



MQ Telemetry Transport Ethernet Driver FS-8705-101

Chipkin - Enabling Integration

salesgroup1@chipkin.com

Tel: +1 866 383 1657

© 2021 CHIPKIN AUTOMATION SYSTEMS

Driver Version: 0.0.1
Document Revision: 3

TABLE OF CONTENTS

1 MQTT DESCRIPTION3

2 CONNECTION DESCRIPTION4

3 MQTT CONFIGURATION5

 3.1 CREATE CONNECTION.....5

 3.2 CREATE NODE.....6

 3.3 CREATE TASK.....8

 3.4 SAVING THE SERVER CONFIGURATION9

 3.5 RESETTING THE SERVER CONFIGURATION9

4 MQTT TEST TOOLS10

 4.1 HIVEMQ MQTT BROWSER CLIENT10

5 LICENSE11

 5.1 HOW TO GENERATE A PRE-KEY11

 5.2 HOW TO ACTIVATE A PRODUCT KEY12

6 IMPORTING AND EXPORTING CONFIGURATIONS13

 6.1 HOW TO EXPORT THE CONFIGURATION.....13

 6.2 HOW TO IMPORT THE CONFIGURATION13

 6.3 HOW TO IMPORT A PE CONFIGURATION.....14

7 REVISION HISTORY15

APPENDIX A. TROUBLESHOOTING16

 APPENDIX A.1 DEBUGGING A MQTT CONNECTION16

 APPENDIX A.2 USING HIVEMQ MQTT BROWSER CLIENT FOR TESTING16

 APPENDIX A.3 TESTING FIELDSEVER AS A MQTT SUBSCRIBER19

 APPENDIX A.4 TESTING FIELDSEVER AS A MQTT PUBLISHER.....20

APPENDIX B. EXAMPLE CONFIGURATION22

APPENDIX C. MARKETING24

 APPENDIX C.1 CASE STUDY24

 APPENDIX C.2 KEYWORD.....24

APPENDIX D. GLOSSARY OF TERMS25

1 MQTT Description

The MQTT Driver allows the FieldServer to transfer data from devices over Ethernet using the MQTT protocol. The MQTT Driver uses TCP. The default port is 1883 and is configurable.

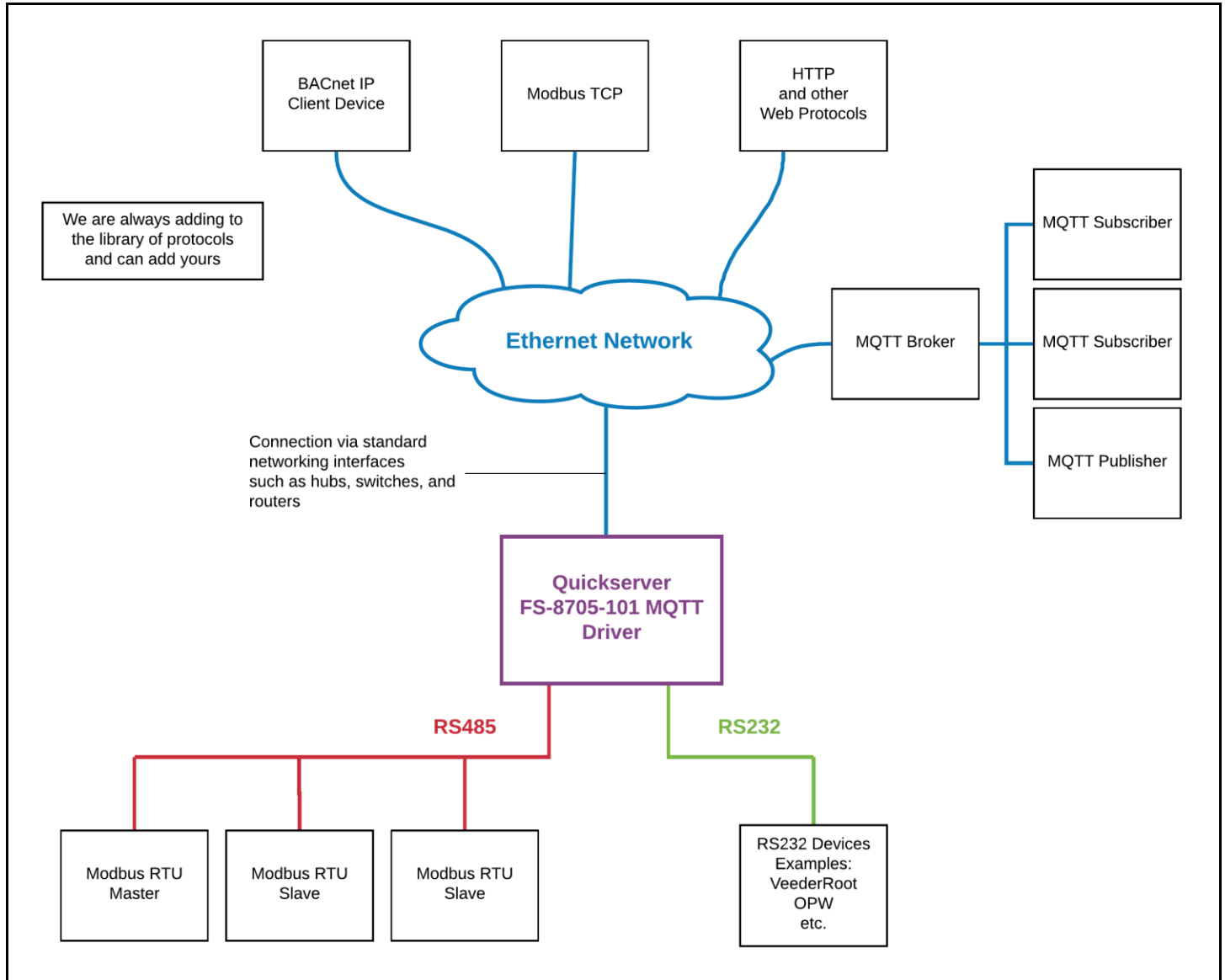
The driver was developed for the streaming MQ Telemetry Transport (MQTT) protocol. MQTT is an OASIS standard messaging protocol for the Internet of Things (IoT). It is designed as an extremely lightweight publish/subscribe messaging transport that is ideal for connecting remote devices with a small code footprint and minimal network bandwidth. See <https://mqtt.org/> for more information.

