MQTT driver for the REXYGEN system (the MQTTDrv module)

User guide

REX Controls s.r.o.

Version 3.0.4
Plzeň (Pilsen), Czech Republic
2025-03-27

Contents

Chapter 1

The MQTTDrv driver and the REXYGEN system

1.1 Introduction

This manual describes the MQTTDrv driver for handling of communication over the MQTT protocol within the REXYGEN system. The driver was developed by the REX Controls company.

MQTT is a lightweight messaging protocol for small sensors and mobile devices, optimized for high-latency or unreliable networks. Typical architecture based on MQTT communication protocol consists of a single Broker, multiple devices that produce messages and hence act as Publishers and devices that consume the messages and hence act as Subscribers. See Fig. 1.1. The messages are organized into topics. Every message has a topic specified when being published. The Broker collects all the messages from the Publishers and delivers them to the Subscribers that showed an interest in the topic by sending a subscription request to the Broker beforehand. See the MQTT specification [1] for more details.

From the perspective of the MQTT protocol REXYGEN can be in the role of a Publisher and/or a Subscriber. REXYGEN is not meant to be used as a Broker but should be compatible with any MQTT Broker implementation supporting MQTT Version 3.1.1.

1.2 Installation of the driver on the target device

1.2.1 Windows machines

The target part of the driver, which is used for running REXYGEN MQTTDrvon Windows 10/11 is included in the Development tools of the REXYGEN system.

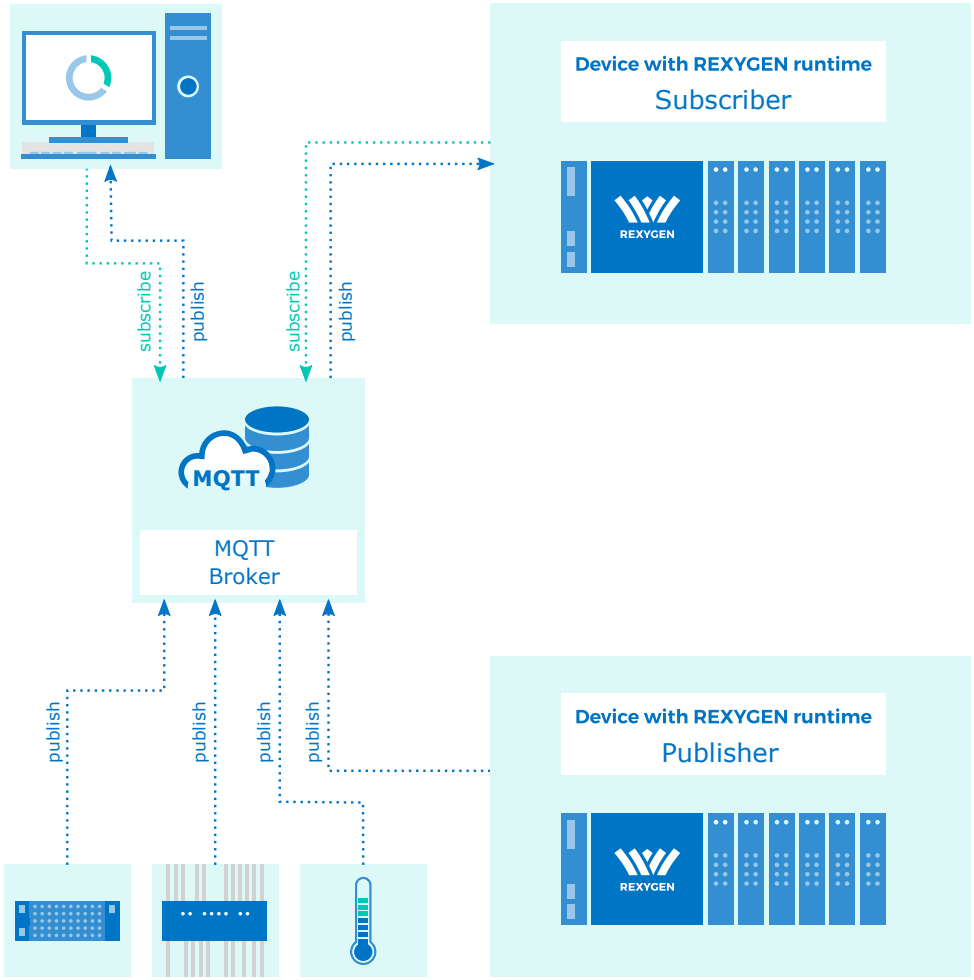

Figure 1.1: Example of an architecture based on MQTT protocol

1.2.2 Linux machines

If there is no RexCore runtime module installed on your target device, install it first using the Getting started guide of REXYGEN [2]. The installation includes all necessary drivers including `MQTTDrv`.

