ibaPDA-Data-Store-MQTT
Data streaming into MQTT broker

Manual
Issue 1.3
Manufacturer
iba AG
Koenigswarterstr. 44
90762 Fuerth
Germany

Contacts
Main office +49 911 97282-0
Fax +49 911 97282-33
Support +49 911 97282-14
Engineering +49 911 97282-13
E-mail iba@iba-ag.com
Web www.iba-ag.com

Unless explicitly stated to the contrary, it is not permitted to pass on or copy this document, nor to make use of its contents or disclose its contents. Infringements are liable for compensation.

© iba AG 2020, All rights reserved.

The content of this publication has been checked for compliance with the described hardware and software. Nevertheless, discrepancies cannot be ruled out, and we do not provide guarantee for complete conformity. However, the information furnished in this publication is updated regularly. Required corrections are contained in the following regulations or can be downloaded on the Internet.

The current version is available for download on our web site www.iba-ag.com.

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Revision - Chapter / Page</th>
<th>Author</th>
<th>Version SW</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.3</td>
<td>09-2020</td>
<td>Time behavior</td>
<td>st</td>
<td>7.2.0</td>
</tr>
</tbody>
</table>

Windows® is a brand and registered trademark of Microsoft Corporation. Other product and company names mentioned in this manual can be labels or registered trademarks of the corresponding owners.
Content

1  About this manual.............................................................................................................4
   1.1  Target group and previous knowledge .................................................................4
   1.2  Notations ...............................................................................................................5
   1.3  Used symbols.........................................................................................................6

2  Introduction ..................................................................................................................7
   2.1  System requirements ............................................................................................8

3  Data store configuration.............................................................................................9
   3.1  Add a data store ..................................................................................................9
   3.2  MQTT data store ................................................................................................10
   3.2.1  Time behavior................................................................................................13
   3.2.2  File buffer behavior .......................................................................................13
   3.3  Configuring topics ..............................................................................................14

4  Signal selection ..........................................................................................................17

5  Trigger mode ...............................................................................................................19

6  Diagnostics ................................................................................................................23
   6.1  Data storage status ..............................................................................................23
   6.2  Diagnostics of data stores ..................................................................................24

7  Appendix .....................................................................................................................25
   7.1  Description of the data formats ..........................................................................25
   7.1.1  Data format JSON (grouped) .........................................................................25
   7.1.2  Data format JSON (per signal) .......................................................................27

8  Support and contact ....................................................................................................30
1 About this manual

This documentation describes the function and application of the data store *ibaPDA-Data-Store-MQTT*.

This documentation is a supplement to the *ibaPDA* manual. Information about all the other characteristics and functions of *ibaPDA* may be found in the *ibaPDA* manual or in the online help.

You will find basic information about data storage in *ibaPDA* in the *ibaPDA* manual part 5.

1.1 Target group and previous knowledge

This documentation addresses qualified professionals, who are familiar with handling electrical and electronic modules as well as communication and measurement technology. A person is regarded as professional if he/she is capable of assessing the work assigned to him/her and recognizing possible risks on the basis of his/her specialist training, knowledge and experience and knowledge of the standard regulations.

This documentation in particular addresses persons, who are concerned with the configuration, test, commissioning or maintenance of the supported database, cloud or cluster storage technology. For the handling of *ibaPDA-Data-Store-MQTT* the following basic knowledge is required and/or useful:

- Windows operating system
- Basic knowledge of *ibaPDA*
- Basic knowledge of databases, cloud or cluster storage technology
### 1.2 Notations

In this manual, the following notations are used:

