G13

User Manual



G13

User Manual

Document ID: 2CMC489001M0201

Revision: A

2013-04-25

Disclaimer

The information in this document is subject to change without notice and should not be construed as a commitment by ABB AB. ABB AB assumes no responsibility for any errors that may appear in this document.

In no event shall ABB AB be liable for direct, indirect, special, incidental or consequential damages of any nature or kind arising from the use of this document, nor shall ABB AB be liable for incidental or consequential damages arising from use of any software or hardware described in this document.

Copyrights

This document and parts thereof must not be reproduced or copied without written permission from ABB AB, and the contents thereof must not be imparted to a third party nor used for any unauthorized purpose.

The software or hardware described in this document is furnished under a license and may be used, copied, or disclosed only in accordance with the terms of such license.

© Copyright 2013 ABB AB. All rights reserved.

Trademarks

ABB AB is a registered trademark of the ABB Group. All other brand or product names mentioned in this document may be trademarks or registered trademarks of their respective holders.

Contact

ABB AB

P.O. BOX 1005

SE-611 29 NYKÖPING

SWEDEN

Tel: +46 155 295000

Fax: +46 155 288110

Table of Content

1 Product Overview 5
1.1 The Parts of the Gateway 6
1.2 The Parts of the Web Interface 8
2 Installation 9
2.1 Installing the Gateway 10
2.1.1 Wiring Diagrams 11
3 Technical Data 12
3.1 Technical Specifications 13
3.2 Physical Dimensions 15
4 User Interface and Setup 16
4.1 User interface 17
4.2 Gateway Settings 21
4.3 Firmware update 25
4.4 Device Registration 27
4.5 Connect a meter via the Energy Meter Gateway interface 28
4.5.1 Scan meter 29
4.5.2 Add scanned meter 31
4.5.3 User Management 31
5 Meter Settings 35
5.1 Setting and Configuration 36
5.1.1 General Settings 36
5.1.2 Previous Value Configuration 39
5.1.3 Load Profile Configuration 40
5.1.4 Demand Configuration 41
5.1.5 Tariff Configuration 42
5.1.6 Input/ Output Configuration 44
5.1.7 Pulse Output Configuration 45
5.1.8 Alarm Configuration 47
6 JSON Communication 48
6.1 About JSON 49
6.2 Table of Resources 52
6.3 Resources 55
6.3.1 GET /about 55
6.3.2 POST /login 56
6.3.3 POST /logout 57
6.3.4 GET /configuration 58
6.3.5 POST /configuration/ip 60
6.3.6 POST /configuration/eqbus 62
6.3.7 POST /configuration/mbus-wired 64
6.3.8 POST /configuration/mbus-ir 66
6.3.9 GET /gateway 68
6.3.10 POST /gateway 69
6.3.11 GET /gateway/datetime 71
6.3.12 POST /gateway/datetime 72
6.3.13 GET /gateway/events 74
6.3.14 DELETE /gateway/events 77
6.3.15 POST /gateway/reboot 77
6.3.16 POST /gateway/execute 79
6.3.17 GET /users 81

6.3.18 POST /users/<user> 82
6.3.19 PUT /users 83
6.3.20 DELETE /users/<user> 84
6.3.21 GET /users/<user>/bindings 85
6.3.22 PUT /users/<user>/bindings 86
6.3.23 DELETE /users/<user>/bindings 87
6.3.24 GET /meters/firmwareupdatestatus 87
6.3.25 POST /firmware 89
6.3.26 PUT /firmware 90
6.3.27 PUT /meters/firmware 91
6.3.28 GET /parametermapping 92
6.3.29 GET /storablequantities 98
6.3.30 GET /meters/<serial>/associationobjects 100
6.3.31 GET /meters 108
6.3.32 POST /meters/<serial> 111
6.3.33 PUT /meters 112
6.3.34 DELETE /meters/<serial> 114
6.3.35 GET /meters/scanned 116
6.3.36 POST /meters/scanned/<token> 118
6.3.37 DELETE /meters/scanned 121
6.3.38 GET /permittedmeters 123
6.3.39 PUT /permittedmeters 124
6.3.40 DELETE /permittedmeters 126
6.3.41 GET /meters/<serial>/datetime 128
6.3.42 POST /meters/<serial>/datetime 130
6.3.43 GET /meters/<serial>/info 132
6.3.44 GET /meters/<serial>/energy/conversionfactor 134
6.3.45 POST /meters/<serial>/energy/conversionfactor 135
6.3.46 GET /meters/<serial>/hardwareversion 137
6.3.47 GET /meters/<serial>/mbusinfo 138
6.3.48 POST /meters/<serial>/transformersettings 141
6.3.49 GET /meters/<serial>/status 142
6.3.50 GET /meters/<serial>/statusflags 143
6.3.51 GET /meters/<serial>/events/<datetime>/<count> 146
6.3.52 GET /meters/<serial>/alarms/configuration 153
6.3.53 POST /meters/<serial>/alarms/configuration 156
6.3.54 GET /meters/<serial>/energy/<type> 158
6.3.55 GET /meters/<serial>/energy/<type>/<mode> 161
6.3.56 GET /meters/<serial>/energy/resettable 163
6.3.57 POST /meters/<serial>/energy/resettable 165
6.3.58 GET /meters/<serial>/energy/resetcounter 166
6.3.59 GET /meters/<serial>/powers 167
6.3.60 GET /meters/<serial>/instrument 169
6.3.61 GET /meters/<serial>/harmonics/voltage 172
6.3.62 GET /meters/<serial>/harmonics/current 177
6.3.63 GET /meters/<serial>/io 180
6.3.64 GET /meters/<serial>/io/configuration 182
6.3.65 GET /meters/<serial>/io/pulse 184
6.3.66 POST /meters/<serial>/io 186
6.3.67 POST /meters/<serial>/io/configuration 189
6.3.68 POST /meters/<serial>/io/pulse 191
6.3.69 GET /meters/<serial>/previousvalues/<fromdate>/<count|todate> 194
6.3.70 GET /meters/<serial>/previousvalues/configuration 196

6.3.71 POST /meters/<serial>/previousvalues/configuration 197
6.3.72 DELETE /meters/<serial>/previousvalues 199
6.3.73 GET /meters/<serial>/loadprofiles/<channel>/<fromdate>/<count|todate> ... 200
6.3.74 GET /meters/<serial>/loadprofiles/configuration 202
6.3.75 POST /meters/<serial>/loadprofiles/configuration 204
6.3.76 DELETE /meters/<serial>/loadprofiles 208
6.3.77 GET /meters/<serial>/demand/<fromdate><count|todate> 209
6.3.78 GET /meters/<serial>/demand/configuration 213
6.3.79 POST /meters/ <serial>/demand/ configuration 216
6.3.80 DELETE /meters/<serial>/demand 218
6.3.81 GET /meters/ <serial> /tariff 220
6.3.82 POST /meters <serial> /tariff 222
6.3.83 GET /meters/<serial>/tariff/dayprofiles 224
6.3.84 POST /meters/<serial>/tariff/dayprofiles 228
6.3.85 GET /meters/<serial>/tariff/weekprofiles 230
6.3.86 POST /meters <serial> /tariff /weekprofiles 232
6.3.87 GET /meters/<serial>/tariff/seasonprofiles 234
6.3.88 POST /meters/<serial>/tariff/seasonprofiles 236
6.3.89 GET /meters/<serial>/tariff/specialdayprofiles 238
6.3.90 POST /meters/ <serial>/tariff/specialdayprofiles 240
6.3.91 GET /lasterror/<id> 242
6.3.92 Annexure1 243

Table of Content

Chapter 1: Product Overview

Overview

This chapter describes the parts of the gateway and of the web interface. It also explains the functions of the push buttons on the gateway and the behavior of the status LEDs.

In this chapter

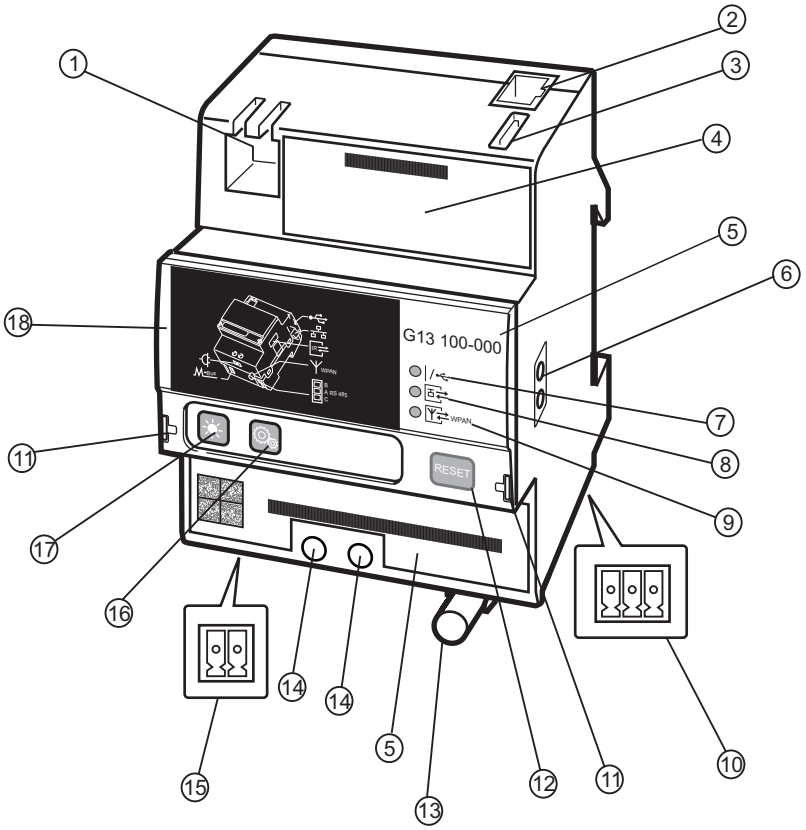
The following topics are covered in this chapter:

1.1 The Parts of the Gateway 6
1.2 The Parts of the Web Interface 8

1.1 The Parts of the Gateway

Illustration

The parts of the Gateway are shown in the illustration below:



Parts description

The following table describes the parts of the Gateway:

Item	Part	Comments
1		Not used
2	Ethernet socket	
3	USB port	For future usage
4	Product data	MAC/EUI address
5	Product data	
6	IR Interface	
7	LED A	Power status
8	LED B	Wired meter connection status
9	LED C	Wireless meter connection status
10	Terminal block	RS-485 connection
11	Sealing point	
12	Reset	
13	Antenna	

Item	Part	Comments
14	Terminal block	Voltage connection
15	Terminal block	M-Bus connection
16	USB configuration button	For future usage
17	LED suppress button	LED intense control
18	Sealable cover	

Push buttons

The following table describes the push buttons of the gateway and their functions:

Button	Description
	USB configuration button (for future usage)
	LED Suppress button. Press the button to reduce intensity on the LEDs.
	Reset button. For Factory Settings, press the button for ≥ 10 sec. **Note –** When the reset button is engaged the reset indication will override all USB status indication on LED A.

Status LEDs

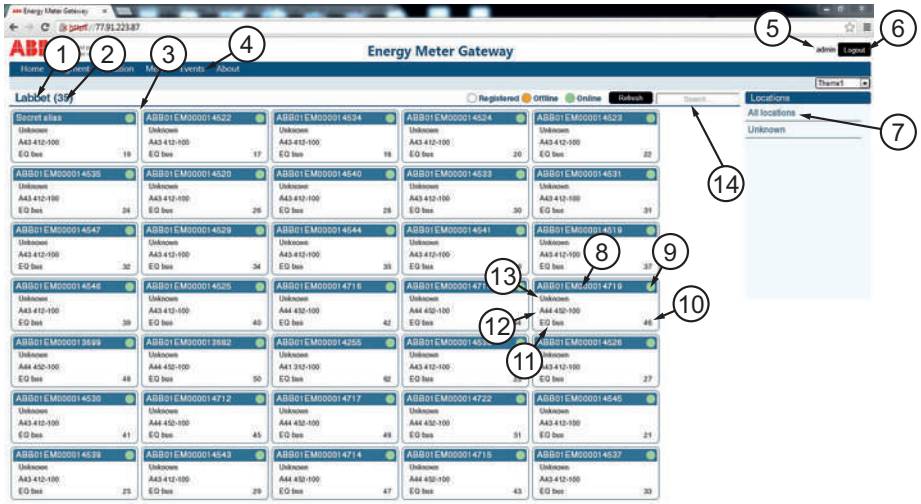
The following tables describe the behaviour of the status LEDs:

LED A - Power status	Behaviour
In operation	Steady green
Power OFF	OFF
LED A - USB status	
USB Initialize	Flashes green twice
USB Idle	Steady green
Data Transfer	Continuous flashing green
LED A - Reset status	
Soft Reset	Slow flashing green
Factory Default Reset	Fast flashing green
LED B - Wired meter connection status	
Communication Idle	Steady green
Data Transfer	Continuous flashing green
Connection error	OFF
LED C - Wireless meter connection status	
Communication Idle	Steady green
Data Transfer	Continuous flashing green

1.2 The Parts of the Web Interface

Illustration

The parts of the web interface are shown in the illustration below:



Parts description

The following table describes the parts of the web interface:

Item	Part	Comments
1	Location	The presently selected location (Group of Meters)
2	Meters in location	Number of meters associated with the present location.
3	Meter list	A list of each meter in the selected location
4	Menu bar	The main menu bar used for navigation
5	Username	The username of the current active user
6	Logout	Button to log out of the gateway
7	Location list	List of selectable locations (group of meters)
8	Device name	Device ID, name or alias of the meter
9	Com. status	Status of the device (Registered, Online or Offline)
10	Address	Communication address
11	Protocol	Communication protocol
12	Type designation	Type designation of the meter, if the meter supports it
13	Location	The location the meter belongs to
14	Search field	A search field to search for specific meter

Chapter 2: Installation

Overview

This chapter describes how to mount and connect the G13 gateway to an electricity network.

In this chapter

The following topics are covered in this chapter:

2.1 Installing the Gateway 10
2.1.1 Wiring Diagrams 11

2.1 Installing the Gateway



Warning – Electrical equipment should only be installed, accessed, serviced and maintained by qualified electrical personnel.

Working with high voltage is potentially lethal. Persons subjected to high voltage may suffer cardiac arrest, burn injuries, or other severe injuries. To avoid such injuries, make sure to disconnect the power supply before you start the installation.



Warning – For safety reasons it is recommended that the equipment is installed in a way that makes it impossible to reach or touch the terminal blocks by accident.

The best way to make a safe installation is to install the unit in an enclosure. Access to the equipment should further be limited through use of lock and key controlled by qualified electrical personnel.



Warning – The Gateway must always be protected by a fuse on the incoming side.

In order to allow for maintenance of the Gateway a readily accessible disconnect device should be incorporated external to the equipment.

Installation requirements

Products with built-in wireless communication should not be installed closer than 0.2 m from people.

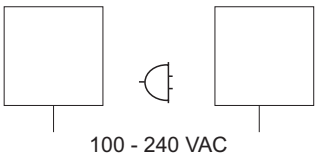
Install the meter

Follow the steps in the table below to install the Gateway:

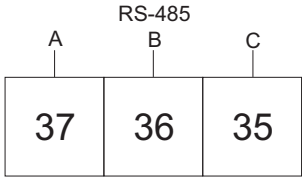
Step	Action
1	Turn off the main power.
2	Place the Gateway on the DIN rail on the left side of the electrical meter and make sure it snaps onto it. ! **Caution –** If IR communication will be used between Gateway and electricity meter, then a distance of more than 2 mm between the gateway and the meter may cause the IR communication to fail.
3	Strip off 6 mm of the insulation on the RS-485 cables. Connect the cables according to the marking on the product label and tighten the screws (0.25 Nm).
4	Strip off 6 mm of the insulation on the M-Bus cables. Connect the cables according to the marking on the product label and tighten the screws (0.25 Nm).
5	Strip off 6 mm of the insulation of the cable to the voltage connection.
6	Connect the cables according to the marking on the product label and tighten the screws (0.5 Nm).
7	Install the circuit protection (max. 50 A).
8	Turn on the power.
9	Verify that LED A shines steady green.