The FieldServer can emulate both a MQTT Subscriber (Client) and Publisher (Server). When configured as a Subscriber, the MQTT driver will connect to the configured MQTT brokers and subscribe to the configured topics for data. This data is stored on the FieldServer to be mapped to other protocols or simply to be viewed. When configured as a Publisher, the MQTT driver connects to the configured MQTT brokers and publishes data received from other protocols to make them available to MQTT Subscribers.

The information that follows describes how to expand upon the factory defaults provided in the configuration files included with the FieldServer.

2 Connection Description

This block diagram lists common network connections that can monitor MQTT data using other protocols like Modbus® RTU/TCP, BACnet® and HTTP.



3 MQTT Configuration

To configure the MQTT driver, from the home page, visit the following link: http://{IP_ADDRESS}/mqttAEDriver/ui/

To configure the FieldServer, follow the instructions below to add a Connection (information on how to setup the connection), Nodes (MQTT Broker information), and finally Tasks (subscriptions and publishes).

3.1 Create Connection

To set up the FieldServer as a passive MQTT server, first create a connection. The connection contains information about the physical port.

Connections

Connections information subtitle

| Name | Type | Parameters | Actions |
|--|------|------------|---------|
| <div style="display: inline-block; background-color: #007bff; color: white; padding: 5px 15px; border-radius: 3px; text-decoration: none;">Create Connection</div> | | | |

1. Click on the “Create Connection” button to open the Create Connection form.
2. Fill out the fields in the form. The fields are as follows:

| Column Title | Function | Legal Values |
|------------------|--|----------------------|
| Name | Name of the server, used internally as an identifier | Text, must be unique |
| Type | The type of connection this is | ethernet |
| Parameters: Port | The physical port on the FieldServer to use | n1 |

* Bolded values are defaults

3. Click the “Save” button to add the connection.

If successful, the new entry will be populated in the Connections table:

Connections

Connections information subtitle

| Name | Type | Parameters | Actions |
|----------|----------|------------------|---|
| Ethernet | Ethernet | { "port": "n1" } | Edit Delete |

[Create Connection](#)

3.2 Create Node

Follow the instructions below to configure the information to connect to a MQTT Broker.

Nodes

Nodes information subtitle

| Name | Connection | Broker | Keepalive | Client Id | Username | Password | Clean | Actions |
|------|------------|--------|-----------|-----------|----------|----------|-------|---------|
|------|------------|--------|-----------|-----------|----------|----------|-------|---------|

[Create Node](#)

1. Click on the “Create Node” button to open the Create Node form.
2. Fill out the fields in the form. The fields are as follows:

| Column Title | Function | Legal Values |
|--------------|--|---|
| Name | The name of the broker | Text, must be unique |
| Connection | The name of the connection to use. | Text (Use the name of the Connection created in the previous section) |
| Broker | The URL of the MQTT broker in the format of {protocol}://{host}:{port} | Example: mqtt://broker.hivemq.com:1883 |
| KeepAlive | The number of seconds to keep the connection alive, set to 0 to disable | 0-3600, 60 |
| ClientId | The MQTT ClientId that represents this device connecting to the broker | Text, must be unique, Example: fs_qs_mqttAEDriver_01234 |
| Username | The username required by the broker, if any | Text |
| Password | The password required by the broker, if any | Text |
| Clean | Specifies where the connection starts as a new Session or is a continuation of an existing Session | Boolean, True |