<table>
<thead>
<tr>
<th>Action</th>
<th>Notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Menu command</td>
<td>Menu <em>Logic diagram</em></td>
</tr>
<tr>
<td>Calling the menu command</td>
<td><em>Step 1 – Step 2 – Step 3 – Step x</em></td>
</tr>
<tr>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>Select the menu <em>Logic diagram - Add - New function block</em>.</td>
</tr>
<tr>
<td>Keys</td>
<td><em>&lt;Key name&gt;</em></td>
</tr>
<tr>
<td></td>
<td>Example: &lt;Alt&gt;; &lt;F1&gt;</td>
</tr>
<tr>
<td>Press the keys simultaneously</td>
<td><em>&lt;Key name&gt; + &lt;Key name&gt;</em></td>
</tr>
<tr>
<td></td>
<td>Example: &lt;Alt&gt; + &lt;Ctrl&gt;</td>
</tr>
<tr>
<td>Buttons</td>
<td><em>&lt;Key name&gt;</em></td>
</tr>
<tr>
<td></td>
<td>Example: &lt;OK&gt;; &lt;Cancel&gt;</td>
</tr>
<tr>
<td>File names, paths</td>
<td>&quot;Filename&quot;, &quot;Path&quot;</td>
</tr>
<tr>
<td></td>
<td>Example: &quot;Test.doc&quot;</td>
</tr>
</tbody>
</table>
1.3 Used symbols

If safety instructions or other notes are used in this manual, they mean:

---

Danger!

⚠️ The non-observance of this safety information may result in an imminent risk of death or severe injury:

- Observe the specified measures.

---

Warning!

⚠️ The non-observance of this safety information may result in a potential risk of death or severe injury!

- Observe the specified measures.

---

Caution!

⚠️ The non-observance of this safety information may result in a potential risk of injury or material damage!

- Observe the specified measures.

---

Note

ℹ️ A note specifies special requirements or actions to be observed.

---

Tip

💡 Tip or example as a helpful note or insider tip to make the work a little bit easier.

---

Other documentation

📚 Reference to additional documentation or further reading.
2 Introduction

Different types of data stores are available in ibaPDA for different purposes and methods of data storage. Depending on the licenses registered in the dongle, different types of data stores are available for configuration in the dialog.

This documentation describes the “MQTT time-based data store” type of recording.

This type of data store sends time-based signal data to an MQTT broker. The broker publishes these values to clients who have subscribed to these signals.

MQTT is based on an event-driven publish/subscribe architecture. The core is a central server (broker) to which the transmitter as well as the receiver connect. The data is sent (published) and received (subscribed) via so-called topics. Topics are basically communication channels in which the transmitters, e.g. sensors, write their data. The broker checks which receivers (clients) have opened a channel for this data and passes on the data to the clients.

This means for ibaPDA that ibaPDA acts as MQTT client. ibaPDA sends (publishes) topics with measured values to the MQTT broker, which distributes these further.

![MQTT architecture](image)

Chapter 📖 Signal selection, page 17 describes the selection of the signals that are to be recorded.

The data can be continuously recorded or recorded by trigger. See chapter 📖 Trigger mode, page 19

---

**Note**

The MQTT broker is not part of ibaPDA. This must be purchased, installed and configured separately. The transmission behavior is decisively influenced by the configuration and the performance of the broker.
2.1 System requirements

The following system requirements are necessary when using MQTT timebased data store:

- *ibaPDA* v7.2.0 or higher
- License for *ibaPDA-Data-Store-MQTT*

The licenses are staggered according to the number of signals that should be sent to an MQTT broker. The number of used data stores is unlimited.