If you want to install `MQTTDrv` separately, it can be done from the command line of Linux using the command

```
sudo apt-get install rex-mqttdrvt
```

Chapter 2

Including the driver in the project

The driver is included in the project as soon as the driver is added to the project main file and the inputs and outputs are connected in the control algorithm(s).

2.1 Adding the MQTTDrv driver

The project main file with the MQTTDrv driver included is shown in Figure 2.1.
There is one block which must be added to the project to include the driver. A block of type IODRV renamed to MQTT and connected to the Drivers output of the main EXEC block. The name of this block (MQTT, see Fig. 2.1), is the prefix of all input and output signals provided by this driver. The three most important parameters are:

- module – name of the module linked to the driver, in this case MQTTDrv – the name is CASE SENSITIVE!
- classname – class of the driver, in this case MQTTDrv
- cfgname – name of the driver configuration file, e.g. mqtt_cfg.rio
- factor – multiple of the EXEC block's tick parameter defining the execution period of the driver

The above mentioned parameters of the IODRV function block are configured in REXYGEN Studioprogram. The configuration dialog is shown also in Fig. 2.1.

The Configure button opens the configuration dialog of the MQTTDrv driver, which is described in chapter 2.2.

2.2 Configuration dialog of the MQTTDrv driver

The configuration dialog can be activated from REXYGEN Studio by pressing the Configure button in the parameters dialog of the IODRV block (renamed to MQTT, see chapter 2.1).

The Connection section shown in Fig. 2.2 configures the connection to the external MQTT broker. The MQTT Examples shipped with the installation use publicly available

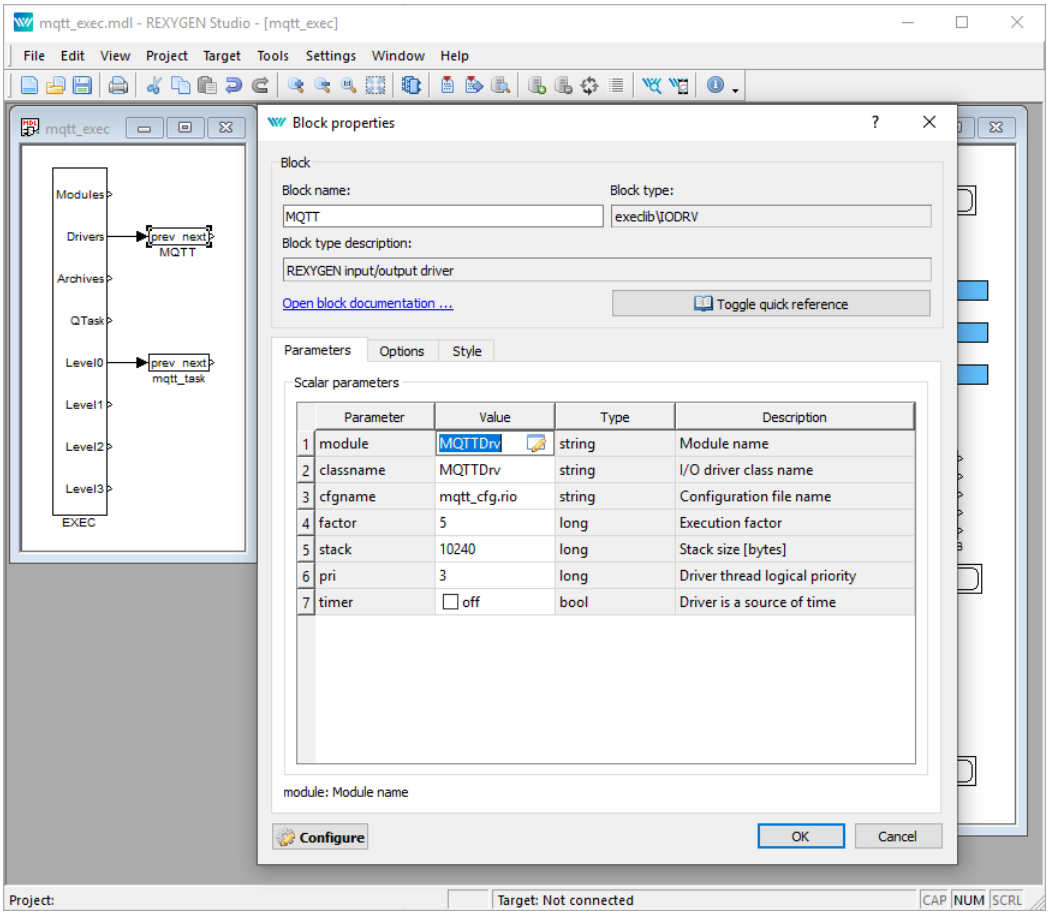

Figure 2.1: An example of project main file with the `MQTTDrv` driver included

brokers. Note that the *Client id* should be unique for every device connected to the broker.

