

MQTT-flex

NETIO M2M API protocols docs

Protocol version: MQTT-flex Version 1.0

Short summary

Standard **MQTT M2M API** is standard NETIO communication protocol, where NETIO device is publisher providing output status and subscriber allowing control of NETIO power outputs.

Extended version **MQTT-flex** is defined by text config file. In this config file can user define the MQTT topics and payloads structure.

MQTT-**flex** is MQTT with **flexibility** in configuration capabilities and usage (customizable MQTT)

- Flexibility in communication parameters for MQTT broker
- Flexibility of topics and payload definition
- User defined communication triggers (period / delta values)
- Easy configuration using structured config in JSON format entered on the web GUI.

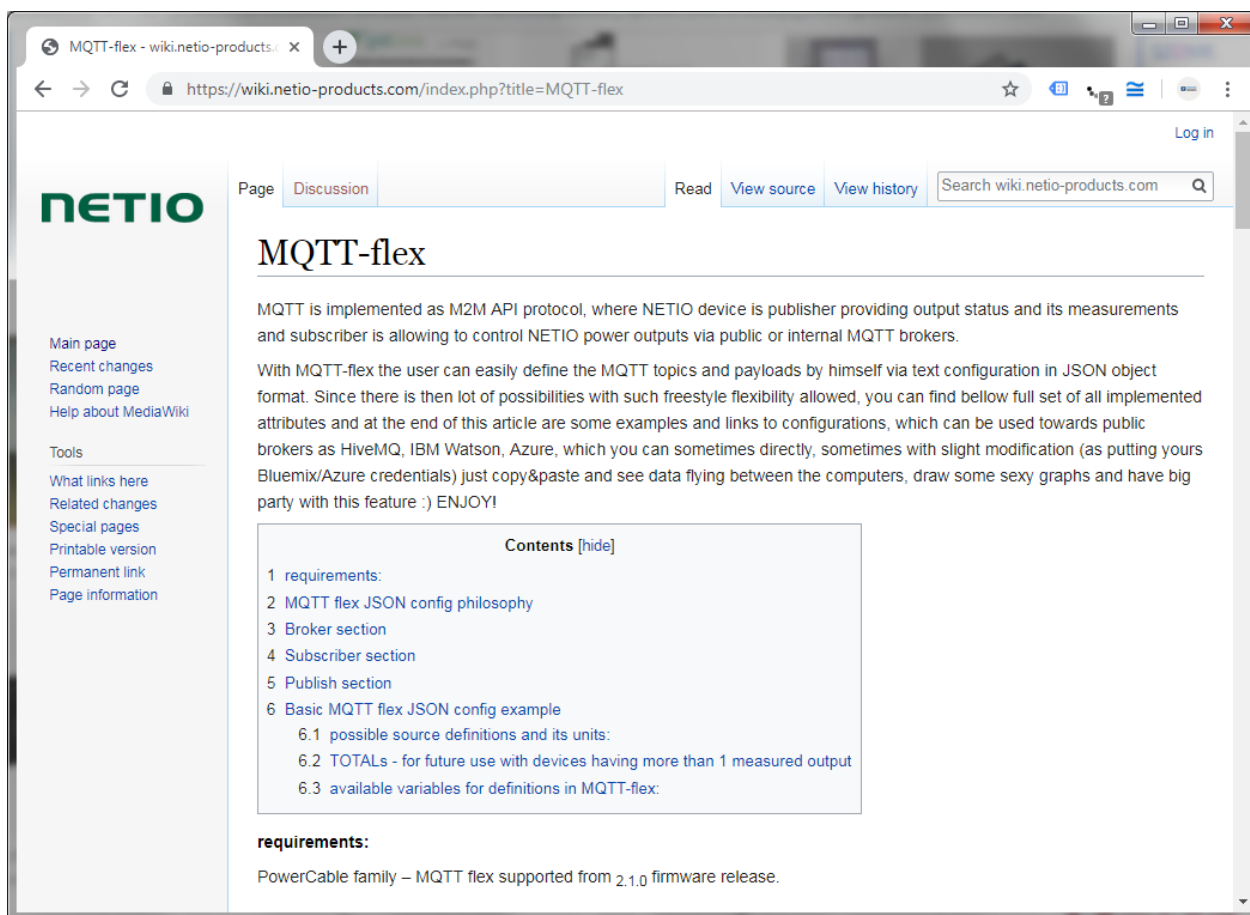
	NETIO MQTT-flex	NETIO MQTT
PowerCable MQTT	Yes	-
PowerPDU 4C	-	Yes
NETIO 4	-	Yes
NETIO 4All	-	Yes
NETIO MQTT fixed structure (JSON payload)	Yes*	Yes
User definable topics	Yes	-
User definable payloads	Yes	-
MQTT templates	Yes	-
Time period / triggered delta values for pushing data	Yes	-