<table>
<thead>
<tr>
<th>Order No.</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>30.671000</td>
<td>ibaPDA-Data-Store-MQTT-16</td>
<td>Data streaming into MQTT broker, 16 signals</td>
</tr>
<tr>
<td>30.671001</td>
<td>ibaPDA-Data-Store-MQTT-64</td>
<td>Data streaming into MQTT broker, 64 signals</td>
</tr>
<tr>
<td>30.671002</td>
<td>ibaPDA-Data-Store-MQTT-256</td>
<td>Data streaming into MQTT broker, 256 signals</td>
</tr>
<tr>
<td>30.671003</td>
<td>ibaPDA-Data-Store-MQTT-1024</td>
<td>Data streaming into MQTT broker, 1024 signals</td>
</tr>
<tr>
<td>30.671010</td>
<td>upgrade-ibaPDA-Data-Store-MQTT-16 to 64</td>
<td>Signal extension 16 signals to 64 signals</td>
</tr>
<tr>
<td>30.671011</td>
<td>upgrade-ibaPDA-Data-Store-MQTT-64 to 256</td>
<td>Signal extension 64 signals to 256 signals</td>
</tr>
<tr>
<td>30.671012</td>
<td>upgrade-ibaPDA-Data-Store-MQTT-256 to 1024</td>
<td>Signal extension 256 signals to 1024 signals</td>
</tr>
</tbody>
</table>

Table 1: Available licenses for the MQTT data store
3 Data store configuration

3.1 Add a data store

The dialog for data storage configuration can be opened in the Configure – Data storage main menu or by clicking on the button in the main toolbar.

In order to add a new data store, click on the blue link Add data store in the tree structure. You can also right-click on the data store node in the tree structure and choose Add data store from the context menu.

Select MQTT timebased data store for the recording of timebased data to an MQTT broker.

![Fig. 2: Add a data store](image-url)
3.2 MQTT data store

Fig. 3: Configure MQTT data store

Locked
A data store can be locked in order to prevent an accidental or unauthorized change of settings.

Active
A data store must be enabled in order to work. However, you can configure various data stores and disable data stores that are not required.

Data store name
You can enter a name for the data store here.

Broker address
Enter the address of the broker here. The address can be an IP address, a host name or a URL.

Port
Port that is to be used for the connection. The standard MQTT port is 1883, or 8883 when using SSL.

Client ID
When connecting to a broker, each client must choose a unique name that is used only once for this broker. The client ID is pre-set by ibaPDA automatically.

Base Topic Path
This path is added in front of each registered topic (optional setting). The topic configuration can be made easier if all topics reside within a common path.
Note: Do not use a leading forward slash “/” in a topic, because this would add an empty path entry to the topic. Two separators next to each other (“/”) are also not allowed.

**Identifier**
The identifier is a text-based value that can be included in the MQTT messages sent to the broker. For the subsequent processing of data, this may be useful for distinguishing between several ibaPDA systems that write to the same broker.

**Persistent session**
If the client is disconnected, it receives the last values sent to the broker when reconnecting with the same client ID as before. For this purpose, the QoS must be set to “Exactly once” or “At least once” not only for ibaPDA, but also for the sender, which delivers the values to the broker.

**Keep Alive**
The time for sending a *Keep Alive* telegram to the broker to make sure that the connection is still online.

**QoS**
*Quality of Service*, which is used when registering to a topic on the broker. The possible values are:
- At most once (messages can be dropped)
- At least once (messages are repeated if an acknowledgement is not sent within a certain time)
- Exactly once (a secured handshake protocol for each message sent)

*<Test connection>*
When clicking the *<Test connection>* button, ibaPDA will try to establish a connection to the MQTT broker, using the selected security settings.

**Buffering options**
The data is buffered if the connection from ibaPDA to the MQTT broker and the connection from the MQTT client to the broker are set to use a *persistent session*, with QoS *At least once* or *Exactly once*. If the sessions are not marked as *persistent*, the client and the broker will remove all values that are not sent on connection loss. If QoS is set to *At most once* data is not buffered. So if the broker gets multiple messages to one topic before sending out data to the clients, only the newest message is sent out.

As MQTT messages themselves contain no time information, it is recommended to use the JSON format and add the timestamp metadata, so the client can retrieve that information together with the value.

**Maximum memory buffer size**
The maximum memory buffer size determines how much data is buffered in ibaPDA in the event of a temporary connection loss with the MQTT broker. The buffer is held in RAM. The buffer contents are deleted when the acquisition is restarted.