2.1.1 Wiring Diagrams

Voltage

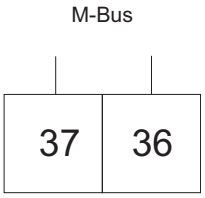


RS-485



Note – For RS-485, signal A should be wired to A on the meter and equivalent for signals B and C.

M-Bus



Note – Wiring of M-Bus is polarity independent

Chapter 3: Technical Data

Overview

This chapter contains the technical specifications and the physical dimensions of the product.

In this chapter

The following topics are covered in this chapter:

3.1 Technical Specifications 13
3.2 Physical Dimensions 15

3.1 Technical Specifications

Specifications for G13 Ethernet gateway

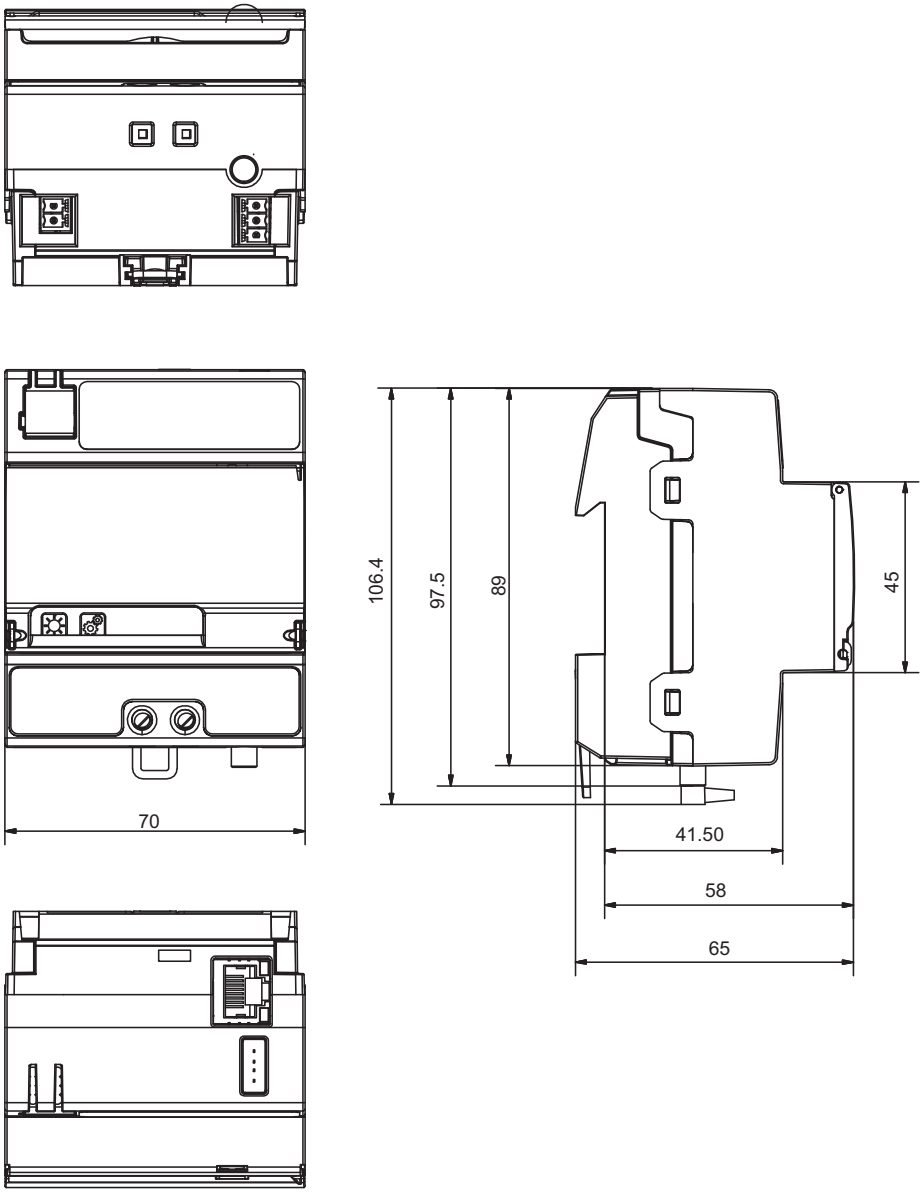
Power supply	
Voltage	100 - 240 V AC (-20% - +15%)
Fuse	0.5 - 80 A
Frequency	50 or 60 Hz ± 5%
Power consumption in standby	10 W
Power consumption in operation	Max. 20 W
Terminal wire area	0.75 - 1.5 mm^2
Recommended tightening torque	0.5 Nm
RS-485	
Baudrate	1200-500 000 bps
Terminal wire area solid	0.05 - 1.5 mm^2
Terminal wire area stranded	0.05 - 1 mm^2
Recommended tightening torque	0.25 Nm
M-Bus	
M-Bus master	Supports up to 16 unit loads.
Protocol	M-Bus
Baudrate	300 - 38400 bps
Terminal wire area solid	0.05 - 1.5 mm^2
Terminal wire area stranded	0.05 - 1 mm^2
Recommended tightening torque	0.25 Nm
Mechanical	
Enclosure material	Polycarbonate
Height	106 mm
Width	70 mm
Depth	65 mm
Product weight	190 g
Environmental	
Operating temperature range	-25°C to +70°C
Storage temperature range	-40°C to +85°C
Humidity	75% yearly average, 95% on 30 days/year
Resistance to fire and heat	650°C (IEC 60695-2-1)
Mechanical environment	Class M1 in accordance with the Measuring Instrument Directive (MID), (2004/22/EC).
Electromagnetic environment	Class E2 in accordance with the Measuring Instrument Directive (MID), (2004/22/EC).
Standards	
LVD	IEC/EN 60950-1
EMC	IEC/EN 61000-6-3, IEC/EN 61000-6-2

Protection class	Class II (double isolation)
IP Class	IP 20
Default IP address	https://192.168.1.12

3.2 Physical Dimensions

G13

The following drawing shows the physical dimensions of the G13 gateway in mm:



Chapter 4: User Interface and Setup

Overview

This chapter overviews the web interface menus for the gateway. It also describes how to setup a gateway and how to add a meter.

In this chapter

The following topics are covered in this chapter:

4.1 User interface 17
4.2 Gateway Settings 21
4.3 Firmware update 25
4.4 Device Registration 27
4.5 Connect a meter via the Energy Meter Gateway interface 28
4.5.1 Scan meter 29
4.5.2 Add scanned meter 31
4.5.3 User Management 31

4.1 User interface

General

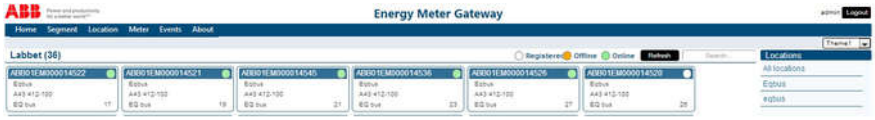
This chapter contains an overview of the different menus in the user interface and how to access the menus, the following menus can be accessed from the menu bar:

- **Home**
- **Segment**
- **Location**
- **Meter**
- **Events**
- **About**

Changes in the web interface are always executed to the meter by pressing the **Send** button. The changes are confirmed with a small rectangular green or blue pop-up window in the right lower corner. If a red rectangle is represented, then something is not correct or communication has failed. From the web interface it is possible to configure meters.

Home Menu

The **Home** menu displays the connected meters.



To access the **Home** menu:

1. Click **Home** in the menu bar.



Segment Menu

The **Segment** menu shows the following options:

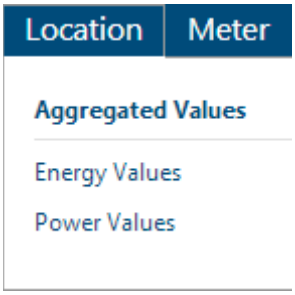


To access the **Segment** menu:

1. Click the **Segment** button in the menu bar.

Location menu

The **Location** menu shows the following options:

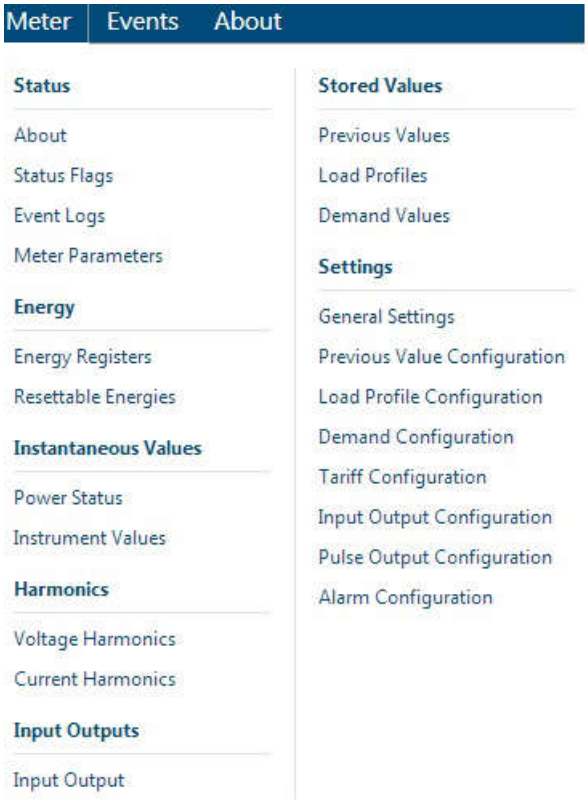


To access the **Location** menu:

1. Click **Location** in the menu bar.

Meter menu

The **Meter** menu shows the following options (Depending on functionality of the Meter):



To access the **Meter** menu:

1. Click **Meter** in the menu bar.

Events menu

The **Events** menu shows the Gateway events:



To access the **Events** menu:

1. Click **Events** in the menu bar.

To reset the Gateway Events:

1. Click **Reset**.

About menu



The **About** menu shows the following information about the gateway:

- Firmware Version
- Serial No
- JSON API Version
- COSEM Version
- Logical Device Name
- Release Date
- Licenses
- RAM Usage

To access the **About** menu:

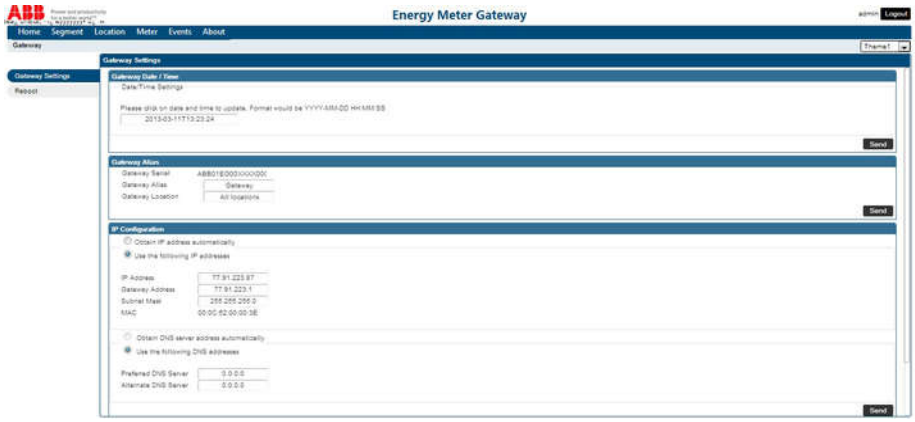
1. Click **About** in the menu bar.

4.2 Gateway Settings

Gateway settings

The basic settings for the Gateway consists of the following:

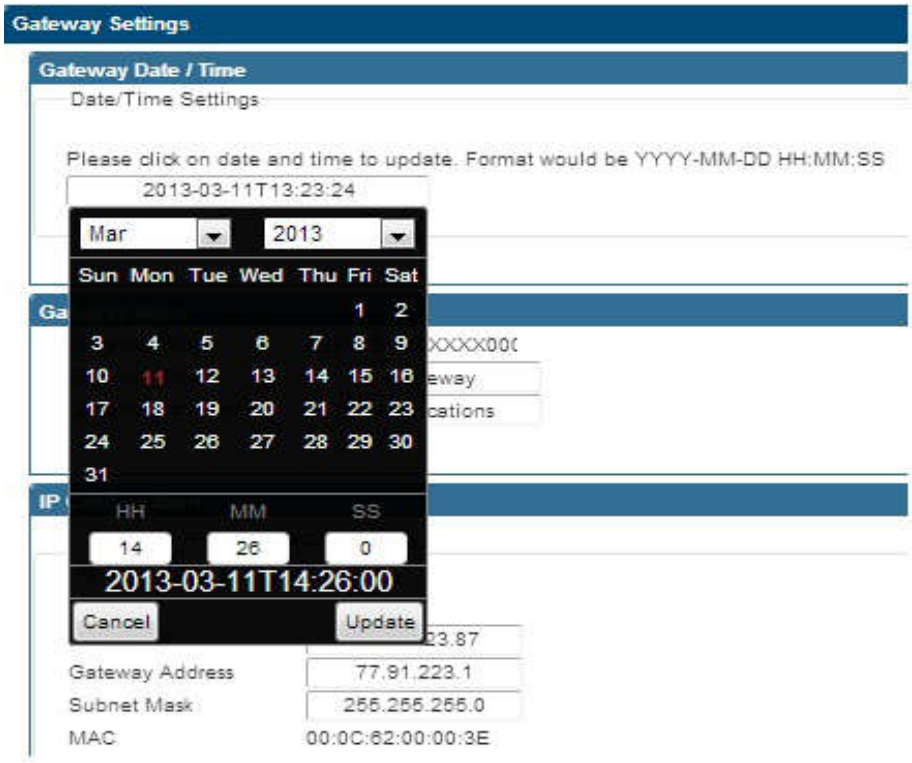
- **Gateway Date / Time**
- **Gateway Alias**
- **IP Configuration**
- **RS-485 Configuration**



Note – Only Gateway admin can change Gateway Settings.

Date/Time Settings

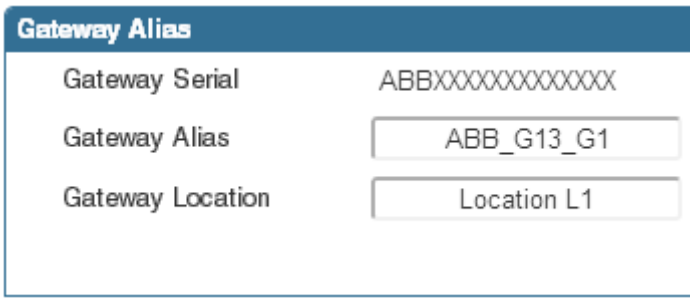
The **Date/Time Settings** collects the time from the computer clock.



To set **Date/Time**:

1. Click on date/time in the field.
2. Set date and time.
3. Click **Update**.
4. Press **Send**.

Gateway Alias and Location

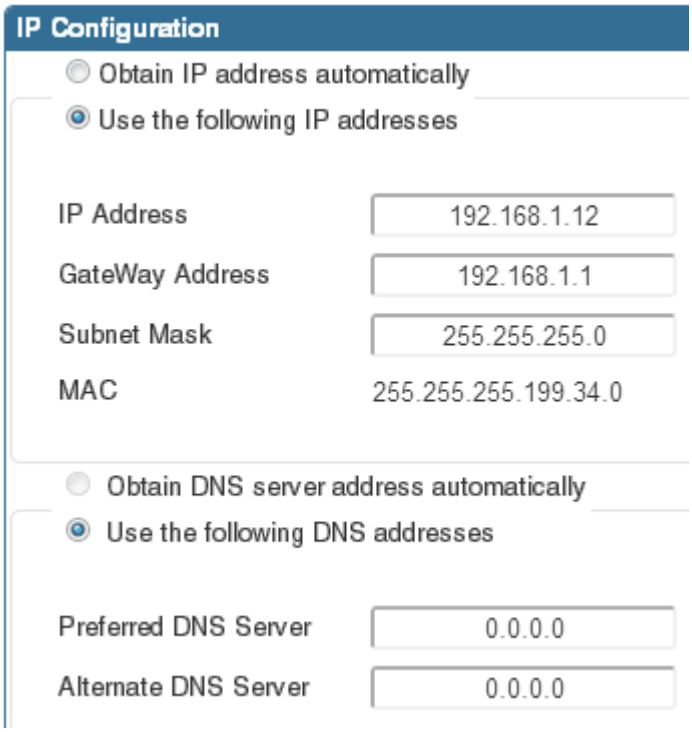


To set the **Gateway Alias** and **Gateway Location**:

1. Click in the **Gateway Alias** field and set the name.
2. Click on the **Gateway Location** field and set the name.
3. Click **Send**.