***Note*:** Default values are bolded

3. Click on the “Save” button to add the node.

If successful, the new entry will be populated in the Nodes table:

| Nodes | | | | | | | | |
|--|------------|-------------------------------|-----------|----------------------------|----------|----------|-------|---|
| Nodes information subtitle | | | | | | | | |
| Name | Connection | Broker | Keepalive | Client Id | Username | Password | Clean | Actions |
| HiveMq | Ethernet | mqtt://broker.hivemq.com:1883 | 60 | fs_qs_mqttTestClient_01234 | | | | <input type="button" value="Edit"/> <input type="button" value="Delete"/> |
| <input type="button" value="Create Node"/> | | | | | | | | |

Repeat the above steps to add additional universes.

3.3 Create Task

Create tasks to add subscriptions to topics or to add publishing tasks to publish data to the specified topic.

| Tasks | | | | | | | | | |
|---|------|-------------|-------|---------------|------|----------|-----|---------|--|
| Tasks information subtitle | | | | | | | | | |
| Name | Node | Data Broker | Topic | Scan Interval | Type | Template | Qos | Actions | |
| <div style="display: flex; justify-content: space-between; align-items: center;"> Create Task </div> | | | | | | | | | |

1. Click on the “Create Task” button to open the Create Task form.
2. Fill out the fields in the form. The fields are as follows:

| Column Title | Function | Legal Values |
|-------------------|--|---|
| Name | The name of the variable to add. | Text, must be unique |
| Node | The broker to use. | Text (Use the name of a node created in the previous section) |
| DataBroker: Name | The data array in the protocol engine to retrieve the value. | One of the Data Array names |
| DataBroker: Start | The starting offset in the array to retrieve the value | 0 to (“Data_Array_length” - 1) |
| Topic | The MQTT topic to subscribe to or publish to | Text, must not contain MQTT Topic wildcars |
| ScanInterval | How often to subscribe or publish data | 0-3600, 60 |
| Type | The type of task this is. Valid values: subscribe, publish | Subscribe, Publish |
| Template | Currently not use | Text |
| QOS | The quality of server | 0 , 1, 2 |

***Note*:** Default values are bolded

3. Click the “Save” button to add the task.

If successful, the new entry will be populated in the Tasks table:

| Tasks | | | | | | | | |
|-----------------------------|--------|-------------|-------------------------|---------------|-----------|----------|-----|---|
| Tasks information subtitle | | | | | | | | |
| Name | Node | Data Broker | Topic | Scan Interval | Type | Template | Qos | Actions |
| TestSubscriptionTemperature | HiveMq | PE:DA_AI:0 | chipin/test/temperature | 60 | Subscribe | | 0 | Edit Delete |
| TestPublishSetPoint | HiveMq | PE:DA_AI:1 | chipkin/test/SetPoint | 30 | Publish | | 0 | Edit Delete |

[Create Task](#)

Repeat the above steps to add additional variables.

3.4 Saving the Server Configuration

When the configuration is complete, click on the “Save Configuration” button to save all of the updates and changes. For the configuration to take effect, reboot the system.



3.5 Resetting the Server Configuration

To clear the configuration and start over, click the “Reset Configuration” button. Then follow the instructions in the sections above to create new connections, nodes, and tasks.



4 MQTT Test Tools

A list of MQTT testing tools that you can use to test the functionality of your system.

4.1 HiveMQ MQTT Browser Client

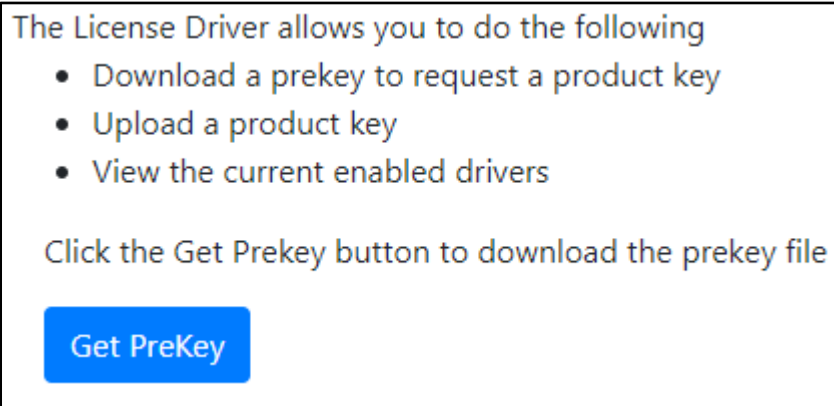
The MQTT is a MQTT client that runs in the browser. It can be used to test subscriptions by publishing to topics that the FieldServer is subscribe to and can also subscribe to topics that the FieldServer is publishing to. For more information, see [The Public MQTT Broker by HiveMQ - Check out our MQTT Demo](#)

5 License

Some drivers such as MQTT require a license product key. To generate this license product key a pre-key from the hardware is required.

