Read/Write

Since MQTT lacks intrinsic support for request/response semantics, ISDU read/write requests and responses are handled via a pair of topics:

- MqttTopicBase/port/n/isdu/request/client_transaction_id
- MqttTopicBase/port/n/isdu/response/client_transaction_id

Requests for ISDU read/write are published by other clients to the "request" topic shown above. The *client_transaction_id* is an arbitrary string chosen by the requesting client and should be chosen to be unique. After the ISDU operation is completed, the IO-Link Master will publish the response to the corresponding "response" topic (with the same *client_transaction_id* as the request).

1.5 ISDU Request Payload

The ISDU request payload is a JSON object with the fields described below.

Name	Type	Description
op	string	Required — must be "read" or "write"
index	integer	Required
subindex	integer	Optional (defaults to 0 if not provided)
Fields specific to read requests:		
format	string	Optional — if present, it determines the format of the returned read data in the response. Should be one of "str" "raw" "uint". If not provided, read data will be returned in all formats.
Fields specific to write requests (exactly one of uint, raw, or str must be present)		
raw	array	Array of integer byte values (decimal)
uint	integer	Integer data value (requires *len* field)
str	string	UTF-8 data string (*len* field is optional)
len	integer	Required for *uint* data, optional for *str* data. Controls number of data bytes written.

Table 1

In a write request with str data and a len field, the string will be NULL-padded to the requested length before being written to the device.

1.6 ISDU Response Payload

The ISDU response payload is a JSON object with the fields described below:

Name	Type	Description
op	string	"op" value from request
index	integer	"index" value from request
subindex	integer	"subindex" value from request (if present and non-zero)
status	string	"OK" if the request was successful, otherwise an error message.
Fields specific to read response (one or more of raw, str, uint may be present:		
raw	array	Array of integer byte values (decimal)
uint	integer	Unsigned integer value
str	string	UTF-8 string data
len	integer	Number of bytes read

Table 2

If no format is specified in the read request, then the read response will contain data in all three formats when *len* ≤ 4. If *len* > 4, only raw and str formats will be returned. If the read operation fails, no len value or data values will be returned.

In a read response, the *str* value will have any trailing NULL bytes removed. The *len* field will always indicate the total number of bytes read (including any trailing NULL bytes for string values)

Below is an example of a write-string request/response followed by a read-string request/response and a read- raw request/response where the topic base path is IOLM:

```
IOLM/port/1/isdu/request/66b127b7-f39d-40e7-b786-
1cffc8d344a0

{

"op": "write",

"index": 24,

"str": "hi there"

}


IOLM/port/1/isdu/response/66b127b7-f39d-40e7-b786-
1cffc8d344a0

{
```

PEPPERL+FUCHS

```
"op": "write",

"index": 24,

"status": "OK"

}


IOLM/port/1/isdu/request/2ee5141e-335b-4e33-bf4e-
dedf01a0ff7b

{

"op": "read",

"index": 24,

"format": "str"

}


IOLM/port/1/isdu/response/2ee5141e-335b-4e33-bf4e-
dedf01a0ff7b

{

"op": "read",

"index": 24,

"str": "hi there"

"len": 16,

"status": "OK"

}


IOLM/port/1/isdu/request/1c510d4d-151e-49b3-bbad-
0847a272812e

{

"op": "read",

 "index": 24,

"format": "raw"

}


IOLM/port/1/isdu/response/1c510d4d-151e-49b3-bbad-
0847a272812e

{

"op": "read",
"index": 24,
"raw":[104,105,32,116,104,101,114,101,0,0,0,0,0,0,0,0],
"len": 16,
"status": "OK"
}

IOLM/port/3/isdu/request/1234
{
"op": "read",
"index": 15,
"format": "str"
}

IOLM/port/3/isdu/response/1234
Example: negative response ‘Index not available’
{
"op": "read",
"index": 15,
"status": "Error",
"errormsg": "index invalid",
"errortype": {
"status": 1,
"code": 128,
"addcode": 17
}
}
```

1.7 PDO Write

PDO values may be written by publishing to *MqttTopicBase/port/n*/pdo/wr. The payload may contain PDO data fields in one of two formats: raw or uint. Data in raw format must match the PDO length exactly. Data in uint format is supported only for PDO lengths of 4 bytes or less. If an IODD file is present that defines Process Data groups and fields, then those may be used to write to individual field values. The available groups/fields are shown on the MQTT diagnostics page as "PDGroups" in the MQTT Port Status table. The published object may also contain a boolean valid flag.