IP Configuration

IP addresses can be set manually or automatically.



To set IP address automatically:

1. Check radio button **Obtain IP address automatically**.
2. Click **Send**.

To set IP address manually:

1. Check the radio button **Use the following IP addresses**.
2. Click the **IP address** field.
3. Set the IP address.
4. Click **Send**.

To obtain DNS server address automatically.

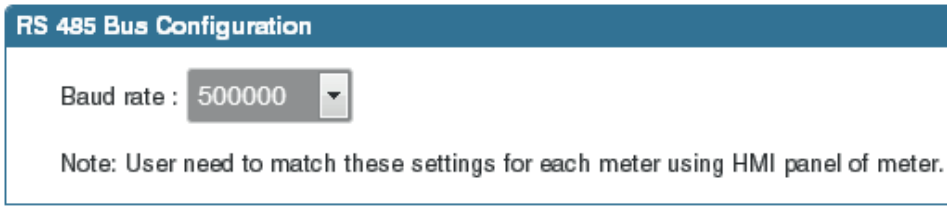
1. Check the radio button **Obtain DNS server address automatically**.
2. Click **Send**.

To set DNS server addresses:

1. Check the radio button **Use the following DNS addresses**.
2. Click the **IP address** field.
3. Set the IP address.
4. Click **Send**.

RS-485 Bus Configuration

The baudrate is set to 500000 by default for RS-485 Bus.



To set the baudrate:

1. Select **baudrate**.
2. Click **Send**.

4.3 Firmware update

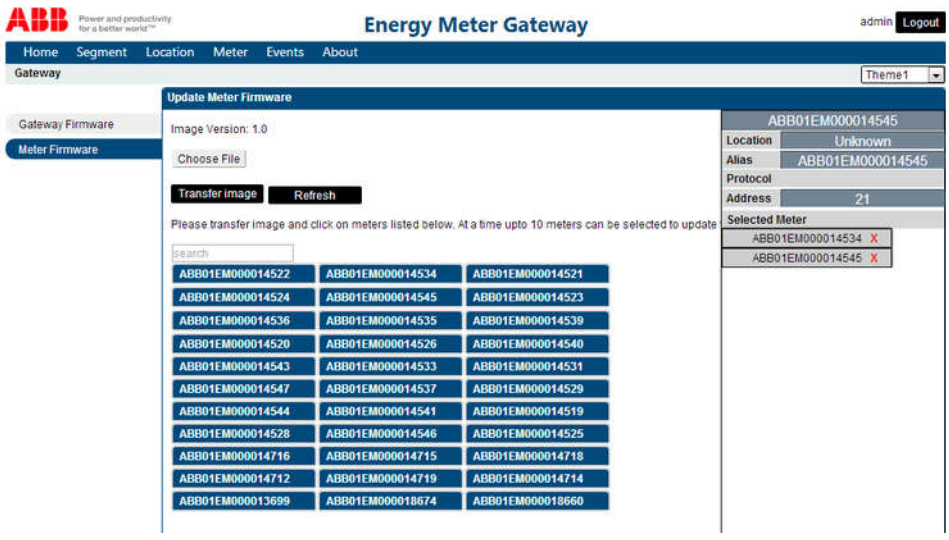
Gateway firmware



To update Gateway firmware:

1. Click **Segment** in the menu bar.
2. Click **Gateway firmware.** The current firmware version is shown in the upper part of the Gateway firmware square.
3. Click on **Choose file** and browse for preferred firmware version.
4. Click on **Transfer image.** The firmware is now sent to the gateway.
5. Click on **Update firmware.** The Gateway firmware will now be updated.

Meter firmware



To update Meter firmware:

1. Click **Segment** in the menu bar.
2. Click **Meter firmware.** The current firmware version is shown in the upper part of the Meter firmware square.
3. Choose meters to update by clicking on them. They will appear in the list on the right side.
4. Click on **Choose file** and browse for preferred firmware version.
5. Click on **Transfer image.** The firmware is now sent to the gateway.
6. Click on **Update firmware.** The Meter firmware will now be updated on the chosen meters.



Note – Maximum number of ten meters can be updated at the same time.

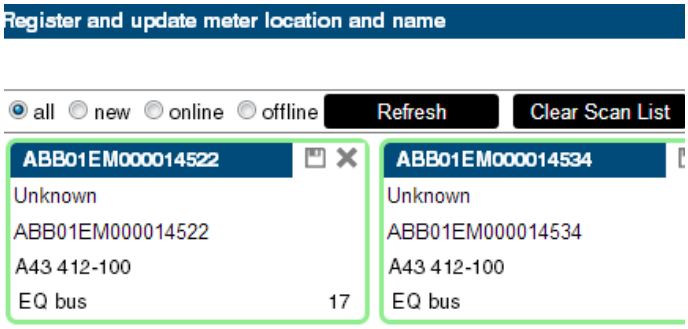
4.4 Device Registration

General

A meter connected to the gateway can be uniquely named in the **Device Registration** from **Segment** menu.



Setting a name for the meter



To set the name of the meter:

1. Click **Segment** in the menu bar.
2. Click **Device Registration**.
3. Click **Refresh** to update available meters.
4. Click on the field below ABB, named **Unknown** in the image above. Set location name of the meter.
5. Click in the field below, named **ABB01EM000014522** in the image above. Set device name.
6. Click on the **Save** icon (The floppy disk in the upper right of the meter square.).
7. Click **Send**.

To delete a meter, click on the **Delete** icon (The X mark in the upper right corner of the meter square.).

4.5 Connect a meter via the Energy Meter Gateway interface

General

It is always necessary to scan a meter to connect it to the gateway. The following chapters describe how to connect a meter to the gateway.

Login

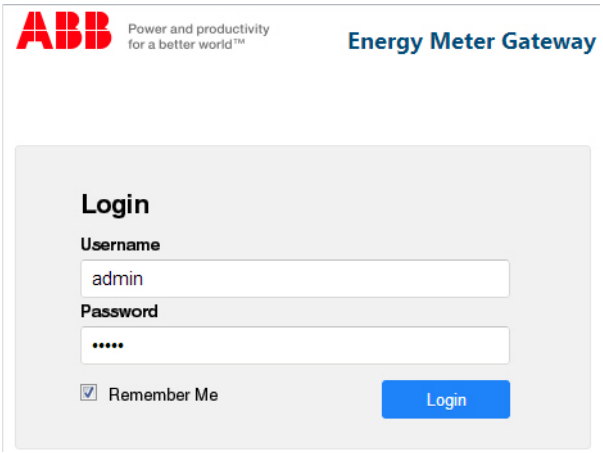
The following image shows the login to **Energy Meter Gateway** on https://192.168.1.12.

To log on to the Gateway:

1. **Username**: admin
2. **Password**: admin
3. Click **Login**.

Gateway will automatically prompt for change of password at first login.

Internet browser will warn for unsafe security since the Gateway certificate is self-issued by ABB.



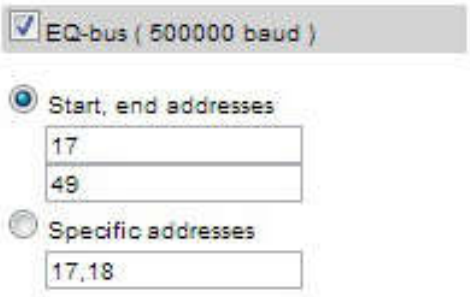
4.5.1 Scan meter

The menu appears the first time a user logs in. To configure a meter the meter has to be scanned and connected to the gateway via correct bus.

The menu is found under the **Segment** menu. Select **Device scan**.



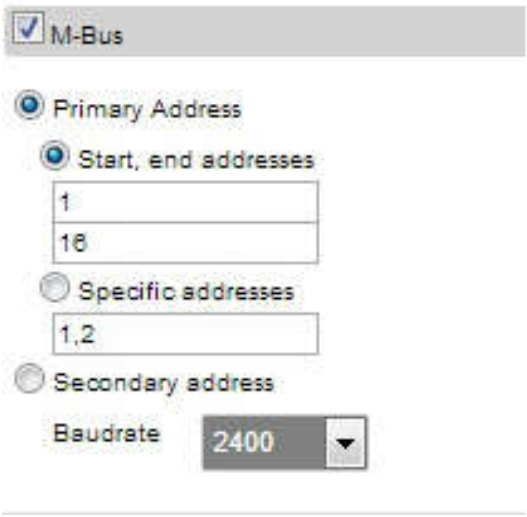
Scan new meter with EQ bus



To scan a meter with EQ bus.

1. Check the **EQ bus** box.
2. Check radio button **Start, end addresses** or **Specific addresses**
3. Set specific address/addresses.
4. Press **Scan**.

Scan new meter with M-Bus



To scan and add a meter via M-Bus.

1. Check the **M-Bus** box.
2. Check **Primary address** and check **Start, end addresses** or **Specific addresses** and set specific address/addresses or check **Secondary address.**
3. Select the same **Baudrate** corresponding to the baudrate in the meter.
4. Press **Scan**.

Scan meter with M-Bus IR



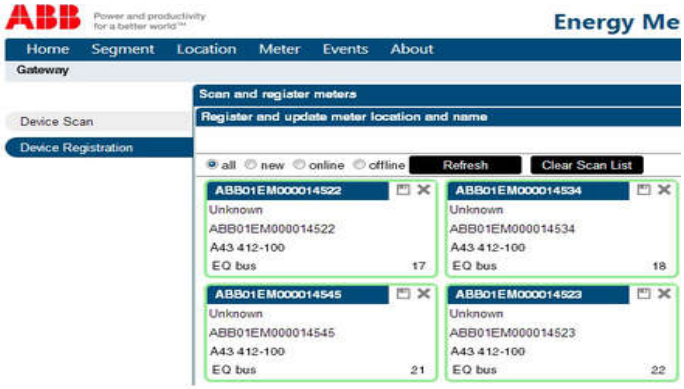
Scan and add a via **M-Bus IR**:

1. Check the **M-Bus IR** box.
2. Check **Primary Address** or **Secondary Address** radio button.
3. Select corresponding baudrate to the meter.
4. Press **Scan**.

4.5.2 Add scanned meter

The scanned meter is presented with an **ADD** button in the top right corner. The meter has no name or location.

If a scanned meter is not added, it will be listed in the Scan and register meters the next time Device scan is performed.



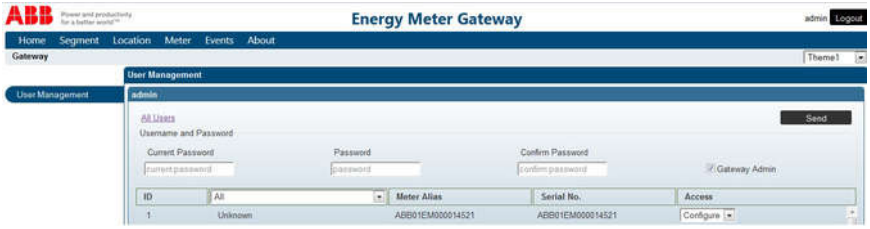
To add the meter:

1. Set **Location name** by clicking on current name. Write new location name.
2. Set **Device Alias** by clicking on Device Name. Write new meter alias.
3. Press **ADD**.
4. Press **Send**.

Changes are not made until data is sent.

4.5.3 User Management

Access level

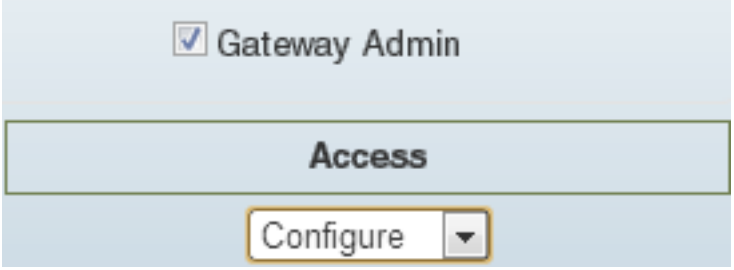


Distributing **User Management** from **Segment**/**User Management** menu. It is possible to set the meter access level for administrators and users in three levels:

- No Access - No access to read or configure the meter.
- View only - Possible to view values from the meter.
- Configure - Full rights to configure the meter.

Enable Gateway admin access

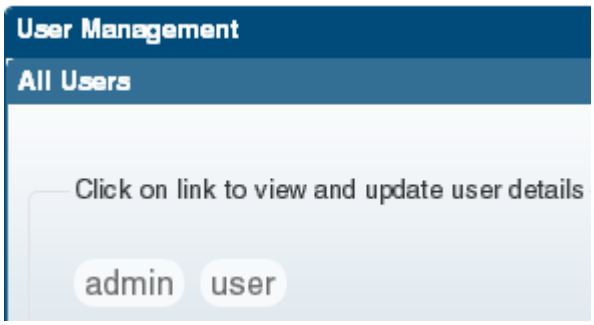
Only Gateway admin can make changes in Gateway settings, reboot the Gateway, update Gateway firmware and clear Event log.



To enable Gateway admin access:

1. Click **Segment** menu.
2. Click **User Management**.
3. Click **Admin**.
4. Check the **Gateway Admin** box.
5. Click **Send**.

Add meter admin access

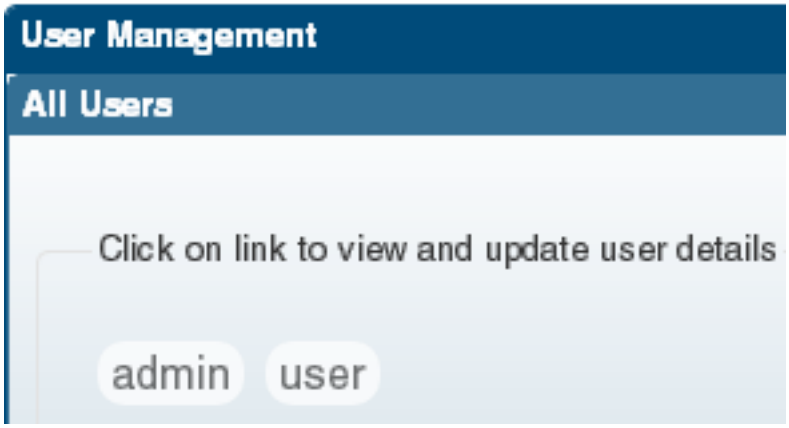


To add meter admin access to a user:

1. Click **Segment** menu.
2. Click **User Management**.
3. Click **User**.
4. Select location.
5. Select access.
6. Click **Send**.

Set Password





To set password:

1. Select **Segment** menu.
2. Select **User Management**.
3. Set password in the **Password** field.
4. Confirm password in **Confirm Password** field.
5. Click **Send**.

Note – Gateway admin can set new password for users without the old password.

Change password

To change password:

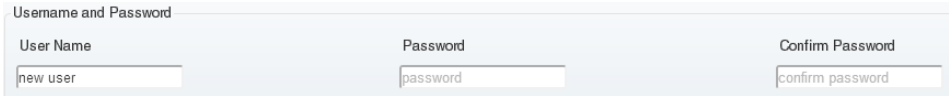
1. Select **Segment** menu.
2. Select **User Management**.
3. Set current password in **Current Password** field.
4. Set the new password in the **Password** field.
5. Confirm the new password in **Confirm Password** field.
6. Click **Send**.