5.1 How to generate a pre-key

1. Goto the license page http://{IP_ADDRESS}/chipkinLicenseDriver/ui/
2. Click the Get Pre-Key button.



A pre-key.txt file will be created and downloaded to your system. Send this pre-key.txt and your Job number (FSE1234) to Chipkin support.

5.2 How to activate a product key

Chipkin support can generate a license product key from the hardware pre-key. The product key will be sent as a text file via email.

1. Goto the license page http://{IP_ADDRESS}/chipkinLicenseDriver/ui/
2. Click “Browse” button and select the productkey-XXXXX.txt file provided to you by Chipkin Support.
3. Click the “Upload Product Key” button and wait for the product key to finish uploading.

Upload a product key. Select the product key to upload, then click the Upload Product Key button

The list of enabled product codes can be viewed in the “Enabled Product Codes” list.

Enabled Product Codes

The list of product codes that have been enabled by uploaded product keys

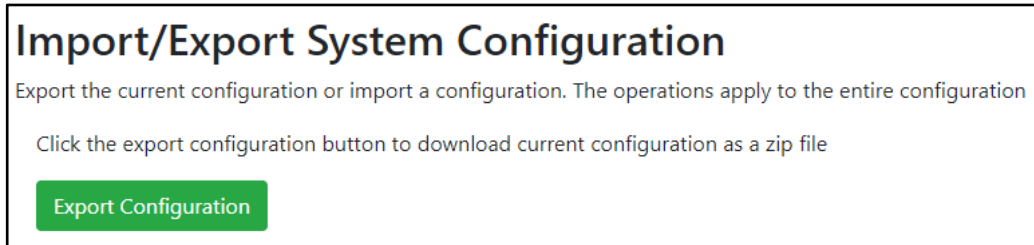
No product keys have been uploaded.

6 Importing and Exporting Configurations

It is possible to export the current configuration to back it up or simply to make some edits. Users can also import either the entire configuration via a zip file or a PE (Protocol Engine) configuration.

6.1 How to Export the Configuration

1. Goto the system configuration page http://{IP_ADDRESS}/chipkinConfiguration/ui/
2. Click the Export Configuration button.



6.2 How to Import the Configuration

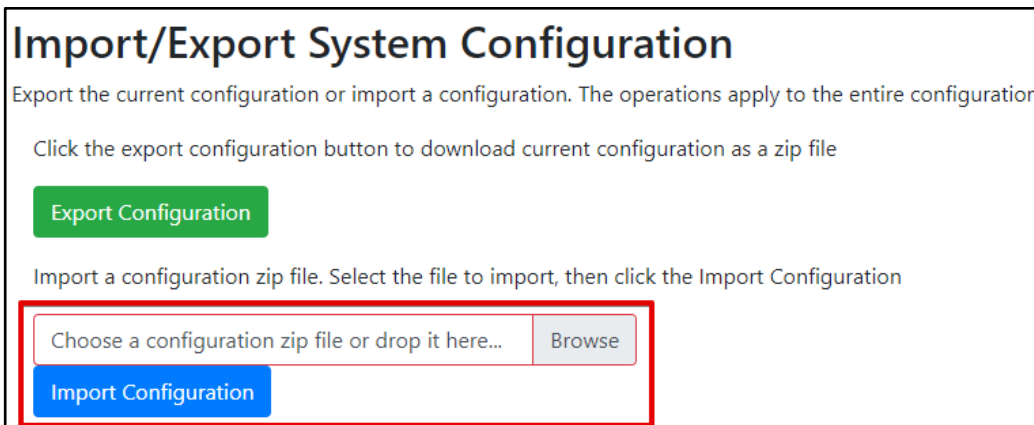
The file to import the configuration must be a zip file. The zip file should contain the following folders:

- ae - this folder contains any configuration files for the ae configuration
- documents - this folder contains any driver specific documents. For example, license product keys, etc.
- pe - this folder contains one config.csv file for the pe configuration.

To make sure the folder directory is correct, do an Export first, then extract the files, edit them, then zip them up again.