* 3Q 2019

Supported devices and firmware

PowerCable firmware – 2.2.0 and later

NETIO Wiki = Details about MQTT-flex



The screenshot shows a web browser window displaying the NETIO Wiki page for MQTT-flex. The browser's address bar shows the URL: <https://wiki.netio-products.com/index.php?title=MQTT-flex>. The page features the NETIO logo on the left and a navigation menu with options like 'Page', 'Discussion', 'Read', 'View source', and 'View history'. The main content area is titled 'MQTT-flex' and contains the following text:

MQTT is implemented as M2M API protocol, where NETIO device is publisher providing output status and its measurements and subscriber is allowing to control NETIO power outputs via public or internal MQTT brokers.

With MQTT-flex the user can easily define the MQTT topics and payloads by himself via text configuration in JSON object format. Since there is then lot of possibilities with such freestyle flexibility allowed, you can find below full set of all implemented attributes and at the end of this article are some examples and links to configurations, which can be used towards public brokers as HiveMQ, IBM Watson, Azure, which you can sometimes directly, sometimes with slight modification (as putting yours Bluemix/Azure credentials) just copy&paste and see data flying between the computers, draw some sexy graphs and have big party with this feature :) ENJOY!

A 'Contents [hide]' box lists the following sections:

- 1 requirements:
- 2 MQTT flex JSON config philosophy
- 3 Broker section
- 4 Subscriber section
- 5 Publish section
- 6 Basic MQTT flex JSON config example
 - 6.1 possible source definitions and its units:
 - 6.2 TOTALs - for future use with devices having more than 1 measured output
 - 6.3 available variables for definitions in MQTT-flex:

Below the contents box, the text reads:

requirements:
PowerCable family – MQTT flex supported from 2.1.0 firmware release.

<https://wiki.netio-products.com/index.php?title=MQTT-flex>

General protocol info

MQTT is a machine-to-machine (M2M) / "Internet of Things" connectivity protocol. It was designed as an extremely lightweight publish/subscribe messaging transport. It is useful for connections with remote locations where a small code footprint is required and/or network bandwidth is at a premium.

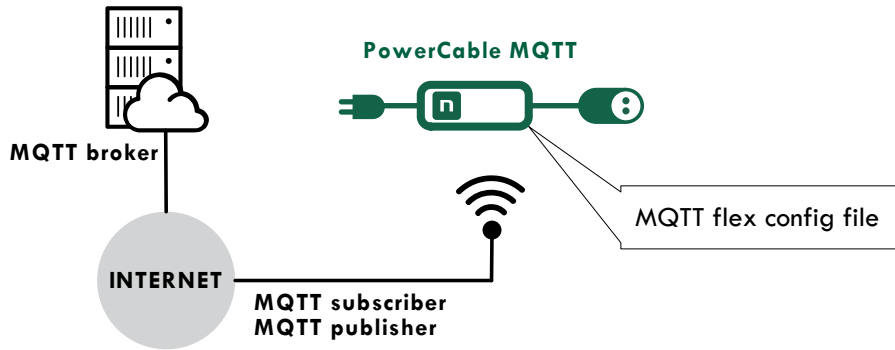
Sources:

<http://mqtt.org/>

<https://en.wikipedia.org/wiki/MQTT>

NOTE: *This document provides basic info about the M2M API protocol.
Other device functions are described in the product manual.*

Quick start with MQTT & NETIO



Simple configuration example

MQTT broker url: broker.hivemq.com
MQTT broker port: 1883
SSL: no
Username: freedom
Password: peace|LOVE|empathy4ALL

Subscribe topic for control of the output:

- netio/<DEVICE_NAME>/output/1/action with payload for output control : (0 – off, 1 – on, 2 – short off, 3 – short on, 4 – toggle, 5 – no change)

Publish topics for monitoring of the output state and load:

- netio/<DEVICE_NAME>/output/1/state with payload providing output 1 state value - published when the output state change
- netio/<DEVICE_NAME>/output/1/load with payload providing actual load of output 1 – published every 1111 seconds or when load change by 1W

Config example

```
{
  "config":{
    "broker":{
      "url":"broker.hivemq.com",
      "protocol":"mqtt",
      "port":1883,
      "ssl":false,
      "type":"generic",
      "username":"freedom",
      "password":"peace | LOVE | empathy4ALL"
    },
    "subscribe":[
      {
        "topic":"netio/${DEVICE_NAME}/output/1/action",
        "qos":0,
        "target":"OUTPUTS/1/ACTION",
        "action":"${payload}"
      }
    ],
    "publish":[
      {
        "topic":"netio/${DEVICE_NAME}/output/1/state",
        "qos":0,
        "retain":true,
        "payload":"${OUTPUTS/1/STATE}",
        "events":[
          {
            "type":"change",
            "source":"OUTPUTS/1/STATE"
          }
        ]
      },
      {
        "topic":"netio/${DEVICE_NAME}/output/1/load",
        "qos":0,
        "retain":false,
        "payload":"${OUTPUTS/1/LOAD}",
        "events":[
          {
            "type":"timer",
```

```

        "period":1111
    },
    {
        "type":"delta",
        "source":"OUTPUTS/1/LOAD",
        "delta":1
    }
]
}
}
}

```

PowerCable MQTT 1970-01-01 13:10:23 admin [Sign out](#)

PowerCable-6E

Outputs

M2M API Protocols

Users

Settings

Log

MQTT-flex

Netio Push

Enable MQTT-flex

MQTT-flex config:

```

1 {
2   "config":{
3     "broker":{
4       "url":"broker.hivemq.com",
5       "protocol":"mqtt",
6       "port":1883,
7       "ssl":false,
8       "type":"generic",
9       "username":"freedom",
10      "password":"peace|LOVE|empathy4ALL"
11     },
12   "subscribe":[
13     {
14       "topic":"netio/${DEVICE_NAME}/output/1/action",
15       "qos":0,
16       "target":"OUTPUTS/1/ACTION",
17       "action":"${payload}"
18     }
19   ],
20   "publish":{
21     {

```

Save Changes

[Product manual](#) [NETIO products a.s.](#) 2.1.3 - 1.23(1.23) - 108 (0c3f389)

Picture 1 – M2M API Protocols / MQTT-flex settings GUI with config example

General NETIO output functions

Output status – “read” function

- **0** – Power **OFF**
- **1** – Power **ON**

Output actions – “write” function

- **0** – Turn **OFF**
- **1** – Turn **ON**
- **2** – Short OFF delay (restart)
- **3** – Short ON delay
- **4** – **Toggle** (invert the state)
- **5** – No change

Short ON / OFF delay

This command switches a power output On / Off for a defined time. It is useful for example to power-cycle a server with a defined switch-off time, or to switch on a pump for a defined time.