Parameters *Username* and *Password* are not mandatory and are meant to be set only if broker requires authentication. Another approach to setting *Username* and *Password* is to use the corresponding driver inputs. (see chapter 3).

Parameter *Ping period* specifies how often should the driver send a ping message to the broker in order to maintain continuous communication flow. During the connection handshake between the client and the broker thr client must specify a *keep-alive* parameter. If the broker does not receive any message from the client for a period longer than this parameter than the broker should close the communication socket and consider the client disconnected. By default, three times the ping period is used. If the ping period is set to zero then no ping messages are sent.

The *Reconnection timeout* parameter sets the period after which the driver tries to reconnect to the broker if the connection is lost. Setting this parameter to 0 means that

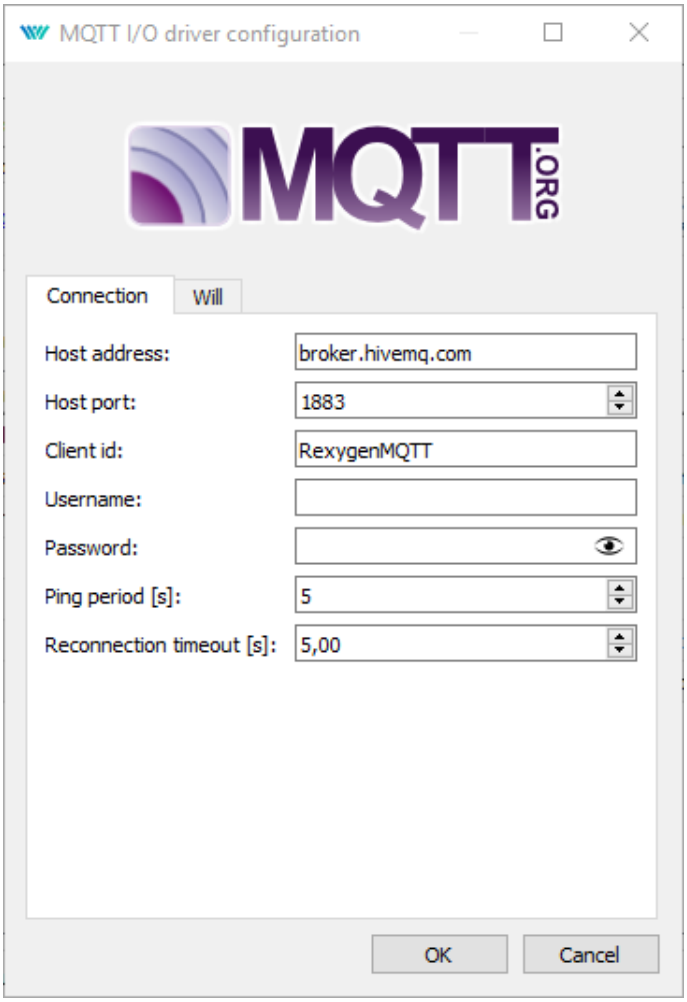

Figure 2.2: MQTT connection configuration dialog

the driver will try to reconnect as soon as possible.

The Will section shown in Fig. 2.3 configures the broker to send a will message on the specified MQTT topic when the connection to the client is lost. See the MQTT specification [1] for more details.