To import the configuration:

1. Goto the system configuration page http://{IP_ADDRESS}/chipkinConfiguration/ui/
2. Click the "Browse" button in the "Import/Export System Configuration" section and select the zip file containing the configuration to import.
3. Click the "Import Configuration" button and wait for the configuration to finish importing.
4. If successful, a success message will appear prompting a reboot of the Fieldserver for the changes to take effect.



6.3 How to Import a PE Configuration

It is possible to import a PE (Protocol Engine) configuration separately. To import a PE configuration:

1. Goto the system configuration page http://{IP_ADDRESS}/chipkinConfiguration/ui/
2. Click the “Browse” button in the “Import Specific Configuration” section and select the csv file containing the pe configuration to import.
3. Click the “Import PE Configuration” button and wait for the configuration to finish importing.
4. If successful, a success message will appear prompting a reboot of the Fieldserver for the changes to take effect.

Import Specific Configuration

Use the following control to import a specific portion of the configuration, this includes:

- Import PE Configuration

Import a pe configuration csv file. Select the file to import, then click the Import PE Configuration

7 Revision History

| Date | Resp | Doc. Rev. | Comment |
|--------------|------|--------------|--------------------------|
| 08 Dec 2020 | ACF | 1 | Created initial document |
| 09 Dec 2020 | ACF | 2 | Removed Firmware Section |
| 02 June 2021 | YC | 3 | Updated document format |
| | | | |
| | | | |
| | | | |

Appendix A. Troubleshooting

Appendix A.1 Debugging a MQTT connection

- If the FieldServer is not receiving any data from subscriptions, verify the MQTT Broker connection is correct and that the topic exists on the Broker.
- If the FieldServer is not publishing any data, verify the MQTT Broker connection is correct and check that the data point being published exists.
- Check if the MQTT Broker requires a username and password.
- Verify comms by taking a wireshark log or a FieldServer diagnostics log.

Appendix A.2 Using HiveMQ MQTT Browser Client for Testing

Follow the steps in this section to setup the HiveMQ MQTT Browser Client tool to test MQTT devices.

1. Access the Browser Client: [MQTT Websocket Client \(hivemq.com\)](https://www.hivemq.com/mqtt/websocket-client/)
2. Fill out the Broker Connection information:

The screenshot shows the 'Connection' configuration interface for the HiveMQ MQTT Browser Client. At the top right, there is a red circle and the text 'disconnected'. The form contains several input fields and a 'Connect' button:

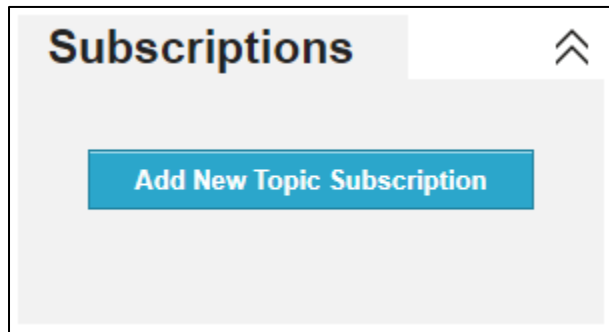
- Host:** broker.hivemq.com
- Port:** 8000
- ClientID:** clientId-KGKIHuUSwW
- Username:** (empty)
- Password:** (empty)
- Keep Alive:** 60
- Clean Session:**
- Last-Will Topic:** (empty)
- Last-Will QoS:** 0
- Last-Will Retain:**
- Last-Will Message:** (empty text area)

- **Host:** Fill out the host name of the MQTT Broker. If using the example broker, use broker.hivemq.com
- **Port:** This tool uses websockets, so connect using port 8000.
- Keep all the other options with their default values.

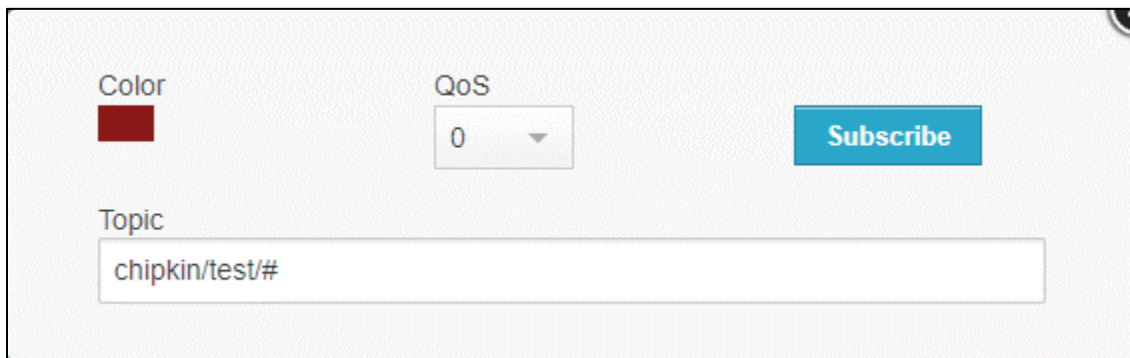
Finally click the “Connect” button to connect to the Broker. If successful, you will see the following:



3. Setup a Subscription or Publish:
 - a. To setup a subscription:
 - i. Click the “Add New Topic Subscription” button



- ii. Fill out the form to subscribe to a topic:

A screenshot of a subscription form. It features three main components: a "Color" field with a red square, a "QoS" dropdown menu set to "0", and a blue "Subscribe" button. Below these is a "Topic" text input field containing the text "chipkin/test/#".

- **Color:** Choose a color to represent the subscription.
- **QoS:** Select the Quality of Service. If unsure, select 0.
- **Topic:** Add the topic to subscribe to. If testing all points, you can use wildcards. In the image above, this will record all data points published to chipin/test/{any topic}. Otherwise, specify the specific topic, for example: chipin/test/SetPoint.

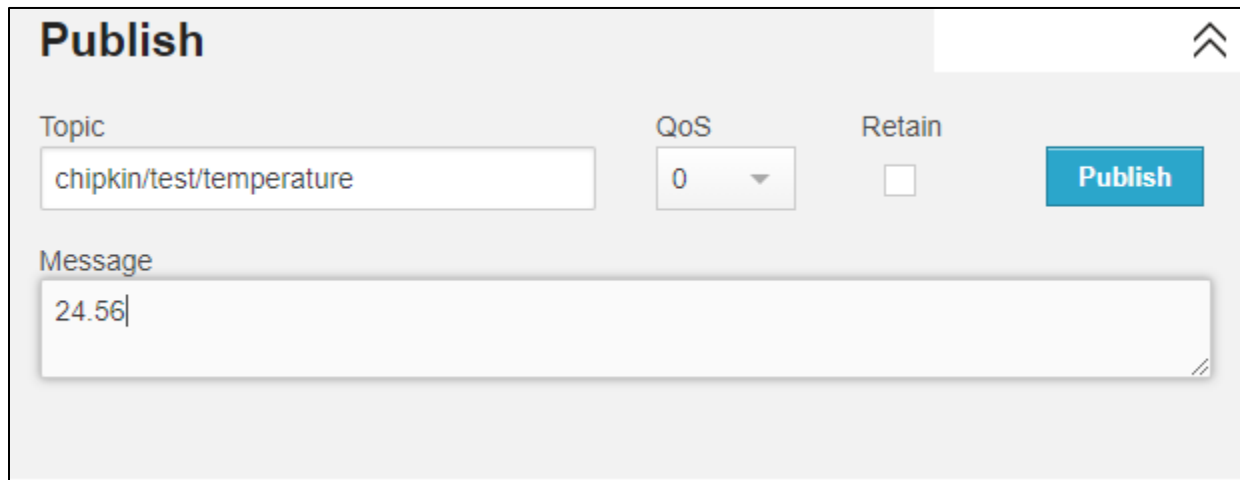
Click the “Subscribe” button to add the subscription. New data points will appear in the Messages section:



Messages

2020-12-08 09:17:21 Topic: chipkin/test/SetPoint Qos: 0
 {"value":50,"lastModified":1607447836753,"id":"mqttAEDriver/HiveMq/Test
 PublishSetPoint"}

- b. To publish to a topic:
- i. Fill out the Publish form:



Publish

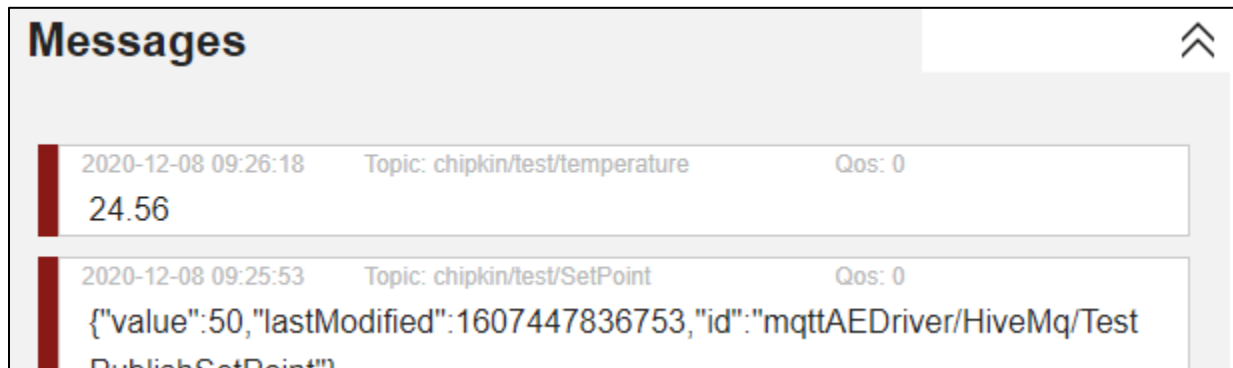
Topic: QoS: Retain:

Message:

- **Topic:** The topic to publish to (cannot contain wildcards)
- **QoS:** Select the Quality of Server, set to 0 if unsure.
- **Message:** The data value to publish.