Example payloads:

```
IOLM/port/5/pdo/wr

{

"uint": 349718

}


IOLM/port/1/pdo/wr

{

"raw": [1,254,75]

}


IOLM/port/1/pdo/wr

{

"uint": 15,

"valid": true

}


IOLM/port/5/pdo/wr

{

"raw": [1, 243,79,103, 253,12],

"valid": true

}


IOLM/port/3/wr

{
```

```
"PDOut": {
"LevelSetpoint": 119.34
},
"valid": true
}

IOLM/port/1/pdo/wr
{
"valid": true
}

IOLM/port/8/pdo/wr
{
"valid": false
}
```

1.8 MQTT Settings Parameters

The following table illustrates the *MQTT Port Configuration settings*.

Field	Type	Default	Description
Process Data Publish Interval min (100 - 999999)	int	1000	Minimum interval (milliseconds) between successive PDI (or PDO) messages for the port.
Process Data Publish Interval max(100 - 999999)	int	0	Maximum interval (seconds) between successive PDI (or PDO) messages for a port (0 == infinite).
PDO Write Enable	enum	disable	Enable PDO write

Table 3

The following table provides information about *MQTT Configuration* for client data that are global MQTT configuration settings.

Field	Type	Default	Description
MQTT Client Enable	enum	disable	Enable/disable the MQTT client
Server Name/IP	string		MQTT server hostname or IPv4 address
Server Port (0 - 65535)	int	1883	MQTT server port (0-65535). Probably 8883 if TLS is enabled.
Use SSL/TLS	boolean	false	Use SSL/TLS encryption
Server Authentication Certificate	file		The Server Authentication Certificate is used by the IO-Link Master to verify the server's identity. TheX509 certificate used to verify server identity (PEM encoding). The **Use SSL/TSL** option must be enabled for this option to work.
Verify Server Certificate Name	boolean	false	Enable verification of information (e.g. name) in server auth certificate.
Client Certificate	file		The Client Authentication Certificate is sent by the IO-Link Master (the client) to the server to verify the client's identity. The X509 certificate sent to server for authentication (PEM encoding). The **Use SSL/TSL** option must be enabled for this option to work.

Field	Type	Default	Description
Client Private Key	file		The Client Private Key is required to use the Client Authentication Certificate as described above. This is the private key for above certificate (PEM encoding). The **Use SSL/TSL** option must be enabled for this option to work.
Username	string		Username sent to server for authentication
Password	string		Password sent to server for authentication
Client ID	string		Client ID sent to server when connecting
Request Broker Retention	boolean	false	Request that broker retain published messages
Publish Request QoS	enum	at most once	QoS level 0 requested when publishing
Subscribe Request QoS	enum	at most once	QoS level 0 requested when subscribing
Topic Base Path	string		Path prefix used for all publish messages
Don't Allow Illegal Characters in PDI/ PDO Field Names	boolean	true	Convert PDI/PDO field names to valid JavaScript identifiers by replacing illegal characters with underscores
Client Status Publish Interval (0 - 999999)	int	10	Publishing interval (seconds) for the client status message (0 == disable)
ISDU Write Enable	enum	disable	Enable ISDU write.

1.9 Configuring MQTT

Configuring MQTT settings

Use this procedure to configure MQTT settings.

Note

By default, MQTT is disabled.

1. Refer to the MQTT Settings table if you require definitions or values for the options.
2. If necessary, click **Configuration | MQTT**.

3. To configure port-level values, click the **EDIT** button below the port that you want to update.
4. Click the **SAVE** button after updating the settings.
5. Click the **MQTT Configuration EDIT** button.
6. Select **enable** from the **MQTT Client Enable** option drop box.
7. Enable other options that your environment requires.
8. Scroll to the top of the page and click the **SAVE** button.

Your automation, our passion.

Explosion Protection

- Intrinsic Safety Barriers
- Signal Conditioners
- FieldConnex® Fieldbus
- Remote I/O Systems
- Electrical Ex Equipment
- Purge and Pressurization
- Industrial HMI
- Mobile Computing and Communications
- HART Interface Solutions
- Surge Protection
- Wireless Solutions
- Level Measurement

Industrial Sensors

- Proximity Sensors
- Photoelectric Sensors
- Industrial Vision
- Ultrasonic Sensors
- Rotary Encoders
- Positioning Systems
- Inclination and Acceleration Sensors
- Fieldbus Modules
- AS-Interface
- Identification Systems
- Displays and Signal Processing
- Connectivity

Pepperl+Fuchs Quality
Download our latest policy here:

www.pepperl-fuchs.com/quality

PEPPERL+FUCHS

www.pepperl-fuchs.com
Subject to modifications · © Pepperl+Fuchs
Printed in Germany · 12/2020 · DOCT-7094