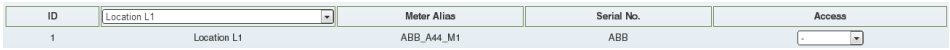
Add User



To add user:

1. Click **Segment**.
2. Click **User Management**
3. Click **Add user**, located to the right.
4. Set the new user name in the **User Name** field.
5. Set the password in the **Password** field.
6. Confirm the password in the **Confirm Password** field.
7. Select **Access** level to preferred meters.
8. Click **Send**.





Note – Username must only consist of lowercase letters and numbers from 0-9.

Chapter 5: Meter Settings

Overview

This chapter is an overview of how to configure the meter settings via the User Interface.

In this chapter

The following topics are covered in this chapter:

5.1 Setting and Configuration 36
5.1.1 General Settings 36
5.1.2 Previous Value Configuration 39
5.1.3 Load Profile Configuration 40
5.1.4 Demand Configuration 41
5.1.5 Tariff Configuration 42
5.1.6 Input/ Output Configuration 44
5.1.7 Pulse Output Configuration 45
5.1.8 Alarm Configuration 47

5.1 Setting and Configuration

General

Depending on the meter type, all or a subset of the following functions can be configured from the meter menu:

- **General Settings**
- **Previous Value Configuration**
- **Load Profile Configuration**
- **Demand Configuration**
- **Tariff Configuration**
- **Input / Output Configuration**
- **Pulse Output Configuration**
- **Alarm Configuration**

Setting a Value

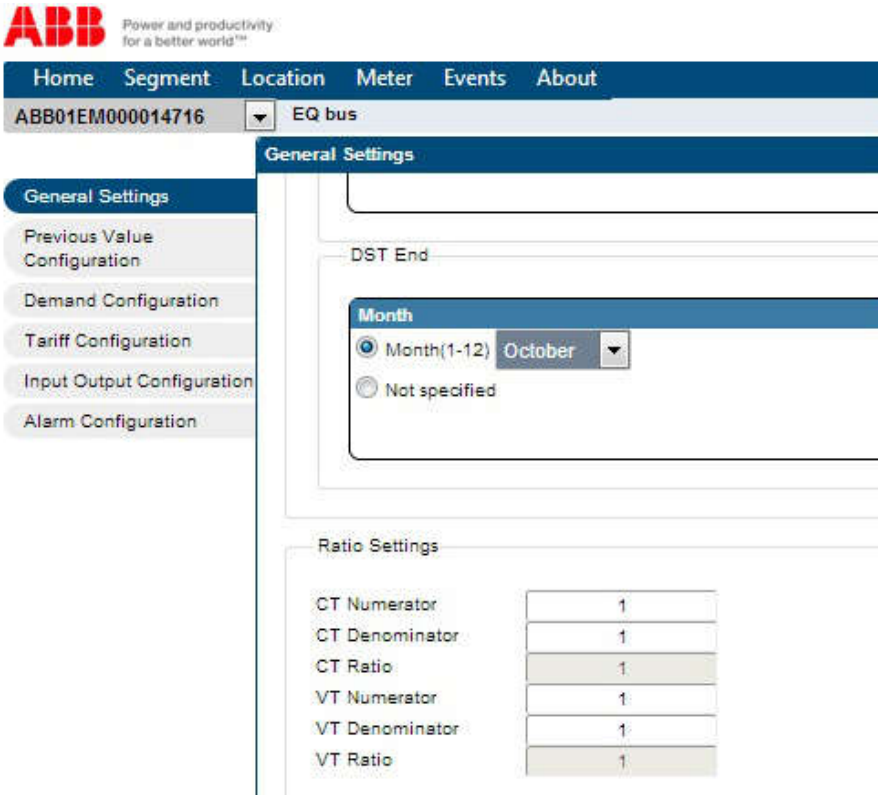
When a value is changed, the **Send** button sends the value to the meter.

5.1.1 General Settings

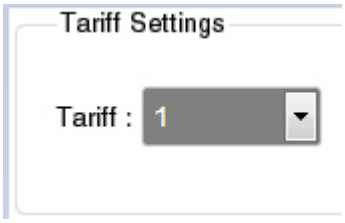
Ratio Settings

To set the ratio, perform the following steps:

1. Select **General Settings** from the **Meter Menu**.
2. Scroll down to **Ratio Settings**.
3. Set the CT/VT ratio by entering the Numerator and Denominator fields (For example: 20 / 1 or 100 / 5)
4. Click **Send**.



Tariff Settings



Tariff settings can change the source of the Tariffs as well selecting the active Tariff on the specific meter. If the meter has a built-in clock the tariff source Clock can be used. If the meter has Inputs the tariffs can be controlled by Inputs. If the meter has communication the tariffs can be controlled by communication.

To set the Active tariff in a meter, perform the following steps

1. Select **Meter Menu**.
2. Select **General Settings** and scroll down to **Tariff Settings**.
3. Select **Tariff Source** from drop-down menu.

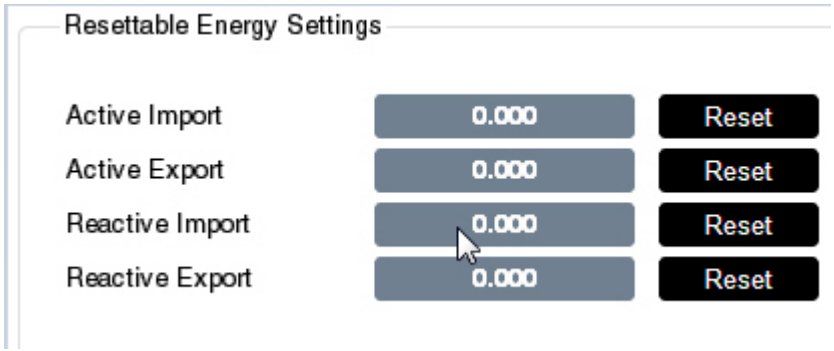


4. Select **Tariff** from drop-down menu.



5. Click **Send**.

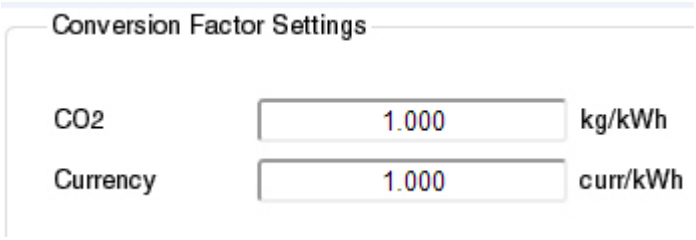
Resettable Energy Settings



To reset registers, perform the following steps:

1. Select **General Settings** from the **Meter Menu**.
2. Scroll down to **Resettable Energy Settings**.
3. Select energy setting to reset.
4. Click **Reset**.
5. Click **Send**.

Conversion Factor Settings

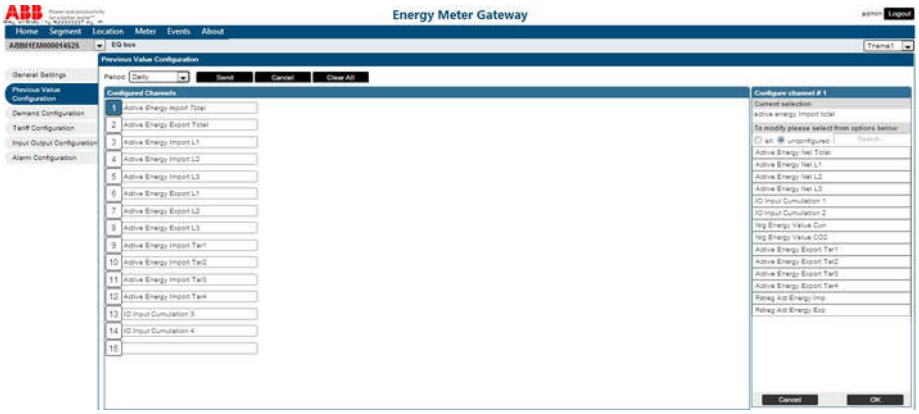


To set the **Conversion Factor Settings** perform the following steps:

1. Select **General Settings** from the **Meter Menu**.
2. Scroll down to **Conversion Factor Settings**.
3. Set the **CO_2** value.
4. Set the **Currency** value.
5. Click **Send**.

5.1.2 Previous Value Configuration

At the end of a defined period, up to 23 configurable channels, which can contain energy register values, input counter values and currency/CO_2 values, are stored together with the current time/date and a snapshot is taken of the predefined values at the set period.



The period options are:

- Daily
- Weekly
- Monthly

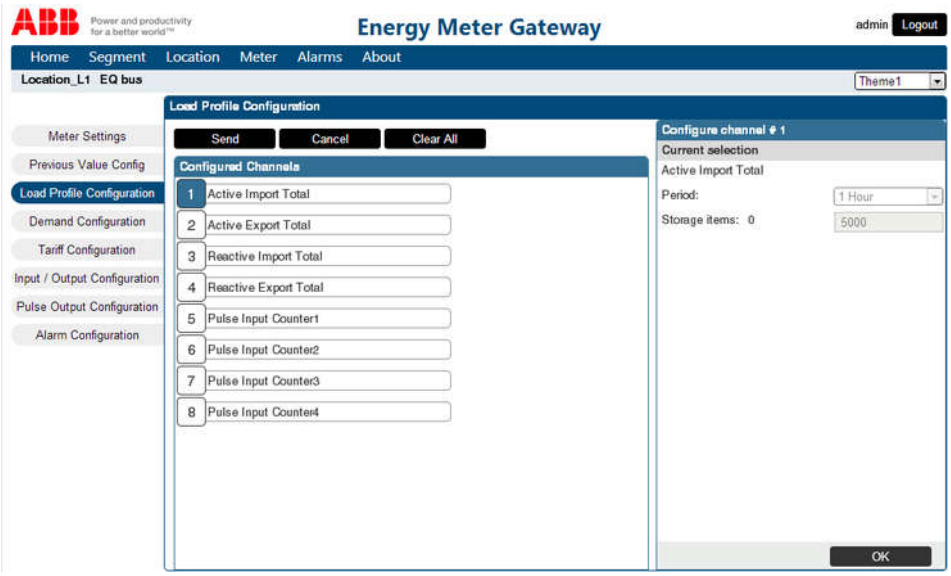


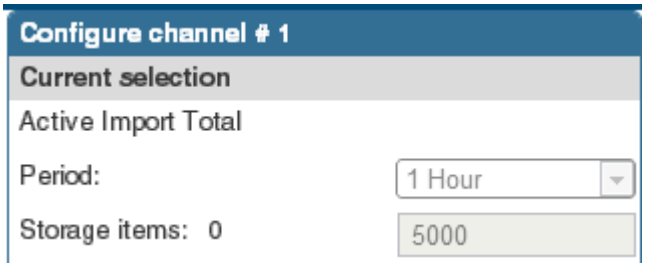
Configure Previous values channels

1. Select or add a new channel to configure.
2. Click on a value in the **Configure Channel** field.
3. Select a value from the options list.
4. Click **OK**.
5. Click **Send**.

5.1.3 Load Profile Configuration

Each channel is configurable from the option window located to the right for each channel separately.



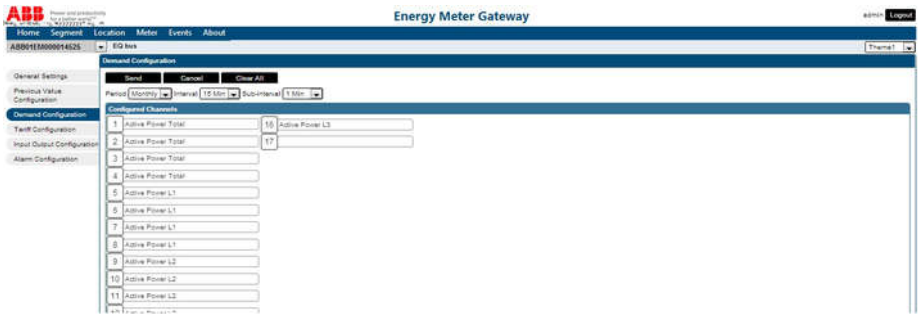


To configure a value for a channel:

1. Choose a channel from the **Configured Channels** list.
2. Select **Period**.
3. Select **Storage items**.
4. Select a value from the options list.
5. Click **OK**.
6. Click **Send**.

5.1.4 Demand Configuration

The Demand Configuration records values over time for 50 channels with a sliding functionality and graphical display for each channel.



1. Set **Period** between: **Monthly**, **Daily** or **Weekly**.
2. Set the **Interval** between: 1- 60 min (sub-intervals: 1, 2, 5, 10, 15, 30, 60 min).
3. The **Sub-Interval** can be set between 1-15 min depending on the selected interval.

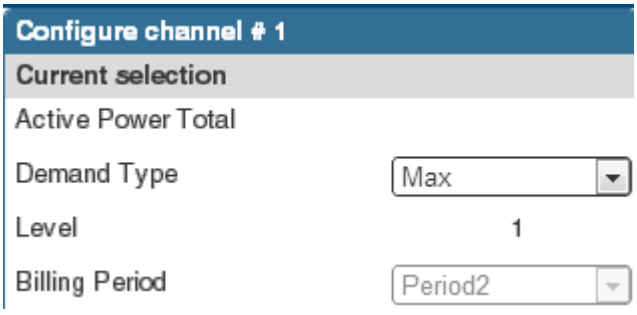
Example:

If interval 1-60 minutes is chosen, then the sub-intervals between 1, 2, 5, 10, 30, 60 minutes are available.

If interval 1-15 minutes is chosen, then, the sub-intervals between 1, 5, 10, 15 minutes are available.

Note that sub-interval 2 is not available for interval 1-15 minutes, since 15 is not even dividable with 2.

Each channel is separately configurable from the option window located to the right.



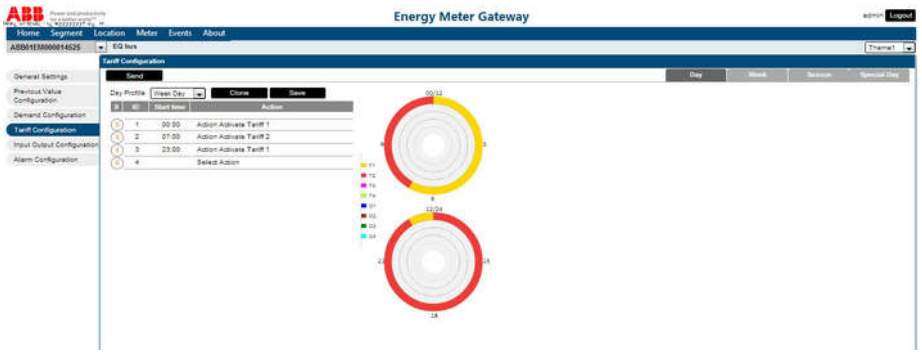
To configure a value for a channel:

1. Click a channel from the **Configured Channels** list.
2. Select **Demand Type** between: **Min**, **Max**, **Min Sliding** and **Max Sliding**.
3. Select a value from the options list.
4. Click **OK**.
5. Click **Send**.