When done, click the “Publish” button.

If the client has been setup with a subscriptions for the same topic that you are publishing to, the published message will show up in the Messages section.



Messages

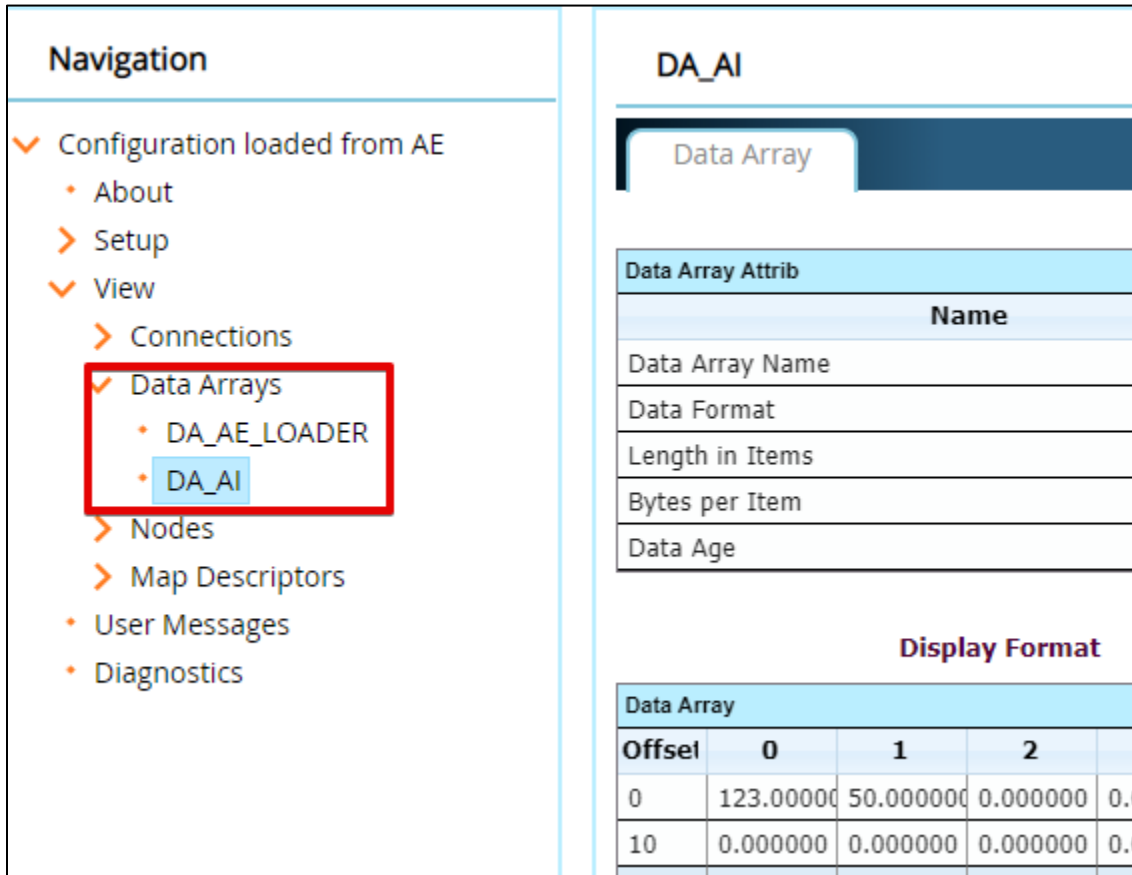
2020-12-08 09:26:18 Topic: chipkin/test/temperature Qos: 0
24.56

2020-12-08 09:25:53 Topic: chipkin/test/SetPoint Qos: 0
{"value":50,"lastModified":1607447836753,"id":"mqttAEDriver/HiveMq/Test
PublishSetPoint"}

Appendix A.3 Testing Fieldserver as a MQTT Subscriber

The following instructions are how to confirm that a FieldServer that has been configured as a MQTT Subscriber is working correctly.