**Configured maximum memory buffer size**
This field shows the time span that can be buffered in the buffer with the configured settings. Specification in d.hh:mm:ss.
Pause after sending ... messages (for 10 ms)

ibaPDA pauses sending the telegrams after the number of telegrams set here is sent to the broker without interruption. This should avoid possible problems of the broker with a continuous data stream. Regarding the number, take into account that each topic is treated as a separate telegram. Set the value to 0 to disable the feature.

Use file buffering

If the file buffer is enabled, the buffer is not only stored in memory (RAM), but written to files. More information about the behavior of the buffer memory or file buffer can be found in chapter ➔ File buffer behavior, page 13.

If the Use file buffering option is enabled, the maximum buffer size (RAM) can be reduced to 1 MB instead of the default size of 4 MB.

File storage path

Use the Browse button <...> to select a directory where the buffer files will be stored.

Maximum file buffer size

The maximum disk space in GB that the buffer files may occupy. If this size is exceeded, the oldest data is deleted from the memory.

Maximum file buffer time

The maximum age of the buffered data in hours. Data older than this is removed from memory.

Configured maximum file buffer size

This field shows the time period that can be temporarily stored in the file buffer with the configured settings (specified in d.hh:mm:ss) and the available space on the hard disk.

Security options

Use authentication

Enter the user name and password as login when connecting to the broker. This option has to be configured accordingly on the broker, otherwise the connection will fail.

Use SSL connection

Use SSL instead of plain TCP connection. The port must be set to 8883 by default. This option has to be configured accordingly on the broker, otherwise the connection will fail.

The Windows Certificate Store is used to store the certificate.

Last Will

Enable Last Will

If this option is enabled, the absence of this client is announced to other clients, if the connection breaks without regular disconnect.

Topic

Enter a path to the topic used for the last will message. The base topic path is not applied to this topic.

Message

Enter a message which should be sent in the text field.
Retain
If this option is enabled, the message in this topic is stored on the broker until it is overwritten.

QoS
The **Quality of Service** used for sending the last will message to other clients. The QoS settings are as described above.

Connect Message
*Connect Message* has the same options as *Last Will*, but the *Connect Messages* are sent on a regular connect or disconnect.

### 3.2.1 Time behavior

The processing of the data written via the MQTT data store, is cyclical. The duration of the cycle corresponds to the "minimum output time".

![Fig. 4: Timing tab](image)

As a result, signal samples that are published faster than the minimum output time are sent bundled in the cycle of the minimum output time.

### 3.2.2 File buffer behavior

The *Use file buffering* option is available for the following types of data stores:

- DB/cloud timebased
- Kafka cluster timebased
- MindSphere timebased
- MQTT timebased

With this option the buffer can be stored in files instead of only in the memory buffer. A greater number of items can therefore be buffered, while potentially consuming less memory.

When data is acquired by *ibaPDA*, it is stored in the memory buffer. If a connection to the configured broker and/or database exists, this data is sent there immediately. If the connection to the broker and/or database is lost, or the data cannot be sent out fast enough, the data is temporarily stored in the memory buffer. If the memory buffer grows beyond the configured size, the data is removed from the memory buffer and, if enabled, stored in the file buffer. When the
file buffer grows beyond the configured size, or the stored values are older than the configured maximum time, the oldest values are removed.

When sending data, always the oldest data is sent out first. So when a connection is reestablished after some time, *ibaPDA* starts sending the oldest values, while the newer values are still stored in the buffer.

When the acquisition is stopped, if a connection to a broker and/or database exists and the file buffering is not currently used, *ibaPDA* tries to send out the remaining data. If this takes more than 30 seconds, the process is stopped. If file buffering is currently used, this step is skipped, as sending out all data would take too long anyway. All remaining values are then stored in the file buffer.

When the acquisition is started and there are still files left from the previous session, these values are put into the queue again and sent out. If the configuration was changed, the stored values are still sent in the format defined by the configuration they were recorded with. If this is not desired, the files have to be manually removed from the storage path before starting the acquisition.