5.1.5 Tariff Configuration

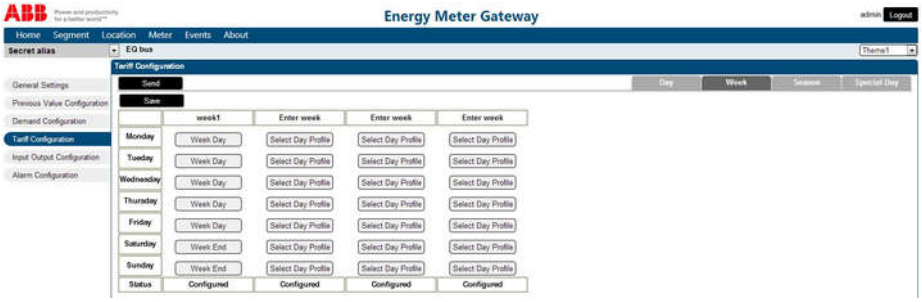
Within Tariff Configuration each tariff can be configured. Tariff Configuration consists of the tabs **Day**, **Week**, **Season** and **Special Day**. The **Save** command saves locally all changes made, while the **Send** command sends the changes made to the meter. When the user leaves a tab, all changes that have not been saved are discarded.

The Day tab



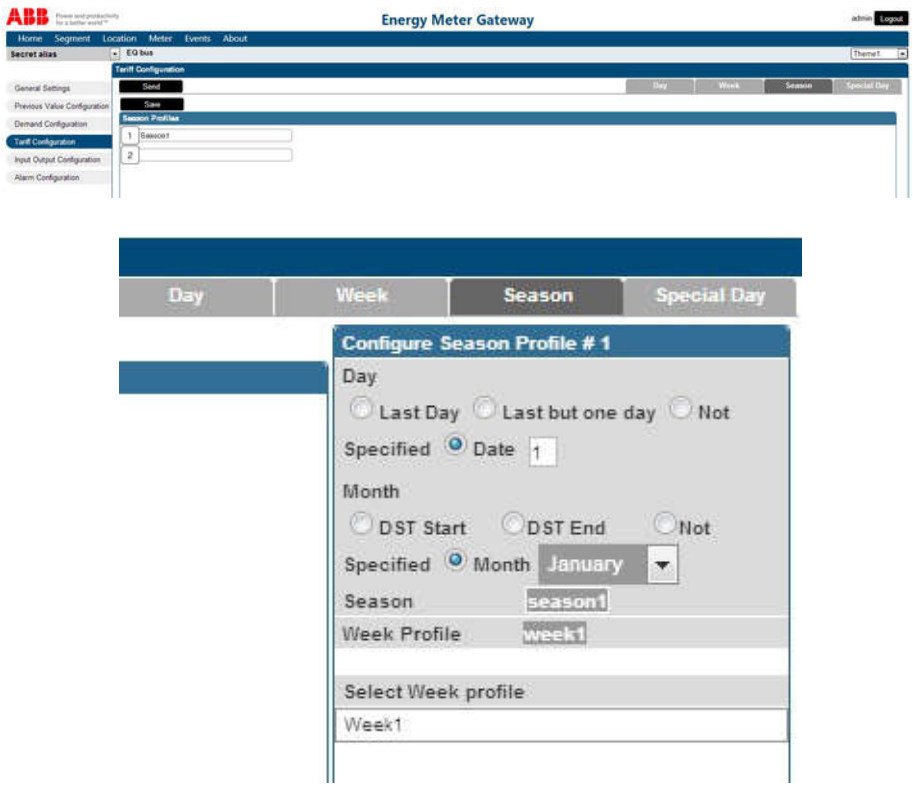
Within the Day tab a list of switch points with **Start time** and **Action** can be set for each day profile. There can be a maximum of 30 switch points per day, and a maximum of 16 day profiles. Remember to **Save** and **Send**.

The Week tab



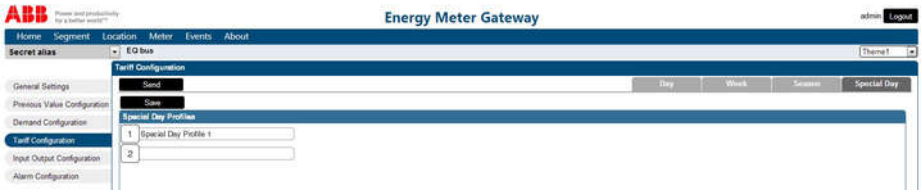
Within the Week tab each day of a selected week can be set to either a **Week Day** or a **Week End**. Remember to **Save** and **Send**.

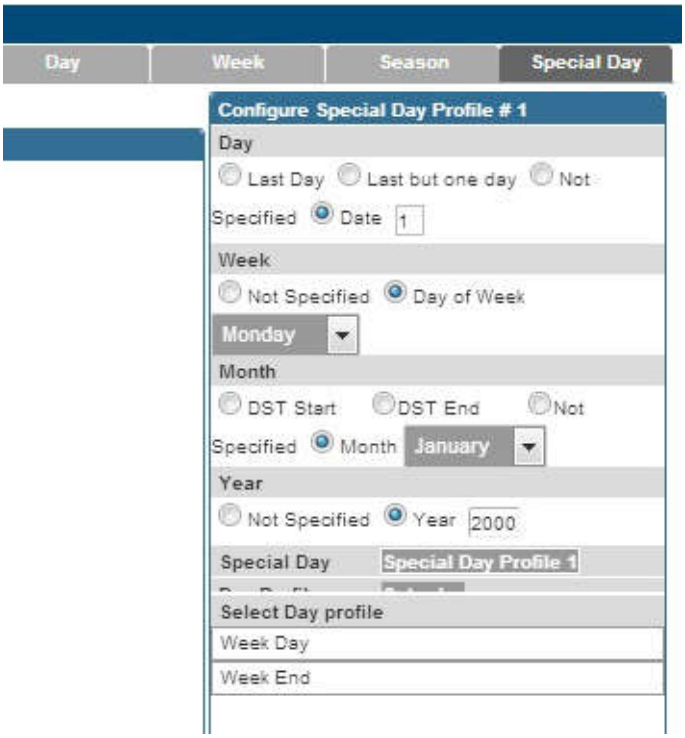
The Season tab



Within the Season tab **Season Profiles** can be configured. Remember to **Save** and **Send**.

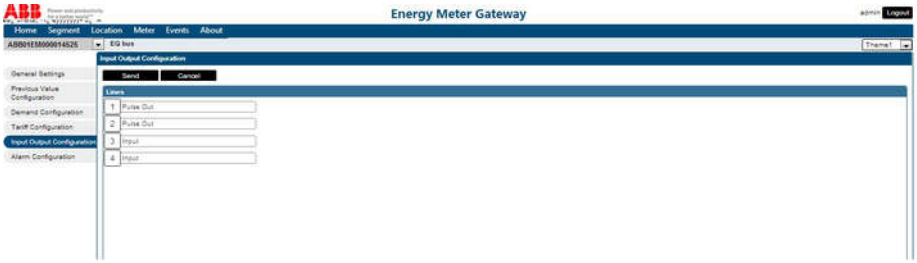
The Special Day tab

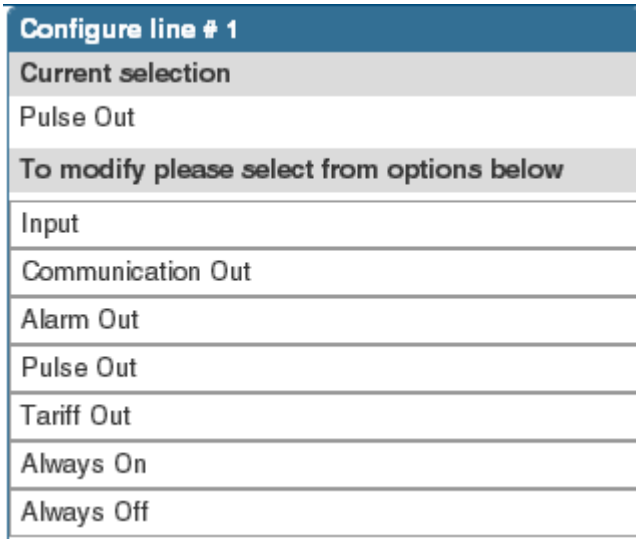




Within the Special Day tab **Special Day Profiles** can be configured. Remember to **Save** and **Send**.

5.1.6 Input/ Output Configuration

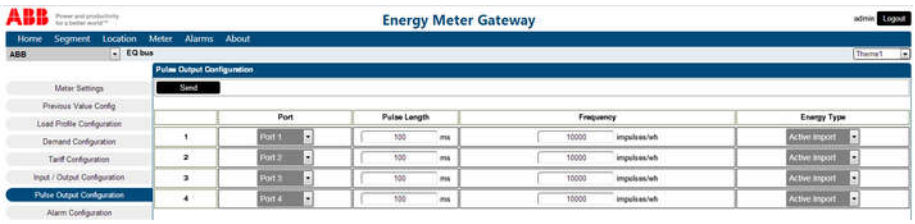




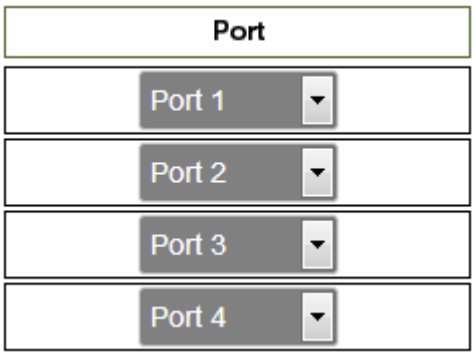
To configure **Input** /**Output** for a channel perform the following steps:

1. Click a line from the **Lines** list.
2. Select a value from the option list located to the right.
3. Click **OK**.
4. Click **Send**.

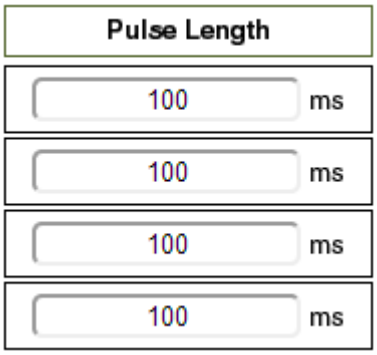
5.1.7 Pulse Output Configuration



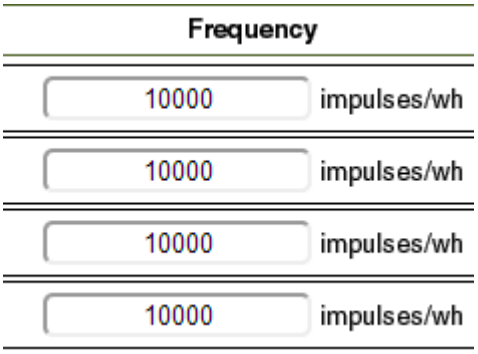
1. To configure Output line:
2. Select **Port** from 1- 4.



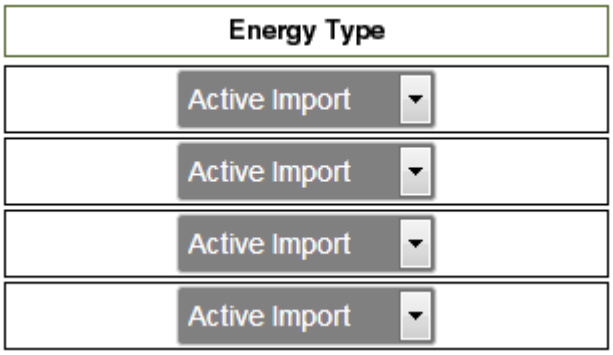
3. Set **Pulse Length**.



4. Set **Frequency**.

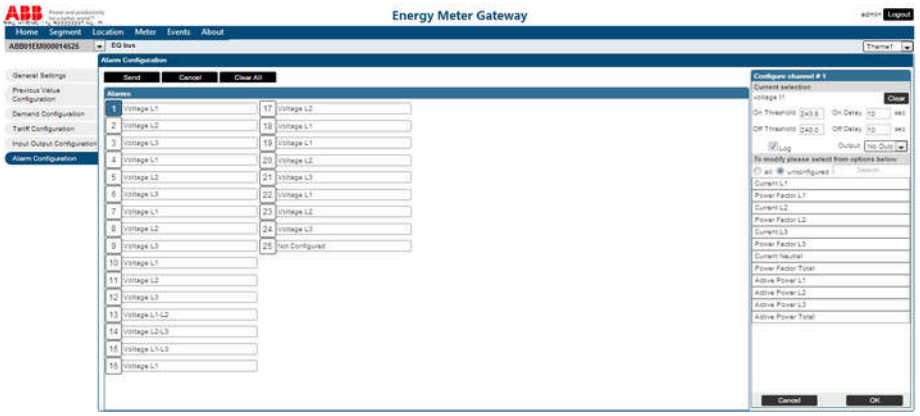


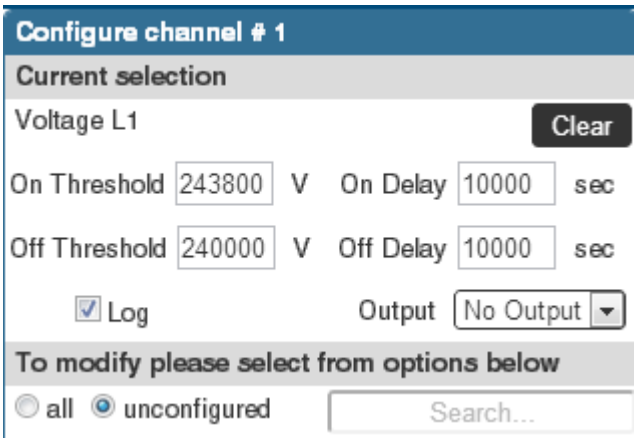
5. Select **Energy Type**.



6. Click **Send**.

5.1.8 Alarm Configuration





To configure an alarm for a channel perform the following steps:

1. Click a channel from the **Alarms** list.
2. Select a value to configure from the option list below.
3. Set **On Threshold** value for activation of the alarm.
4. Set **On Delay** for delay of the alarm.
5. Set **Off Threshold** value to set when the alarm is turned off.
6. Set **Off Delay** to set the time for the alarm to turn off.
7. Check the **Log** box to log the alarms.
8. Select an **Output** between 1-4.
9. Click **OK**.
10. Click **Send**.

Chapter 6: JSON Communication

Overview

This chapter will address communication using the JSON protocol.

In this chapter

The following topics are covered in this chapter:

6.1 About JSON 47
6.2 Table of Resources 50
6.3 Resources 53

6.1 About JSON

Target Groups

The intended target groups for this chapter are user interface developers and API consumers.

Introduction

This is a reference document for RESTful JSON API v1.0(Draft) of ABB Energy Meter Gateway 1.0.

Current version of API supports following HTTP verbs

- HTTP GET
- HTTP POST
- HTTP PUT
- HTTP DELETE

MIME used with all types of http verbs is `application/json`

Authentication

API is hosted by the Gateway Webserver via `https` end point.

Basic authentication mechanism is used for authentication. However all traffic goes through SSL layers.