This “short” delay is protected: the power output will remain in the defined state regardless of any other M2M requests received. During this time, the output state can only be changed by pressing the button on the NETIO device and this action cancel M2M short ON/OFF command for the particular output. Other requests to control the particular output are simply ignored and an ERROR logged with reason rejected in a device Log.

The short ON / OFF delay interval can be defined in the device web administration. It is specified in ms (milliseconds) and rounded up to hundreds of milliseconds (0,1s).

This interval can be also defined using some M2M API protocol commands. In that case, it is valid only for a single protocol session (the following short ON / Short OFF command). When the connection is closed or restarted, the interval is reset to the device default value (defined in the web administration for each output).

Security issues

Do not use default usernames and passwords! Keep your Ethernet and WiFi networks secured.

Power-Up outputs state

All outputs are Off during the power-up.

After this time, all outputs are set to the selected state:

- **Last Output state**

After a power outage, the NETIO device sets each power output to the last stored state of this one output.

- **Off**

After a power outage, the NETIO device keep each power output off.

- **On**

After a power outage, the NETIO device sets each power output to on.

Energy metering variables

Parameters for each power output:

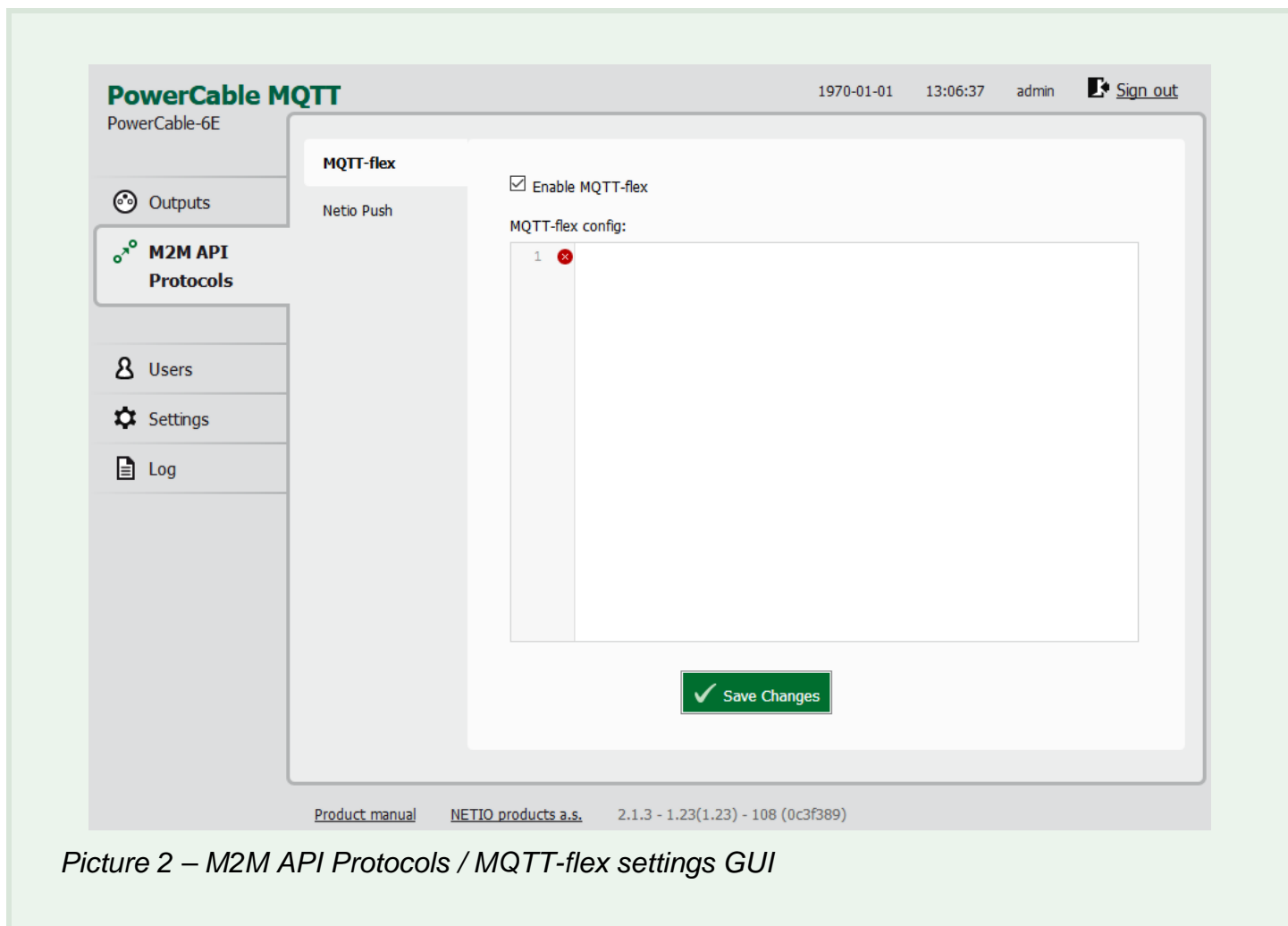
Variable	Unit	Description
Current	mA	Instantaneous current for the specific power output
PowerFactor	-	Instantaneous Power Factor for the specific power output
Load	W	Instantaneous load for the specific power output
Energy	Wh	Instantaneous Energy counter value for the specific power output