1. Follow the instructions in Appendix A.2 to use HiveMQ MQTT Browser Client to publish data to the FieldServer configured as a MQTT Subscriber.
2. Access the Data Array page on the FieldServer interface



3. Follow the instructions in Appendix A.2 to publish a value using the HiveMQ MQTT Browser Client.
4. Verify that the values update in the correct Data Array offsets.

| Data Array | | | | | |
|------------|------------|-----------|----------|----------|----------|
| Offset | 0 | 1 | 2 | 3 | 4 |
| 0 | 123.000000 | 50.000000 | 0.000000 | 0.000000 | 0.000000 |
| 10 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 20 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 30 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 40 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |

Appendix A.4 Testing Fieldserver as a MQTT Publisher

The following instructions are how to confirm that a FieldServer that has been configured as a MQTT Publisher is working correctly.

1. Follow the instructions in Appendix A.2 to use HiveMQ MQTT Browser Client to subscribe to a topic that the FieldServer is publishing to.
2. Access the Data Array page on the FieldServer interface

The screenshot shows the FieldServer interface. On the left is a 'Navigation' sidebar with a tree view. The 'Data Arrays' item is highlighted with a red box, and its sub-item 'DA_AI' is also highlighted. On the right, the 'DA_AI' configuration page is shown, featuring a 'Data Array' tab and a table of attributes.

| Data Array | | | | |
|-------------------|-----------|-----------|----------|-----|
| Data Array Attrib | | | | |
| Name | | | | |
| Data Array Name | | | | |
| Data Format | | | | |
| Length in Items | | | | |
| Bytes per Item | | | | |
| Data Age | | | | |
| Display Format | | | | |
| Data Array | | | | |
| Offset | 0 | 1 | 2 | |
| 0 | 123.00000 | 50.000000 | 0.000000 | 0.0 |
| 10 | 0.000000 | 0.000000 | 0.000000 | 0.0 |

3. Edit the value in the Data Arrays

| Data Array | | | | |
|------------|-----------|-----------|----------|-----|
| Offset | 0 | 1 | 2 | |
| 0 | 123.00000 | 65.000000 | 0.000000 | 0.0 |
| 10 | 0.000000 | 0.000000 | 0.000000 | 0.0 |
| 20 | 0.000000 | 0.000000 | 0.000000 | 0.0 |
| 30 | 0.000000 | 0.000000 | 0.000000 | 0.0 |

4. Verify in the Message section of the HiveMQ MQTT Browser client that the value was published:



The screenshot displays the 'Messages' section of the HiveMQ MQTT Browser client. It shows two messages published to the topic 'chipkin/test/SetPoint' with a QoS of 0. Each message is a JSON object containing 'value', 'lastModified', and 'id' fields.

| Timestamp | Topic | QoS | Message Content |
|---------------------|-----------------------|-----|---|
| 2020-12-08 09:33:51 | chipkin/test/SetPoint | 0 | <pre>{"value":65,"lastModified":1607448806848,"id":"mqttAEDriver/HiveMq/TestPublishSetPoint"}</pre> |
| 2020-12-08 09:33:21 | chipkin/test/SetPoint | 0 | <pre>{"value":50,"lastModified":1607447836753,"id":"mqttAEDriver/HiveMq/TestPublishSetPoint"}</pre> |

Appendix B. Example Configuration

```
{
  "ae": {
    "mqttAEDriver": {
      "connections": [
        {
          "name": "Ethernet",
          "type": "ethernet",
          "parameters": {
            "port": "n1"
          }
        }
      ],
      "nodes": [
        {
          "name": "Mq",
          "connection": "Ethernet",
          "broker": "mqtt://broker.hivemq.com:1883"
        }
      ],
      "tasks": [
        {
          "name": "SubTest",
          "node": "Mq",
          "scanInterval": 30,
          "topic": "chipkin/test/1",
          "type": "subscribe",
          "qos": 0,
          "dataBroker": {
            "pe": {
              "Name": "DA_AI",
              "Start": "30"
            }
          }
        }
      ]
    }
  }
}
```

```
    },  
    {  
      "name": "PubTest",  
      "node": "Mq",  
      "scanInterval": 30,  
      "topic": "chipkin/test/2",  
      "type": "publish",  
      "qos": 0,  
      "dataBroker": {  
        "pe": {  
          "Name": "DA_AI",  
          "Start": "31"  
        }  
      }  
    }  
  ]  
}  
}
```

Appendix C. Marketing

Appendix C.1 Case Study

A series of case studies for MQTT can be found here

[ToDo] – Add Case Study or link to Case Studies...

Appendix C.2 Keyword

MQTT, IOT, Broker, Pub/Sub, Subscriber, Publisher

Appendix D. Glossary of Terms

- **MQTT** – Message Queuing Telemetry Transport
- **TCP** - Transmission Control Protocol