Thus all messages should have the following authorization header as shown below:

```
GET /meter HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

General Conventions

The API is designed with the following assumptions and considerations:

- At any time gateway supports only one version of API
- Query parameters in the form of "`?key=value`" are not allowed with any of the supported http verbs
- Update requests are made with `Content-Type: application/json`.
  `form-urlencoded` style is not allowed.
- Max payload size in case of update requests (POST, PUT, DELETE) is 512 bytes.
  In case of firmware upload with `multipart/form-data` API user can loop data in chunks of 512 bytes.
- Responses for GET requests are always streamed. And response may be of varied length.
- GET request will not have payload. URI itself should provide all required information
- POST, PUT, DELETE requests will have payload as well as arguments in URI as applicable.
- Requests applicable to a particular meter will always start with `/meters/<serial>`.
  Higher level resource `/meters` returns list of all meters and their serial numbers.
- Responses may vary depending on type of meter.
  Example:
  Phase angles are not supported in all types of meters.
  Also there are some resources which are applicable in case of advanced meters only.
  Harmonics are supported in platinum series of meters only.

Error Handling

Error handling is designed around HTTP status codes.

Server would respond with the following status codes:

- 20x - success. Complete or partially success
- 40x - authentication, JSON parsing etc. where the gateway refused to process the request
- 50x - gateway started processing the request, but failed while performing the operation

In case of GET requests response may include valid data or status message.

In case of update (POST, POST, DELETE) requests, the response would always be status message.

Format of status message is `{"status":"error","des":"URI not found."}`.

Where

- `status` field might include `error` or `success`
- `des` includes context specific message

When body of POST request includes multiple transactions, result is derived as success when all transactions succeed. Else result would be error and description includes transaction reference at which the POST operation is aborted.

Note: This might lead to inconsistent state of configuration. Thus it is recommended to verify data in such cases. Else send them individually.

Example: When setting conversion factors for CO2 and Currency, if server could update CO2 and fails to update Currency, then the result would be error though CO2 factor is set. However, if CO2 fails then Currency will not be processed at all.

6.2 Table of Resources

The following table will give an overview of the different resources.

Overview of resources

ID	Resource
1	GET /about
2	POST /login
3	POST /logout
4	GET /configuration
5	POST /configuration/ip
6	POST /configuration/eqbus
7	POST /configuration/mbus-wired
8	POST /configuration/mbus-ir
9	GET /gateway
10	POST /gateway
11	GET /gateway/datetime
12	POST /gateway/datetime
13	GET /gateway/events
14	DELETE /gateway/events
15	POST /gateway/reboot
16	POST /gateway/execute
17	GET /users
18	POST /users/<user>
19	PUT /users
20	DELETE /users/<user>
21	GET /users/<user>/bindings
22	PUT /users/<user>/bindings
23	DELETE /users/<user>/bindings
24	GET /meters/firmwareupdatestatus
25	POST /firmware
26	PUT /firmware
27	PUT /meters/firmware
28	GET /parametermapping
29	GET /storablequantities
30	GET /meters/<serial>/associationobjects
31	GET /meters
32	POST /meters/<serial>
33	PUT /meters
34	DELETE /meters/<serial>
35	GET /meters/scanned
36	POST /meters/scanned/<token>

ID	Resource
37	DELETE /meters/scanned
38	GET /permittedmeters
39	PUT /permittedmeters
40	DELETE /permittedmeters
41	GET /meters/<serial>/datetime
42	POST /meters/<serial>/datetime
43	GET /meters/<serial>/info
44	GET /meters/<serial>/energy/conversionfactor
45	POST /meters/<serial>/energy/conversionfactor
46	GET /meters/<serial>/hardwareversion
47	GET /meters/<serial>/mbusinfo
48	POST /meters/<serial>/transformersettings
49	GET /meters/<serial>/status
50	GET /meters/<serial>/statusflags
51	GET /meters/<serial>/events/<datetime>/<count>
52	GET /meters/<serial>/alarms/configuration
53	POST /meters/<serial>/alarms/configuration
54	GET /meters/<serial>/energy/<type>
55	GET /meters/<serial>/energy/<type>/<mode>
56	GET /meters/<serial>/energy/resettable
57	POST /meters/<serial>/energy/resettable
58	GET /meters/<serial>/energy/resetcounter
59	GET /meters/<serial>/power
60	GET /meters/<serial>/instrument
61	GET /meters/<serial>/harmonics/voltage
62	GET /meters/<serial>/harmonics/current
63	GET /meters/<serial>/io
64	GET /meters/<serial>/io/configuration
65	GET /meters/<serial>/io/pulse
66	POST /meters/<serial>/io
67	POST /meters/<serial>/io/configuration
68	POST /meters/<serial>/io/pulse
69	GET /meters/<serial>/previousvalues/<fromdate>/<count|todate>
70	GET /meters/<serial>/previousvalues/configuration
71	POST /meters/<serial>/previousvalues/configuration
72	DELETE /meters/<serial>/previousvalues
73	GET /meters/<serial>/loadprofiles/<channel>/<fromdate>/<count|todate>
74	GET /meters/<serial>/loadprofiles/configuration
75	POST /meters/<serial>/loadprofiles/configuration
76	DELETE /meters/<serial>/loadprofiles
77	GET /meters/<serial>/demand/<fromdate>/<count|todate>

ID	Resource
78	GET /meters/<serial>/demand/configuration
79	POST /meters/<serial>/demand/configuration
80	DELETE /meters/<serial>/demand/
81	GET /meters/<serial>/tariff
82	POST /meters/<serial>/tariff
83	GET /meters/<serial>/tariff/dayprofiles
84	POST /meters/<serial>/tariff/dayprofiles
85	GET /meters/<serial>/tariff/weekprofiles
86	POST /meters/<serial>/tariff/weekprofiles
87	GET /meters/<serial>/tariff/seasonprofiles
88	POST /meters/<serial>/tariff/seasonprofiles
89	GET /meters/<serial>/tariff/specialdayprofiles
90	POST /meters/<serial>/tariff/specialdayprofiles
91	GET /lasterror/<id>

6.3 Resources

6.3.1 GET /about

A GET call to /about returns version information of key components of the gateway.

Protected

Yes

Request

```
GET /about HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/about

Response

- 200 Successful

```
HTTP/1.1 200 OK
    Content-Type: application/json
    Server: embOS/IP
    Transfer-Encoding: chunked
    Link:/lasterror/0

     {
       "cosemversion": 6,
       "firmwareversion": "G0.0.0.0",
       "hardwareversion": "",
       "model": "",
       "jsonapiversion": "0.9",
       "logicaldevicename": "ABBXXXXXXXXXXXXX",
       "releasedate": "2012-08-30T14:06:26",
       "serial": 1234,
     }
```

Description

- cosemversion[Int]: COSEM version
- firmwareversion[String 17]: Gateway firmware version
- hardwareversion[String 18]: Gateway hardware version
- model[String 12]: Gateway model number
- jsonapiversion[String 10]: JSON API version number
- logicaldevicename[String 17]: Gateway device name
- releasedate[String]: Gateway Firmware build release date
- serial[Int]: Gateway serial number
- 401 Authentication Failure

```
HTTP/1.1 401 Unauthorized
Server: embOS/IP
Accept-Ranges: bytes
Content-Type: text/html
```

6.3.2 POST /login

Creates a session for user with valid credentials.

Protected

Yes (Authentication required)

Request

```
POST /login HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Example

- https://192.168.1.12/login

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Link:/lasterror/0
```

Notes

A successful response to be considered as valid login session. If user did not change default password, provided by administrator, then server will respond with 403 Forbidden. In such case user is expected to set password before proceeding with other operations. JSON API users must call

`[POST] /login` with 'oldkey' and 'key' tags as shown below. Also users can explicitly logout of a session.

```
https://192.168.1.12/login
{
    "name": "Username1",
    "oldkey": "user1234",
    "key": "user4321"
}
```

Description

- name [String(16)] - Username.
- oldkey [String(32)] - Current password of user.
- key [String(32)] - New password of user.
- 401 Unauthorized

```
HTTP/1.1 401 Unauthorized
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 162
Content-Type: text/html

<HTML>
  <HEAD><TITLE>401 Unauthorized</TITLE></HEAD>
  <BODY>
    <H1>401 Unauthorized</H1>
      Browser not authentication-capable or
authentication failed.
  </BODY>
</HTML>
```

- 403 Forbidden

```
HTTP/1.1 403 Forbidden
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 36
Content-Type: text/html
X-Pad: avoid browser bug
Connection: close

Please reset password and try again.
```

6.3.3 POST /logout

Logs out of a valid user session.

Protected Yes (Authentication required)

Request

```
POST /logout HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Example

- https://192.168.1.12/logout

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Link:/lasterror/0
```

Note

Successful response to be considered as successful logout.

- 401 Unauthorized

```
HTTP/1.1 401 Unauthorized
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 162
Content-Type: text/html

<HTML>
  <HEAD><TITLE>401 Unauthorized</TITLE></HEAD>
  <BODY>
    <H1>401 Unauthorized</H1>
      Browser not authentication-capable or
authentication failed.
  </BODY>
</HTML>
```

6.3.4 GET /configuration

A GET call to **/configuration** returns TCP/IP settings, backplane settings and backplane timeouts (EQ bus, M-Bus Wired and M-Bus IR).

Protected

Yes (Authentication required)

Request

```
GET /configuration HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Example

- https://192.168.1.12/configuration

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "ip": {
        "dhcp": false,
        "dns": false,
        "gateway": "192.168.1.1",
        "ip": "192.168.1.12",
        "mac": "255.255.255.199.34.0",
        "pdns": "0.0.0.0",
        "sdns": "0.0.0.0",
        "subnet": "255.255.255.0"
    },
    "eqbus": {
        "baudrate": 500000,
        "interoctettimeout": 40
    },
    "mbus-wired": {
        "ack": 1000,
        "delay_after_telegram": 1000,
        "delay_before_requd2": 200,
        "get_request": 0,
        "meter_contact": 0,
        "response_timeout": 0,
        "response_timeout_base": 800
    },
    "mbus-ir": {
        "ack": 1000,
        "delay_after_telegram": 1000,
        "delay_before_requd2": 200,
        "get_request": 0,
        "meter_contact": 0,
        "response_timeout": 0,
        "response_timeout_base": 800
    }
}
```

Notes

For detailed description of individual parameters, please refer to respective POST URI.

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.5 POST /configuration/ip

A POST call to /configuration/ip sets IP settings.

Protected Yes (Authentication required)

Request

```
POST /configuration/ip HTTP/1.1
   Authorization: Basic XXXXXXXXXXXXXXX=
   Content-Type:application/json
   Content-Length: 179

   {
      "ip":"192.168.1.12",
      "dhcp":false,
      "dns":false,
      "gateway":"192.168.1.1",
      "subnet":"255.255.255.0",
      "pdns":"192.168.1.2"
      "sdns":"192.168.1.2"
   }
```

Description

- dhcp [bool] : Value of **false** means static ip setting and value of **true** means dynamic ip from dhcp server
- dns [bool] : Value of **false** means pdns is the name server and value of **true** means dynamic dns from dhcp server.
- ip[string] : IP Address.
- gateway[string] : Gateway Address.
- subnet[string] : Subnet Mask Address.
- pdns[string] : Primary DNS Address.
- sdns[string] : Secondary DNS Address.
  Notes
  With static IP settings DNS cannot be auto and must be set statically.
  Current implementation supports static DNS settings only.

Example

- https://192.168.1.12/configuration/ip

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 400 Bad Request

```
HTTP/1.1 400 Bad Request
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 57
Content-Type: application/json

{
    "status":"error",
    "des":"missing ip in JSON ."
}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.6 POST /configuration/eqbus

A POST call to **/configuration/eqbus** sets settings for eqbus backplane.

Protected

Yes (Authentication required)

Request

```
POST /configuration/eqbus HTTP/1.1
   Authorization: Basic XXXXXXXXXXXXXXX=
   Content-Type:application/json
   Content-Length: 46

 {
     "baudrate": 500000,
     "interoctettimeout": 1000
 }
```

Description

- baudrate[Int]: Supported baudrates are listed below
  1200
  2400
  9600
  19200
  38400
  57600
  115200
  125000
  230400
  460800
  500000
- interoctettimeout[Int] : should be in between 20 and 6000 seconds.

Notes

Baudrate settings are not propagated to meters. This call changes baudrate in gateway only.

Users need to match settings in all meters using HMI of each meter.

Examples

- https://192.168.1.12/configuration/eqbus

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 400 Bad Request

```
HTTP/1.1 400 Bad Request
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 57
Content-Type: application/json

{
    "status":"error",
    "des":"missing baudrate in JSON ."
}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.7 POST /configuration/mbus-wired

A POST call to **/configuration/mbus-wired** sets timeout parameters for mbus-wired backplane.

Protected

Yes (Authentication required)

Request

```
POST /configuration/mbus-wired HTTP/1.1
 Authorization: Basic XXXXXXXXXXXXXXX=
 Content-Type:application/json
 Content-Length: 186

 {
     "ack": 250,
     "delay_after_telegram": 100,
     "delay_before_requd2": 100,
     "get_request": 200,
     "meter_contact": 150,
     "response_timeout": 300,
     "response_timeout_base": 100
 }
```

Description

- timeout values are expressed in **seconds**.
- ack: value should be between 200 and 2000
- delay_after_telegram: value should be between 50 and 60000
- delay_before_requd2: should be between 50 and 1000
- response_timeout: should be between 300 and 15000
- response_timeout_base: should be between 100 and 10000
- meter_contact: should be between 10 and 25000
- get_request: should be between 10 and 75000

Example

- https://192.168.1.12/configuration/mbus-wired

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 400 Bad Request

```
HTTP/1.1 400 Bad Request
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 57
Content-Type: application/json

{
    "status":"error",
    "des":"missing baudrate in JSON ."
}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.8 POST /configuration/mbus-ir

A POST call to **/configuration/mbus-ir** sets timeout parameters for IR interface.

Protected

Yes (Authentication required)

Request

```
POST /configuration/mbus-ir HTTP/1.1
 Authorization: Basic XXXXXXXXXXXXXXX=
 Content-Type:application/json
 Content-Length: 186

 {
     "ack": 250,
     "delay_after_telegram": 100,
     "delay_before_requd2": 100,
     "get_request": 200,
     "meter_contact": 150,
     "response_timeout": 300,
     "response_timeout_base": 100
 }
```

Description

- timeout values are expressed in **seconds**.
- ack: value should be between 200 and 2000
- delay_after_telegram: value should be between 50 and 60000
- delay_before_requd2: should be between 50 and 1000
- response_timeout: should be between 300 and 15000
- response_timeout_base: should be between 100 and 10000
- meter_contact: should be between 10 and 25000
- get_request: should be between 10 and 75000

Example

- https://192.168.1.12/configuration/mbus-ir

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 400 Bad Request

```
HTTP/1.1 400 Bad Request
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 57
Content-Type: application/json

{
    "status":"error",
    "des":"missing baudrate in JSON ."
}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.9 GET /gateway

A GET call to /gateway returns alias, location and serial.

Protected

Yes (Authentication required)

Request

```
GET /gateway HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Example

- https://192.168.1.12/gateway

Response

- 200 OK

```
HTTP/1.1 200 OK
    Content-Type: application/json
    Server: embOS/IP
    Transfer-Encoding: chunked
    Link:/lasterror/0

    {
      "alias":"Gateway",
      "location":"All Location",
      "serial":"ABBXXXXXXXXXXXX01"
    }
```

Description

- alias [String 21] - Gateway alias.
- location [String 51] - Location of gateway.
- serial [String 17] - Serial of gateway.
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.10 POST /gateway

A POST call to /gateway sets Gateway alias and location specified.

Protected

Yes (Authentication required)

Request

```
POST /gateway HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Type:application/json
Content-Length: 77

{
    "serial": "ABBXXXXXXXXXXXX01",
    "alias": "Gateway",
    "location": "All Location"
}
```

Description

- alias [String 21] - Gateway alias.
- location [String 51] - Location of gateway.
- serial [String 17] - Serial of gateway.

Examples

- https://192.168.1.12/gateway

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 400 Bad Request

```
HTTP/1.1 400 Bad Request
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 57
Content-Type: application/json

{
    "status":"error",
    "des":"missing datetime tag in JSON ."
}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.11 GET /gateway/datetime

A GET call to /gateway/datetime returns current date and time of gateway in standard date time format (YYYY-MM-DDTHH:MM:SS)

Protected

Yes (Authentication required)

Request

```
GET /gateway/datetime HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/gateway/datetime

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "datetime":"2012-05-09T16:25:05",
    "status":"set"
}
```

Description

- datetime [String] - Gateway Datetime.
- status [String(8)] - ClockStatus.["not set","set"].
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.12 POST /gateway/datetime

A POST call to /gateway/datetime sets specified date and time in standard date time format (YYYY-MM-DDTHH:MM:SS)

Protected

Yes (Authentication required)

Request

```
POST /gateway/datetime HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Type:application/json
Content-Length: 34

    {"datetime":"2012-05-09T16:30:52"}
```

Description

- Datetime[String]: Gateway date and time

Example

- https://192.168.1.12/gateway/datetime

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 400 Bad Request

```
HTTP/1.1 400 Bad Request
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 57
Content-Type: application/json

{
    "status":"error",
    "des":"missing datetime tag in JSON ."
}
```

- 500 Server Internal Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.13 GET /gateway/events

A GET call to /gateway/events returns the events log recorded by the gateway.