Parameters for the whole NETIO device:

Variable	Unit	Description
Voltage	V	Instantaneous voltage
Frequency	Hz	Instantaneous frequency
TotalCurrent	mA	Instantaneous total current through all power outputs
OverallPowerFactor	-	Instantaneous Power Factor – weighted average from all meters
TotalLoad	W	Total Load of all power outputs (device's own internal consumption is not included)
TotalEnergy	Wh	Instantaneous value of the Total Energy counter
EnergyStart	-	Date and time of the last reset of all energy counters

NETIO WEB configuration

M2M API protocols can be enabled and configured only over the web administration – select “M2M API Protocols” in the left-hand side menu and then select the “MQTT-flex” tab.



Picture 2 – M2M API Protocols / MQTT-flex settings GUI

- **Enable MQTT-flex** - Enable/disable M2M API protocol
- **MQTT-flex Config:** - input block for configuration

NETIO MQTT-flex protocol structure

Configuration is defined using JSON format and consist of three “sections”:

- Broker
- Subscribe
- Publish

Basic structure

```
{
  "config":{
    "broker":{
      <broker definitions>
    },
    "subscribe":[
      {
        <subscribe topic 1 definitions>
      }
    ],
    "publish":[
      {
        <publish topic 1 definitions>
      },
      {
        <publish topic 2 definitions>
      }
    ]
  }
}
```

Broker section

Parameters:

Item	Description	Example/values
url	MQTT broker URL	broker.hivemq.com
Protocol	Protocol used for communication with broker	[mqtt]
Port	MQTT broker port	1883
Ssl	SSL security selector	[false true]
Type	Communication/setup type option	[generic]
username	Credentials for MQTT broker - username	freedom

password	Credentials for MQTT broker - password	peace LOVE empathy4ALL
clientid	[option] MQTT clientid. Max. 32 characters. Variables \${MAC} or \${DEVICE_NAME} can be used.	myUniqueID1234
keepalive	[option] MQTT keep alive period in seconds	90

Example

```
"broker": {  
  "url": "broker.hivemq.com",  
  "protocol": "mqtt",  
  "port": 1883,  
  "ssl": false,  
  "type": "generic",  
  "username": "freedom",  
  "password": "peace|LOVE|empathy4ALL"  
}
```

Subscribe and Publish sections

There are wide options for subscribe and publish sections and its possibilities expand over the time.