### 3.3 Configuring topics

If you highlight the *Topics* node, you can define the topics that are sent to the broker. Different formats are available, depending on the information required for subscribers.

There are buttons above the list to configure the topics:

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>✉️</td>
<td>Manually add a new topic</td>
</tr>
<tr>
<td>✗</td>
<td>Remove selected topic</td>
</tr>
<tr>
<td>☛</td>
<td>Copy cell content to the clipboard</td>
</tr>
<tr>
<td>☁️</td>
<td>Paste data from the clipboard</td>
</tr>
</tbody>
</table>

*Fig. 5: Configuring topics*

If the configuration was changed, the stored values are still sent in the format defined by the configuration they were recorded with. If this is not desired, the files have to be manually removed from the storage path before starting the acquisition.
Move selected topic up
Move selected topic down

Note
Use <Shift> and <Ctrl> to select multiple cells and copy/paste/delete them.

Meaning of the columns:

Active
Writing in a topic can be enabled/disabled by enabling the checkbox Active.

Name
Enter an unambiguous name for the topic here.

Metadata
If one of the JSON formats was chosen as the data format for the topic, you can choose metadata in the drop-down menu that is saved with the topic.
Available for selection are: Unit, comment 1 + 2, timestamp, signal name, signal ID, identifier. Select the desired metadata with a check mark.

Signal reference
In the drop-down menu, select whether the signal ID or the signal name should be used as a signal reference.

Data format
Choose a data format in the drop-down menu:
- Single value as text: A single signal is written under this topic into the broker and formatted as plain text.
- Single value as binary: A single signal is written under this topic into the broker and saved in binary format.
- JSON (per signal): An individual signal is written under this topic as a JSON-formatted line into the broker and provided with optional metadata.
- JSON (grouped): Several signals are written under this topic as a JSON-formatted line into the broker and provided with optional metadata.
You find examples of the description of the JSON data formats in chapter Description of the data formats, page 25.

Signals
In order to configure the signals that are to be written in a topic, select the topic in the structure tree or click on the <...> button in the Signals column.
In the following dialog, you assign the desired signals to the Topics using the storage profiles. See chapter «Signal selection», page 17.
4 Signal selection

To enable signals to be recorded, they must be assigned to a topic by using a storage profile of the Time type. Click in the Topics node, in the Signals column of the topic list on the <...> button to access the signal selection dialog.

Fig. 7: Calling up the signal selection

Or select a topic in the tree structure.

Fig. 8: Signal selection dialog

Note Additional information about the storage profiles can be found in the ibaPDA manual, part 5.

Select the topic to which you would like to assign certain signals and select a storage profile in the profile list. Set a check mark in the selection fields next to the signals which you would like to assign to this profile.
For the single value formats, only the first signal is used for writing if more than one signal is selected. All other signals are ignored.

MQTT data stores are licensed per number of written signals. The current number of selected signals in all MQTT data stores is shown at the bottom of the dialog, similar to the number of configured signals in the I/O Manager.

The length of the bar corresponds to the licensed number of signals. In the example above, a maximum of 1024 signals can be written via MQTT data stores. 10 signals are currently already used.
5 Trigger mode

The description applies to the following types of data stores:

- Database/cloud timebased
- Kafka cluster timebased
- Mindsphere timebased
- MQTT timebased.

In the *Trigger Mode* node, you determine when data is recorded.

![Image of Trigger mode settings](image_url)

**Start trigger**

You initially choose whether you would like to continuously record or it should be fired by a trigger.

**Unconditional**

The data is continuously recorded with this selection. In this case, the recording will start immediately at the start of the measurement or when pressing the "GO" button.

**Trigger on signal**

If you want the trigger to fire on a measured signal or a virtual signal, you need to check *Trigger on signal* in the option field. In the fields next to this, define the properties of the trigger signal.