Protected

Yes (Authentication required)

Request

```
GET /gateway/events HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Example

- https://192.168.1.12/gateway/events

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
  "events":[

{"category":4,"index":6,"id":9,"timestamp":"2012-
05-17T13:02:02","type":2},

{"category":8,"index":5,"id":4,"timestamp":"2012-
05-17T12:02:02","type":2},

{"category":8,"index":4,"id":4,"timestamp":"2012-
05-17T12:02:02","type":2},

{"category":8,"index":3,"id":4,"timestamp":"2012-
05-17T12:02:02","type":2},

{"category":4,"index":2,"id":9,"timestamp":"2000-
01-01T00:02:02","type":2},

{"category":8,"index":1,"id":6,"timestamp":"2000-
01-01T00:02:02","type":2}
      ]
}
```

Description

- category[Int]: Category of the event
  1 # Exception
  2 # Error
  3 # Warning
  8 # Information
- index[Int]: Serial number out of events logged since last reset
- id[Int] : Id of log structure as per following table

```
0   WS_STARTED                  Watchdog started
1   WS_STOPPED                    Watchdog stopped
2   GW_SETTIME                Gateway date & time set
3   GW_MAXUSER_LIMIT          Gateway user database
reached maximum limit
4   GW_USER_ADDED             User added to Gateway
5   GW_USER_DELETED             User deleted from
Gateway
6   GW_DEVICE_ADDED             Device added to
Gateway
7   GW_DEVICE_DELETED          Device deleted from
Gateway
8   GW_CONNECTIONLIMIT         Gateway has reached
its maximum connections
9   GW_METER_NOTCONNECTED       Connection with
meter lost
10  GW_REBOOT                   Gateway reboot
initiated
11  GW_STARTED                  Gateway started
12  GW_EVENTS_RESET            Gateway events are
reset
13  GW_IP_ADDRESS_CHANGED      Gateway IP address
changed
14  GW_ALIAS_CHANGED          Gateway alias changed
15  GW_LD_FW_UPLOAD_COMPLETE   Firmware upload to
Gateway completed
16  GW_LD_FW_UPLOAD_FAIL       Firmware upload to
Gateway failed
17  GW_FW_UPGRADE_START         Gateway Firmware
upgrade started
18  GW_FW_UPGRADE_COMPLETE      Gateway Firmware
upgrade completed
19  GW_FW_UPGRADE_FAIL          Gateway Firmware
upgrade failed
20  LD_FW_UPGRADE_START         Upgrade firmware
to Logical device started
21  LD_FW_UPGRADE_COMPLETE      Upgrade firmware
```

```
to Logical device complete
22  LD_FW_UPGRADE_FAIL          Upgrade firmware to
Logical device failed
```

- timestamp[String]: timestamp when event occurred
- type[Int]: Type of event occurred
  0 # Empty
  1 # System
  2 # Event
  3 # Audit
  4 # Quality
  5 # Communication
  6 # Transformer Ratio

6.3.14 DELETE /gateway/events

A DELETE call to /gateway/events clears event log of gateway.

Protected Yes (Authentication required)

Request

```
DELETE /gateway/events HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Example

- https://192.168.1.12/gateway/events

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

6.3.15 POST /gateway/reboot

A POST call to /gateway/reboot reboots the Gateway. Only admin can reboot the Gateway.

Protected Yes (Authentication required)

Request

```
POST /gateway/reboot HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Type:application/json
Content-Length: 0
```

Example

- https://192.168.1.12/gateway/reboot

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "status": "Rebooting..."
}
```

- 403 Forbidden

```
HTTP/1.1 403 Forbidden
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 51
Content-Type: application/json

{
    "status": "error",
    "des": "Forbidden. Access Denied"
}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 63
Content-Type: application/json

{
    "status":"error",
    "des":"Inetrnal Server Error. Reboot failed."
}
```

6.3.16 POST /gateway/execute

A POST call to /gateway/execute executes Gateway command to read the parameters supported as below (for the given OBIS and interface class id).

```
VOLTAGE, CURRENT, ACTIVE_POWER, REACTIVE_POWER,
APPARENT_POWER, ACTIVE_ENERGY_IMPORT,

ACTIVE_ENERGY_EXPORT, ACTIVE_ENERGY_NET,
REACTIVE_ENERGY_IMPORT, REACTIVE_ENERGY_EXPORT,

REACTIVE_ENERGY_NET, APPARENT_ENERGY_IMPORT,
APPARENT_ENERGY_EXPORT, APPARENT_ENERGY_NET,

POWER_FACTOR, Differential VOLTAGE, PHASE_ANGLE_POWER,
PHASE_ANGLE

_VOLTAGE, PHASE_ANGLE_CURRENT,

CURRENT_QUADRANT, NRG_CONVERSION_FACTOR_CURR,
NRG_CONVERSION_FACTOR_CO2,

ACTIVE_ENERGY_EXPORT for TARIFF1 to TARIFF4,
REACTIVE_ENERGY_EXPORT for TARIFF1 to TARIFF4,

REACTIVE_ENERGY_IMPORT for TARIFF1 to TARIFF4,
ACTIVE_ENERGY_IMPORT for TARIFF1 to TARIFF4,

NRG_ENERGY_VALUE_CURR, NRG_ENERGY_VALUE_CO2, Fre-
equency, RSTREG

_ACT_ENERGY for IMPORT and EXPORT,

RSTREG_REACT_ENERGY for IMPORT and EXPORT
```

Limitations

1. It supports only one interface class at any time

Protected Yes (Authentication required)

Requested

```
POST /gateway/execute HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Type:application/json
Content-Length: 88

{"devices":["ABB01","ABB02"],"obis":[1,0,32,7,0,255],"clas-
sid": 3, "attributes":[2,3]}
```

Description

- deviceid[String(17)]
  Array of device serial names. Maximum supported devices are 5.
- classid[Int]
  Interface Class Id
- attributes[value,unit]
  It contains array of attribute values for the given parameter to read. It is dependent on interface class id
  value[Int or String]: Parameter value attribute.
  unit[Int or String] : Parameter unit attribute. This is optional.
- obis OBIS type can be an array of integers or String[23].
  To read VOLTAGE_L1: The payload will be

```
{
 "devices": ["ABB01","ABB02"],
  "obis": [1,0,32,7,0,255],
  "classid": 3,
  "attributes": [2,3]
}
```

  To read POWERFACTOR_L1: The payload will be

```
{
      "devices": ["ABB01","ABB02"],
      "obis": [1,0,33,7,0,255],
      "classid": 3,
      "attributes": [2]
  }
```

Example

- https://192.168.1.12/gateway/execute

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0
```

To read 'VOLTAGE' of L1 , the response is as below:
{ "ABB01":["200.0","V"], "ABB02":["200.0","V"] }
To read POWRFACTOR_L1, the response is as below:
{ "ABB01": ["0.9"], "ABB02": ["1.0"] }

Description

ABB01 or ABB02 is device serial name. Each device contains array values[value,unit] for the given parameter to read.
* value[String] - holds the value of the given parameter to read.
* unit[String] - holds the unit of given parameter to read.

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 63
Content-Type: application/json

{
    "status":"error",
    "des":"Inetrnal Server Error."
}
```

6.3.17 GET /users

A GET call to /users returns list of users of gateway.

Protected

Yes (Authentication required)

Request

```
GET /users HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Example

- https://192.168.1.12/users

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "users": ["Username2", "Username3"]
}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.18 POST /users/<user>

A POST call to /users/<user> updates user information.

Protected

Yes (Authentication required)

Request

```
POST /users/user HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Type:application/json
Content-Length: 32

{"name":"Username1","oldkey":"user1234","key":"user4321"}
```

Description

- name [String] - Username.
- oldkey [String] - Current password of user.
- key [String] - New password of user.

Examples

- https://192.168.1.12/users/user3

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

Notes

In case of discrepancy with username field specified in URI and payload, data specified in payload takes precedence.

- 400 Bad Request

```
HTTP/1.1 400 Bad Request
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 57
Content-Type: application/json

{
    "status":"error",
    "des":"missing name and key tag in JSON ."
}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.19 PUT /users

A PUT call to /users adds a new user.

Protected Yes (Authentication required)

Request

```
PUT /users/ HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Type: application/json; charset=UTF-8
Content-Length: 35

{"name":"harsha","key":"harsha123"}
```

Note

Along with PUT /users request, user can send `PUT /users/<user>/bindings` request to register Gateway with `view only` access rights to the newly added user.

Example

- https://192.168.1.12/users

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

6.3.20 DELETE /users/<user>

A DELETE call to /users/<user> removes specified user.

Protected

Yes (Authentication required)

Request

```
DELETE /users/<user> HTTP/1.1
Host: 192.168.1.12
Content-Type: application/json; charset=UTF-8
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Length: 17
{"name":"harsha"}
```

Examples

- https://192.168.1.12/users/harsha

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

6.3.21 GET /users/<user>/bindings

A GET call to /users/<user>/bindings returns list of access levels configured by administrator.

Protected

Yes (Authentication required)

Request

```
GET /users/<user>/bindings HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/users/user3/bindings

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "bindings":[

{"serial":"ABBXXXXXXXXXXXX","access":"Configure"
},
        {"serial":"ABB_3","access":"Configure"}
    ]
}
```

Description

- bindings [Array of Objects]
  serial [String 17] - Serial number of device.
  access [String] - Access can be 'Configure' or 'View Only'
  Notes
  Gateway bindings can be derived against gateway deviceid.
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.22 PUT /users/<user>/bindings

A PUT call to /users/<user>/bindings configures binding to a device.

Protected

Yes (Authentication required)

Request

```
PUT /users/<user>/bindings HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Type: application/json; charset=UTF-8
Content-Length: 57

{"name":"harsha","serial":"ABB_3","access":"Config-
ure"}
```

Examples

- https://192.168.1.12/users/harsha/bindings

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

6.3.23 DELETE /users/<user>/bindings

A DELETE call to /users/<user>/bindings removes user binding for reqeusted device.

Protected

Yes (Authentication required)

Request

```
DELETE /users/<user>/bindings HTTP/1.1
Content-Type: application/json; charset=UTF-8
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Length: 57

{"name":"harsha","serial":"ABB_3","access":"No Access"}
```

Example

- https://192.168.1.12/users/harsha/bindings

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

6.3.24 GET /meters/firmwareupdatestatus

A GET call to /meters/firmwareupdatestatus returns firmware update status of each meter which is requested for upgrade.

Protected

Yes (Authentication required)

Request

```
GET /meters/firmwareupdatestatus HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/firmwareupdatestatus

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
   "devices": [
        {"serial":"ABB_8", "status":"pending"},
        {"serial":"ABB_8", "status":"success"},
        {"serial":"ABB_8", "status":"failed"}
       ],
   "firmwareupdatecomplete":2
}
```

Description

- devices - array of devices.
  serial [string 17] - serial number of device
  status [string] - 'pending', 'success' or 'failed'
- firmwareupdatecomplete[Int] - Indicates the status of firmware upgrade.
  Values can be:
  0 - update not initiated.
  1 - update is in progress.
  2 - update completed.
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.25 POST /firmware

A POST call to /firmware uploads image file to gateway. Image file can be of meter or gateway. At any time only one image file can be stored in gateway.

Protected

Yes (Authentication required)

Request

```
POST /firmware HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Type:application/json
Content-Length: 1534226
Content-Type: multipart/form-data

 ------WebKitFormBoundary4pEAcghjNsS3M8L5
       Content-Disposition: form-data;
       name="image";
       filename="DSCF5183.JPG"
       Content-Type: image/jpeg



 ------WebKitFormBoundary4pEAcghjNsS3M8L5--
```

Examples

- https://192.168.1.12/firmware

Response

- 200 Success

```
HTTP/1.1 200 OK
     Content-Type: text/html;charset=utf-8
     Content-Length: 20
     Connection: Keep-Alive

     {"status":"success"}
```

Description

Updates firmware image stored in gateway.

Firmware update is a two stage process.

- stage1 - firmware image is uploaded and stored in gateway
- stage2 - user initiates a update request

Note

This resource performs stage1 task.

For stage2, please refer to '[PUT] /firmware'

- 400 Bad Request

```
HTTP/1.1 400 Bad Request
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 57
Content-Type: application/json

{
    "status":"error",
    "des":"missing name and key tag in JSON ."
}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.26 PUT /firmware

A PUT call to /firmware will update the firmware of gateway.

Protected Yes (Authentication required)

Request

```
PUT /firmware/ HTTP/1.1
Host: 192.168.1.12
Authorization: Basic XXXXXXXXXXXXXXX=
```

Example

- http://192.168.1.12/firmware

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: text/html;charset=utf-8
Content-Length: 20
Connection: Keep-Alive

{"status":"success"}
```

6.3.27 PUT /meters/firmware

A PUT call to /meterfirmware updates meter firmware for requested set of meters.

Note

Applicable firmware image should be in gateway. Please refer to "POST /firmware"

Protected

Yes (Authentication required)

Request

```
PUT /meters/firmware HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Type:application/json
Content-Length: 35

 {
   "meters":["ABBXXXXXXXXXX1101"]
 }
```

Description

- meters[Array of strings] - Array of serial numbers
  Max of 10 meters can be sent in per request.
  At any time only one update task can be scheduled.

Examples

- https://192.168.1.12/meters/firmware

Response

- 200 Success

```
HTTP/1.1 200 OK
X-Frame-Options: sameorigin
X-Xss-Protection: 1; mode=block
Content-Type: text/html;charset=utf-8
Content-Length: 20
Connection: Keep-Alive
{"status":"success"}
```

- 400 Bad Request

```
HTTP/1.1 400 Bad Request
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 57
Content-Type: application/json

{
    "status":"error",
    "des":"missing name and key tag in JSON ."
}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.28 GET /parametermapping

A GET call to /parametermapping returns the mapping list to be used as a reference for configuring parameters in other URIs like configuration of complex parameters.

Protected

Yes (Authentication required)

Request

```
GET /parametermapping
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- http://192.168.1.12/parametermapping

Response

- 200 Successful

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
  "map": {
    "1": "ACTIVE_ENERGY_IMPORT_TOTAL",
    "2": "ACTIVE_ENERGY_EXPORT_TOTAL",
    "3": "ACTIVE_ENERGY_IMPORT_L1",
    "4": "ACTIVE_ENERGY_IMPORT_L2",
    "5": "ACTIVE_ENERGY_IMPORT_L3",
    "6": "ACTIVE_ENERGY_EXPORT_L1",
    "7": "ACTIVE_ENERGY_EXPORT_L2",
    "8": "ACTIVE_ENERGY_EXPORT_L3",
    "9": "ACTIVE_ENERGY_IMPORT_TAR1",
    "10": "ACTIVE_ENERGY_IMPORT_TAR2",
    "11": "ACTIVE_ENERGY_IMPORT_TAR3",
    "12": "ACTIVE_ENERGY_IMPORT_TAR4",
    "13": "ACTIVE_ENERGY_NET_TOTAL",
    "14": "ACTIVE_ENERGY_NET_L1",
    "15": "ACTIVE_ENERGY_NET_L2",
    "16": "ACTIVE_ENERGY_NET_L3",
    "17": "REACTIVE_ENERGY_IMPORT_TAR1",
    "18": "REACTIVE_ENERGY_IMPORT_TAR2",
    "19": "REACTIVE_ENERGY_IMPORT_TAR3",
    "20": "REACTIVE_ENERGY_IMPORT_TAR4",
    "21": "REACTIVE_ENERGY_IMPORT_TOTAL",
    "22": "REACTIVE_ENERGY_EXPORT_TOTAL",
    "23": "REACTIVE_ENERGY_IMPORT_L1",
    "24": "REACTIVE_ENERGY_IMPORT_L2",
    "25": "REACTIVE_ENERGY_IMPORT_L3",
    "26": "REACTIVE_ENERGY_EXPORT_L1",
    "27": "REACTIVE_ENERGY_EXPORT_L2",
    "28": "REACTIVE_ENERGY_EXPORT_L3",
    "29": "REACTIVE_ENERGY_NET_TOTAL",
    "30": "REACTIVE_ENERGY_NET_L1",
    "31": "REACTIVE_ENERGY_NET_L2",
    "32": "REACTIVE_ENERGY_NET_L3",
    "33": "APPARENT_ENERGY_IMPORT_TOTAL",
```