You will find details and examples at our online resource center:

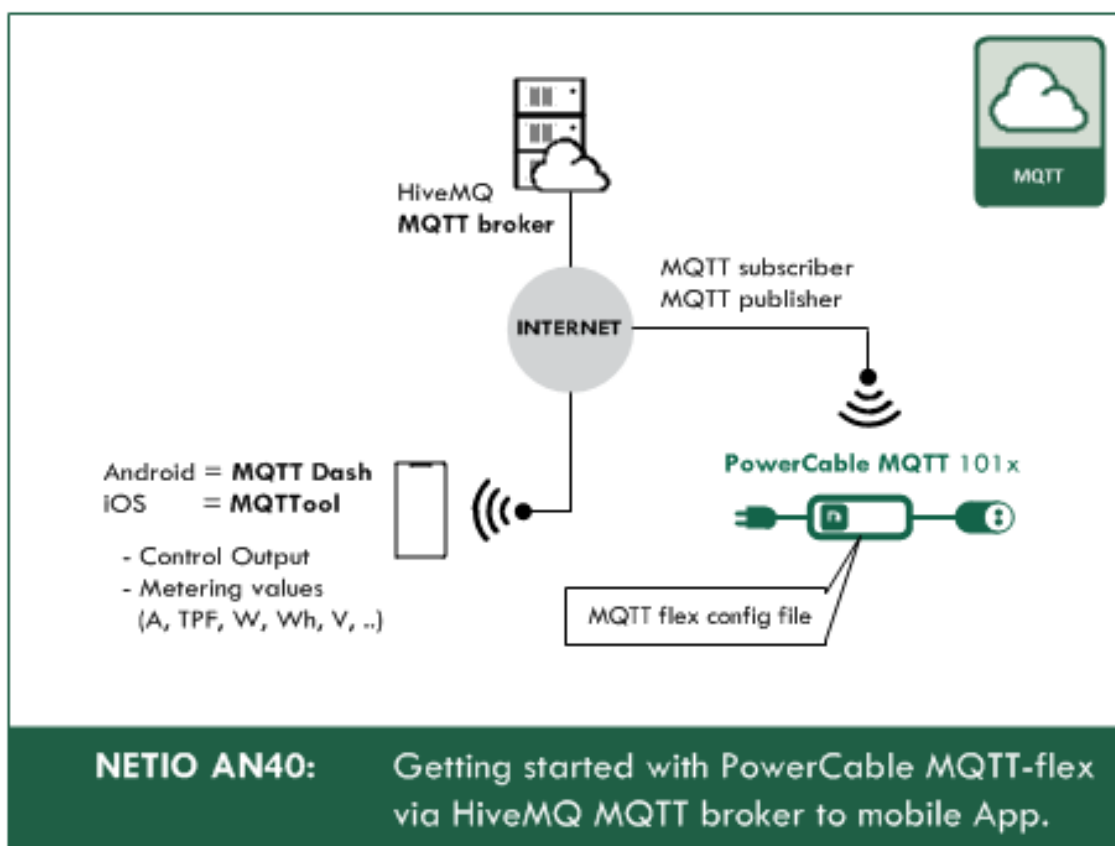
<https://wiki.netio-products.com/index.php?title=MQTT-flex>

Notes

- The “Uptime” value is in seconds [s]

NETIO AN (Application Note)

AN40 PowerCable MQTT – getting started – simple application - HiveMQ



MQTT-flex, supported by PowerCable MQTT, is a protocol for cloud applications. Flex is a method of configuring the standard MQTT with a user-defined topic and payload structure. The customer does not need to adapt to one particular MQTT structure. AN40 demonstrates how to configure MQTT-flex in PowerCable MQTT and connect the Wi-Fi smart socket device to the public HiveMQ broker. A mobile app is used to display the data and control the power output over the MQTT protocol.

>> Read the AN40 on www.netio-products.com

Document history

Document Revision	Publication Date	Description
1.0	11.9.2019	The first version, Initial release