- **Field 1:** Drop-down list for signal selection (available analog and digital signals)
- **Field 2:** Drop-down list for selecting edges or levels
- **Field 3:** Drop-down list for selecting the trigger level value given in the specific physical unit (field 3 is only enabled in case of analog trigger signals)
Both analog and digital signals can serve as triggers. The signal to trigger on is to be selected from the drop-down lists (see picture below, field 1). In the drop-down list, you will find the well-known signal tree containing available signals. Select the signal you want to use as trigger signal.

![Fig. 10: Configuring "Trigger on signal"](image)

Depending on whether a digital or an analog signal was selected, the fields 2 or 3, respectively, are offered allowing the trigger event to be defined more specifically.

As for analog signals, you can choose between level or edge triggers including a predefined level (field 3).

![Fig. 11: Configuring "Trigger on signal", analog signal, edge or level](image)

As for digital signals, you can choose between level or edge triggers including the 2 levels logical 0 (FALSE) and logical 1 (TRUE).

![Fig. 12: Configuring "Trigger on signal", digital signal](image)

**Trigger every ...**

If you want to use a start trigger always at a certain time regularly, you can check the “Trigger every ... minutes starting at ...” option. Enter the period given in minutes, or select it from the
input field. Value range is from 0 to 1440, which equals one day. Then enter or select the start time for the first trigger. Value range is from 00:00 to 23:59, which equals one day.

One sample on change of
When the value of the selected signal changes, a sample is recorded. The recording will stop after one sample, until the next signal change is detected. A deadtime can be configured to determine a minimum amount of time between samples. Before the deadtime has elapsed, no new sample will be recorded.

Pre-trigger time
You can configure a pre-trigger time and then the recording begins by the pre-trigger time before the trigger event. If the trigger condition is met, the incoming data is added to the data buffered during the pre-trigger time.

Trigger dead time
This property is available for the start triggers “Trigger on signal”, “Trigger every …” and "One sample on change of". The trigger dead time determines the time of suppressing subsequent triggers after a trigger occurred.

If the dead time, for instance, is set to 5 seconds, all other triggers are ignored for the duration of 5 seconds after the first trigger occurrence.

Trigger at the start of the acquisition
If you want the recording to start immediately at acquisition start or as soon as you apply a new data storage configuration, you also need to check the Trigger on acquisition start option. If you do not enable this option, the recording first starts once the trigger is fired.

If start trigger occurs again while file is already recording, then:
You can determine here what should happen if a new start trigger occurs while a recording is already running.

- Ignore it:
  Selecting this option will cause the system to ignore any new start trigger during a running recording for as long as the stop trigger occurs

- Extend recording time:
  If this option is enabled, it extends the duration of the running recording upon occurrence of another start trigger during an ongoing recording. This occurs as often as set in the "Maximum number of extensions on single file" field. If the max. number of extensions is reached, all subsequent start triggers will be ignored. Of course, the recording is stopped immediately by any stop trigger.

Stop trigger
The settings for the stop trigger are made in the same way as those for the start trigger. Here, both analog and digital signals can also be used as triggers.

Trigger after recording of x hours x minutes x seconds
Here you can configure a time span according to which the recording is ended - after the occurrence of the start trigger.
**Trigger on signal**
See explanation for start trigger above.

**Post trigger time**
You can configure a post trigger time and then the recording ends by the post trigger time after the stop trigger event.
6 Diagnostics

6.1 Data storage status

The data storage status window shows the current status of the data stores.

Fig. 13: Example of data storage status window

All defined data stores and their respective status are displayed here, depending on the data store, with server address, acquisition duration, write speed, etc.

The icon in front of the name indicates the current status of the storage:

- Wait for the start trigger (only for triggered recording)
- Recording in progress
- Post-trigger phase; stop trigger occurred, but acquisition continues until the post-trigger time is over

Disabled or faulty data store is indicated by a red cross in the data store icon.

Right-clicking on this node allows you to manually send a start or stop trigger.
6.2 Diagnostics of data stores

The Diagnostics node in the data storage tree offers information about the system load by the data stores. The measurement must be running.