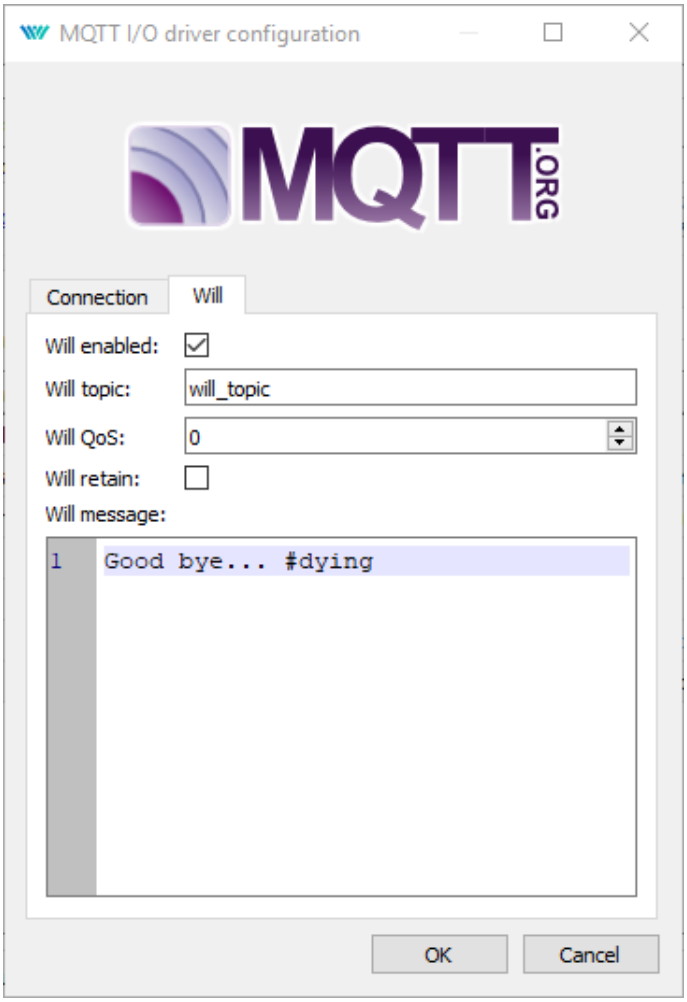

Figure 2.3: MQTT Will configuration dialog

Chapter 3

Connecting the inputs and outputs and using function blocks in the control algorithm

The inputs and outputs of the driver must be interconnected with the individual tasks (`.mdl` files). The individual tasks (`QTASK` or `TASK` blocks) are connected to the `QTask`, `Level0`,..., `Level3` outputs of the main `EXEC` block.

3.1 Direct input and output signals

The inputs and outputs of the `MQTTDrv` driver can be accessed as shown in Fig. 3.1.

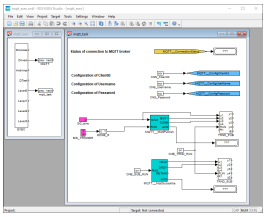

Figure 3.1: Example of input and output flags of the `MQTTDrv` driver

One block of the `From` type allowing the user to read connection status has the `Goto tag` set to `MQTT__ConnectionStatus`. The blocks of `Goto` type allowing the user to configure *ClientId*, *Username* and *Password* of the driver have the `Goto tag`) set to `MQTT__ConfigClientId`, `MQTT__ConfigUserName` and `MQTT__ConfigPassword`. The blocks always have the name of the driver block (recommended value is `MQTT`) as a prefix right at the beginning of the tag followed by two `_` characters (underscore).

All the input and output flags of the `MQTTDrv` driver are available in the example project 0407-00, which is part of the installation package. The most up-to-date examples are available at https://github.com/rexcontrols/REXexamples/archive/v2.50.zip.

3.2 Function blocks

The driver by itself maintains the connection to the MQTT broker and handles the communication over a socket. To publish a message over the MQTT protocol the `MqttPublish` block must be used. To subscribe to a topic and receive messages from the broker the `MqttSubscribe` block must be used. The function blocks of the `MQTTDrv` driver can be used as shown in Fig. 3.1. The blocks always have the name of the driver block (recommended value is `MQTT`) prefix right at the beginning of the tag followed by two `_` characters (underscore). To learn more about the `MqttPublish` and `MqttSubscribe` blocks see [3].

Chapter 4

Examples

To get started quickly the following examples can be used as a reference and you can modify them for your application.

- 0407-01 MQTT/MQTT Data Exchange – The example demonstrates communication between a Publisher and a Subscriber which are both implemented in REXYGEN.
- 0302-09 IoT Integrations/ThingSpeak MQTT API – The example demonstrates communication between REXYGEN and the ThingSpeak IoT cloud platform. REXYGEN can act as both the Publisher or the Subscriber.

Chapter 5

Troubleshooting

In the case that the diagnostic tools of the REXYGEN system (e.g. `Watch` mode in the REXYGEN Studio) report unexpected or incorrect values of inputs or outputs, it is desirable to test the functionality outside the REXYGEN system. There are many free programs that can be used to monitor MQTT communication such as mqtt-spy. Also double check the configuration – the most common problems include:

- Invalid broker connection setting
- Incorrect topic configuration

In the case that the given input or output works with other software tools and does not work in the REXYGEN system, report the problem to us, please. E-mail is preferred, reach us at `support@rexygen.com`. Please include the following information in your description to help us process your request as soon as possible:

- Identification of the REXYGEN system you are using. Simply export it to a file using the REXYGEN Studio (Target → Licensing... → Export).
- Short and accurate description of your problem.
- The configuration files of the REXYGEN system (`.mdl` and `.rio` files) reduced to the simplest case which still demonstrates the problematic behavior.

Bibliography

[1] OASIS. MQTT Version 3.1.1. http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html, 2014.

[2] REX Controls s.r.o.. *Getting started with REXYGEN*, 2024. →.

[3] REX Controls s.r.o.. *Function blocks of REXYGEN – reference manual*, 2024. →.