The performance values of all data stores are shown in the table. There is one row per data store. The rows are grouped according to the threads that write the data.

In each group row is the name of the thread and (in brackets) its share of the load. The load average is displayed by default. But, you can switch between the average and actual value using the context menu.

The Disk column indicates the respective target to which the data is written, for example a hard disk partition, the address of the database, the address of the Kafka cluster, etc. The Write speed indicates how fast the data is written.

The Memory buffer (kB) columns indicate how much data is buffered in ibaPDA. The columns File buffer (MB) indicate how much data is buffered in the file buffer.

The Acquisition Thread load column indicates various information depending on the data stores. For time-based data stores, the Acquisition Thread load column indicates the amount of time needed for the run length encoding and writing to a disk or in a Kafka cluster.

For database/cloud, MQTT and MindSphere data stores, the column indicates the load caused by the analysis of the triggers and creation of the row data.

For HD data stores, the partial processing time will be displayed, that is used for the creation of the data to be written on the HD server. These values already contain the run length encoding for time-based stores, event trigger calculation for event-based stores and the calculation of the length-based data for length-based stores.

Additional information about diagnostics can be found in the ibaPDA manual, part 5.
7 Appendix

7.1 Description of the data formats

7.1.1 Data format JSON (grouped)

The following explains the setup of the data format “JSON (grouped)” using an example with three signals. The signals are defined as follows:

![Signals in the analog tab](image)

**JSON example for signal ID as a signal reference**

```json
{
    "[0:0]": "Actual value",
    "[0:0].ID": "[0:0]",
    "[0:0].Name": "Signalname_0",
    "[0:0].Unit": "Unit_0",
    "[0:0].Comment1": "Example_comment1_0",
    "[0:0].Comment2": "Example_comment2_0",
    "[0:1]": "Actual value",
    "[0:1].ID": "[0:1]",
    "[0:1].Name": "Signalname_1",
    "[0:1].Unit": "Unit_1",
    "[0:1].Comment1": "Example_comment1_1",
    "[0:1].Comment2": "Example_comment2_1",
    "[0:2]": "Actual value",
    "[0:2].ID": "[0:2]",
    "[0:2].Name": "Signalname_2",
    "[0:2].Unit": "Unit_2",
    "[0:2].Comment1": "Example_comment1_2",
    "[0:2].Comment2": "Example_comment2_2",
    "Timestamp": "2020-01-21T13:10:53.0002189Z",
    "identifier": "My identifier"
}
```

Red: optional signal-related metadata
Green: optional group-related metadata
JSON example for signal name as a signal reference

```
{
  "Signalname_0": "Actual value",
  "Signalname_0.ID": "[0:0]",
  "Signalname_0.Name": "Signalname_0",
  "Signalname_0.Unit": "Unit_0",
  "Signalname_0.Comment1": "Example_comment1_0",
  "Signalname_0.Comment2": "Example_comment2_0",
  "Signalname_1": "Actual value",
  "Signalname_1.ID": "[0:1]",
  "Signalname_1.Name": "Signalname_1",
  "Signalname_1.Unit": "Unit_1",
  "Signalname_1.Comment1": "Example_comment1_1",
  "Signalname_1.Comment2": "Example_comment2_1",
  "Signalname_2": "Actual value",
  "Signalname_2.ID": "[0:2]",
  "Signalname_2.Name": "Signalname_2",
  "Signalname_2.Unit": "Unit_2",
  "Signalname_2.Comment1": "Example_comment1_2",
  "Signalname_2.Comment2": "Example_comment2_2",
  "Identifier": "My identifier"
}
```

Red: optional signal-related metadata
Green: optional group-related metadata
### 7.1.2 Data format JSON (per signal)

The following explains the setup of the data format “JSON (per signal)” using an example with three signals. The signals are defined as follows:

<table>
<thead>
<tr>
<th>Signalname</th>
<th>Expression</th>
<th>Unit</th>
<th>Comment 1</th>
<th>Comment 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signalname 0</td>
<td>GeneratesSignal(0)</td>
<td>Unit 0</td>
<td>Example_comment1_0</td>
<td>Example_comment2_0</td>
</tr>
<tr>
<td>Signalname 1</td>
<td>GeneratesSignal(1)</td>
<td>Unit 1</td>
<td>Example_comment1_1</td>
<td>Example_comment2_1</td>
</tr>
<tr>
<td>Signalname 2</td>
<td>GeneratesSignal(2)</td>
<td>Unit 2</td>
<td>Example_comment1_2</td>
<td>Example_comment2_2</td>
</tr>
</tbody>
</table>

![Fig. 16: Signals in the analog tab](image-url)
JSON example for signal ID as a signal reference

```
{
  "Signal": "[0:0]",
  "Value": Actual value,
  "ID": "[0:0]",
  "Name": "Signalname_0",
  "Unit": "Unit_0",
  "Comment1": "Example_comment1_0",
  "Comment2": "Example_comment2_0",
  "Timestamp": "2020-01-21T13:26:50.8784074Z",
  "Identifier": "My identifier"
}
```

```
{
  "Signal": "[0:1]",
  "Value": Actual value,
  "ID": "[0:1]",
  "Name": "Signalname_1",
  "Unit": "Unit_1",
  "Comment1": "Example_comment1_1",
  "Comment2": "ExampleComment2_1",
  "Timestamp": "2020-01-21T13:26:50.8784074Z",
  "Identifier": "My identifier"
}
```

```
{
  "Signal": "[0:2]",
  "Value": Actual value,
  "ID": "[0:2]",
  "Name": "Signalname_2",
  "Unit": "Unit_2",
  "Comment1": "ExampleComment1_2",
  "Comment2": "ExampleComment2_2",
  "Timestamp": "2020-01-21T13:26:50.8784074Z",
  "Identifier": "My identifier"
}
```

Red: optional signal-related metadata
JSON example for signal name as a signal reference

```
{
    "Signal": "Signalname_0",
    "Value": "Actual value",
    "ID": "[0:0]",
    "Name": "Signalname_0",
    "Unit": "Unit_0",
    "Comment1": "Example_comment1_0",
    "Comment2": "Example_comment2_0",
    "Timestamp": "2020-01-21T13:36:37.5310016Z",
    "Identifier": "My identifier"
}
```

```
{
    "Signal": "Signalname_1",
    "Value": "Actual value",
    "ID": "[0:1]",
    "Name": "Signalname_1",
    "Unit": "Unit_1",
    "Comment1": "Example_comment1_1",
    "Comment2": "Example_comment2_1",
    "Timestamp": "2020-01-21T13:36:37.5310016Z",
    "Identifier": "My identifier"
}
```

```
{
    "Signal": "Signalname_2",
    "Value": "Actual value",
    "ID": "[0:2]",
    "Name": "Signalname_2",
    "Unit": "Unit_2",
    "Comment1": "Example_comment1_2",
    "Comment2": "Example_comment2_2",
    "Timestamp": "2020-01-21T13:36:37.5310016Z",
    "Identifier": "My identifier"
}
```

Red: optional signal-related metadata
8 Support and contact

Support

Phone: +49 911 97282-14
Fax: +49 911 97282-33
Email: support@iba-ag.com

Note

If you require support, indicate the serial number (iba-S/N) of the product or the license number.

Contact

Headquarters

iba AG
Koenigswarterstrasse 44
90762 Fuerth
Germany

Phone: +49 911 97282-0
Fax: +49 911 97282-33
Email: iba@iba-ag.com

Mailing address

iba AG
Postbox 1828
D-90708 Fuerth, Germany

Delivery address

iba AG
Gebhardtstrasse 10
90762 Fuerth, Germany

Regional and Worldwide

For contact data of your regional iba office or representative please refer to our web site