```
"34": "APPARENT_ENERGY_EXPORT_TOTAL",
"35": "APPARENT_ENERGY_IMPORT_L1",
"36": "APPARENT_ENERGY_IMPORT_L2",
"37": "APPARENT_ENERGY_IMPORT_L3",
"38": "APPARENT_ENERGY_EXPORT_L1",
"39": "APPARENT_ENERGY_EXPORT_L2",
"40": "APPARENT_ENERGY_EXPORT_L3",
"41": "APPARENT_ENERGY_NET_TOTAL",
"42": "APPARENT_ENERGY_NET_L1",
"43": "APPARENT_ENERGY_NET_L2",
"44": "APPARENT_ENERGY_NET_L3",
"51": "IO_INPUT_CUMULATION_1",
"52": "IO_INPUT_CUMULATION_2",
"53": "IO_INPUT_CUMULATION_3",
"54": "IO_INPUT_CUMULATION_4",
"55": "IO_STORED_INPUT_STATE_1",
"56": "IO_STORED_INPUT_STATE_2",
"57": "IO_STORED_INPUT_STATE_3",
"58": "IO_STORED_INPUT_STATE_4",
"59": "THD_VOLTAGE_L1",
"60": "THD_VOLTAGE_L2",
"61": "THD_VOLTAGE_L3",
"62": "THD_VOLTAGE_L1_L2",
"63": "THD_VOLTAGE_L2_L3",
"64": "THD_VOLTAGE_L1_L3",
"65": "THD_CURRENT_L1",
"66": "THD_CURRENT_L2",
"67": "THD_CURRENT_L3",
"68": "THD_CURRENT_NEUTRAL",
"69": "VOLTAGE_L1",
"70": "CURRENT_L1",
"71": "POWER_FACTOR_L1",
"72": "VOLTAGE_L2",
"73": "CURRENT_L2",
"74": "POWER_FACTOR_L2",
"75": "VOLTAGE_L3",
"76": "CURRENT_L3",
"77": "POWER_FACTOR_L3",
"78": "VOLTAGE_L1_L2",
"79": "VOLTAGE_L2_L3",
"80": "VOLTAGE_L1_L3",
"81": "CURRENT_NEUTRAL",
"82": "POWER_FACTOR_TOTAL",
"83": "NRG_CONVERSION_FACTOR_CURR",
```

```
"84": "NRG_CONVERSION_FACTOR_CO2",
"85": "ACTIVE_ENERGY_EXPORT_TAR1",
"86": "ACTIVE_ENERGY_EXPORT_TAR2",
"87": "ACTIVE_ENERGY_EXPORT_TAR3",
"88": "ACTIVE_ENERGY_EXPORT_TAR4",
"89": "REACTIVE_ENERGY_EXPORT_TAR1",
"90": "REACTIVE_ENERGY_EXPORT_TAR2",
"91": "REACTIVE_ENERGY_EXPORT_TAR3",
"92": "REACTIVE_ENERGY_EXPORT_TAR4",
"93": "ACTIVE_POWER_L1",
"94": "ACTIVE_POWER_L2",
"95": "ACTIVE_POWER_L3",
"96": "ACTIVE_POWER_TOTAL",
"97": "REACTIVE_POWER_L1",
"98": "REACTIVE_POWER_L2",
"99": "REACTIVE_POWER_L3",
"100": "REACTIVE_POWER_TOTAL",
"101": "APPARENT_POWER_L1",
"102": "APPARENT_POWER_L2",
"103": "APPARENT_POWER_L3",
"104": "APPARENT_POWER_TOTAL",
"105": "PHASE_ANGLE_POWER_L1",
"106": "PHASE_ANGLE_POWER_L2",
"107": "PHASE_ANGLE_POWER_L3",
"108": "PHASE_ANGLE_POWER_TOTAL",
"109": "PREVIOUS_VALUES",
"110": "DEMAND",
"111": "LOAD_PROFILE_CH_1",
"112": "ALARM_SETTINGS_1",
"113": "NRG_ENERGY_VALUE_CURR",
"114": "TARIFF_CALENDAR_CONFIG",
"115": "TARIFF_SPECIAL_DAYS_CONFIG",
"116": "IO_SETTING",
"117": "IO_PULSE_LENGTH_1",
"118": "SYSTEM_STATUS_ERROR",
"119": "SYSTEM_STATUS_WARNING",
"120": "SYSTEM_STATUS_INFO",
"121": "LOG_SYSTEM",
"122": "NRG_ENERGY_VALUE_CO2",
"123": "LOG_EVENT",
"124": "LOG_QUALITY",
"137": "SERIAL_NR",
"138": "TYPE_DESIGNATION",
"139": "FIRMWARE_VERSION",
```

```
"140": "HARDWARE_VERSION_0",
"141": "TRAFO_CT_RATIO_NUMERATOR",
"142": "TRAFO_CT_RATIO_DENOMINATOR",
"143": "TRAFO_VT_RATIO_NUMERATOR",
"144": "TRAFO_VT_RATIO_DENOMINATOR",
"146": "CLOCK",
"149": "TARIFF_ACTIVE_DAY_TYPE",
"150": "TARIFF_ACTIVE_SEASON",
"151": "POWER_OUTAGE_TIME",
"152": "POWER_FAIL_COUNTER",
"156": "HDLC_SLAVE_BACKPLANE_1",
"158": "PHASE_ANGLE_VOLTAGE_L1",
"159": "PHASE_ANGLE_CURRENT_L1",
"160": "CURRENT_QUADRANT_L1",
"161": "PHASE_ANGLE_VOLTAGE_L2",
"162": "PHASE_ANGLE_CURRENT_L2",
"163": "CURRENT_QUADRANT_L2",
"164": "PHASE_ANGLE_VOLTAGE_L3",
"165": "PHASE_ANGLE_CURRENT_L3",
"166": "CURRENT_QUADRANT_L3",
"169": "CURRENT_QUADRANT_TOTAL",
"170": "FREQUENCY",
"171": "RSTREG_ACT_ENERGY_IMP",
"172": "RSTREG_ACT_ENERGY_EXP",
"173": "RSTREG_REACT_ENERGY_IMP",
"174": "RSTREG_REACT_ENERGY_EXP",
"175": "RSTREG_ACT_CNTR_IMP",
"176": "RSTREG_ACT_CNTR_EXP",
"177": "RSTREG_REACT_CNTR_IMP",
"178": "RSTREG_REACT_CNTR_EXP",
"179": "HARMONICS_VOLTAGE_L1",
"180": "HARMONICS_VOLTAGE_L2",
"181": "HARMONICS_VOLTAGE_L3",
"182": "HARMONICS_VOLTAGE_L1_L2",
"183": "HARMONICS_VOLTAGE_L2_L3",
"184": "HARMONICS_VOLTAGE_L1_L3",
"185": "HARMONICS_CURRENT_L1",
"186": "HARMONICS_CURRENT_L2",
"187": "HARMONICS_CURRENT_L3",
"188": "HARMONICS_CURRENT_NEUTRAL",
"189": "TARIFF_ACTIVE",
"190": "TARIFF_SOURCE",
"191": "HARDWARE_VERSION_1",
"192": "HARDWARE_VERSION_2",
```

```
        "193": "HARDWARE_VERSION_3",
        "194": "HARDWARE_VERSION_4",
        "195": "HARDWARE_VERSION_5",
        "196": "HARDWARE_VERSION_6",
        "197": "HARDWARE_VERSION_7",
        "198": "MBUS_CLIENT_0",
        "199": "TARIFF_INPUT_CONFIG"
    }
}
```

- 401 Authentication Failure

```
HTTP/1.1 401 Unauthorized
Server: embOS/IP
Accept-Ranges: bytes
Content-Type: text/html
```

- 400 Bad Gateway

```
HTTP/1.1 401 Invalid Request
Server: embOS/IP
Accept-Ranges: bytes
Content-Type: text/html
```

6.3.29 GET /storablequantities

A GET call to /storablequantities returns the list of configurable parameters for storing in meter.

This is a generic list which is used for stored value channel configuration.

Protected Yes (Authentication required)

Request

```
GET /storablequantities
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- http://192.168.1.12/storablequantities

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{

  "pv":
[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,
20,

21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,3
7,38,39,40,

41,42,43,44,51,52,53,54,113,122,85,86,87,88,89,90
,91,92,171,172,173,174],


"lp":[1,2,3,4,5,6,7,8,21,22,23,24,25,26,27,28,33,
34,35,36,37,38,39,40,

51,52,53,54,113,122,69,70,71,72,73,74,75,76,77,78
,79,80,81,82,179,180,
  181,182,183,184,185,186,187,188],


"dm":[51,52,53,54,59,60,61,62,63,64,65,66,67,68,6
9,70,72,73,75,76,78,79,80,
  81,93,94,95,96,97,98,99,100,101,102,103,104],


"ac":[59,60,61,62,63,64,65,66,67,68,69,70,71,72,7
3,74,75,76,77,78,79,80,
  81,82,93,94,95,96,97,98,99,100,101,102,103,104]

}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json
```

```
{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.30 GET /meters/<serial>/associationobjects

A GET call to /meters/<serial>/associationobjects returns list of bitstring and respective access rights for given meter. Readers to note that access rights presented here are nothing to do with access level of gateway. These are access levels as is

in meter firmware. However gateway admin might impose access restrictions up above these for a given user.

Association objects can be used to check if a given parameter is supported by meter or not. Say to know if a meter supports Read Active Export energy, then the following procedure can be followed:

1. Get index for Active Export Energy Total from the following table. In this case it is 2.
2. Check for values with index 2 in result of '/associationobjects'.
3. If Bitstring is '1' that indicates this meter supports active export energy read. If Bitstring is '0' that indicates this meter does not support active export energy read.

Quanity	Index
ACTIVE_ENERGY_IMPORT_TOTAL	1
ACTIVE_ENERGY_EXPORT_TOTAL	2
ACTIVE_ENERGY_IMPORT_L1	3
ACTIVE_ENERGY_IMPORT_L2	4
ACTIVE_ENERGY_IMPORT_L3	5
ACTIVE_ENERGY_EXPORT_L1	6
ACTIVE_ENERGY_EXPORT_L2	7
ACTIVE_ENERGY_EXPORT_L3	8
ACTIVE_ENERGY_IMPORT_TAR1	9
ACTIVE_ENERGY_IMPORT_TAR2	10
ACTIVE_ENERGY_IMPORT_TAR3	11
ACTIVE_ENERGY_IMPORT_TAR4	12
ACTIVE_ENERGY_NET_TOTAL	13
ACTIVE_ENERGY_NET_L1	14
ACTIVE_ENERGY_NET_L2	15
ACTIVE_ENERGY_NET_L3	16
REACTIVE_ENERGY_IMPORT_TAR1	17
REACTIVE_ENERGY_IMPORT_TAR2	18
REACTIVE_ENERGY_IMPORT_TAR3	19
REACTIVE_ENERGY_IMPORT_TAR4	20
REACTIVE_ENERGY_IMPORT_TOTAL	21
REACTIVE_ENERGY_EXPORT_TOTAL	22
REACTIVE_ENERGY_IMPORT_L1	23
REACTIVE_ENERGY_IMPORT_L2	24
REACTIVE_ENERGY_IMPORT_L3	25
REACTIVE_ENERGY_EXPORT_L1	26
REACTIVE_ENERGY_EXPORT_L2	27
REACTIVE_ENERGY_EXPORT_L3	28
REACTIVE_ENERGY_NET_TOTAL	29
REACTIVE_ENERGY_NET_L1	30
REACTIVE_ENERGY_NET_L2	31
REACTIVE_ENERGY_NET_L3	32

APPARENT_ENERGY_IMPORT_TOTAL	33
APPARENT_ENERGY_EXPORT_TOTAL	34
APPARENT_ENERGY_IMPORT_L1	35
APPARENT_ENERGY_IMPORT_L2	36
APPARENT_ENERGY_IMPORT_L3	37
APPARENT_ENERGY_EXPORT_L1	38
APPARENT_ENERGY_EXPORT_L2	39
APPARENT_ENERGY_EXPORT_L3	40
APPARENT_ENERGY_NET_TOTAL	41
APPARENT_ENERGY_NET_L1	42
APPARENT_ENERGY_NET_L2	43
APPARENT_ENERGY_NET_L3	44
IO_PULSE_LENGTH_2	45
IO_PULSE_LENGTH_3	46
IO_PULSE_LENGTH_4	47
IO_PULSE_FREQUENCY_AE_1	48
IO_PULSE_FREQUENCY_AE_2	49
IO_PULSE_FREQUENCY_AE_3	50
IO_INPUT_CUMULATION_1	51
IO_INPUT_CUMULATION_2	52
IO_INPUT_CUMULATION_3	53
IO_INPUT_CUMULATION_4	54
IO_STORED_INPUT_STATE_1	55
IO_STORED_INPUT_STATE_2	56
IO_STORED_INPUT_STATE_3	57
IO_STORED_INPUT_STATE_4	58
THD_VOLTAGE_L1	59
THD_VOLTAGE_L2	60
THD_VOLTAGE_L3	61
THD_VOLTAGE_L1_L2	62
THD_VOLTAGE_L2_L3	63
THD_VOLTAGE_L1_L3	64
THD_CURRENT_L1	65
THD_CURRENT_L2	66
THD_CURRENT_L3	67
THD_CURRENT_NEUTRAL	68
VOLTAGE_L1	69
CURRENT_L1	70
POWER_FACTOR_L1	71
VOLTAGE_L2	72
CURRENT_L2	73
POWER_FACTOR_L2	74
VOLTAGE_L3	75
CURRENT_L3	76

POWER_FACTOR_L3	77
VOLTAGE_L1_L2	78
VOLTAGE_L2_L3	79
VOLTAGE_L1_L3	80
CURRENT_NEUTRAL	81
POWER_FACTOR_TOTAL	82
NRG_CONVERSION_FACTOR_CURR	83
NRG_CONVERSION_FACTOR_CO2	84
ACTIVE_ENERGY_EXPORT_TAR1	85
ACTIVE_ENERGY_EXPORT_TAR2	86
ACTIVE_ENERGY_EXPORT_TAR3	87
ACTIVE_ENERGY_EXPORT_TAR4	88
REACTIVE_ENERGY_EXPORT_TAR1	89
REACTIVE_ENERGY_EXPORT_TAR2	90
REACTIVE_ENERGY_EXPORT_TAR3	91
REACTIVE_ENERGY_EXPORT_TAR4	92
ACTIVE_POWER_L1	93
ACTIVE_POWER_L2	94
ACTIVE_POWER_L3	95
ACTIVE_POWER_TOTAL	96
REACTIVE_POWER_L1	97
REACTIVE_POWER_L2	98
REACTIVE_POWER_L3	99
REACTIVE_POWER_TOTAL	100
APPARENT_POWER_L1	101
APPARENT_POWER_L2	102
APPARENT_POWER_L3	103
APPARENT_POWER_TOTAL	104
PHASE_ANGLE_POWER_L1	105
PHASE_ANGLE_POWER_L2	106
PHASE_ANGLE_POWER_L3	107
PHASE_ANGLE_POWER_TOTAL	108
PREVIOUS_VALUES	109
DEMAND	110
LOAD_PROFILE_CH_1	111
ALARM_SETTINGS_1	112
NRG_ENERGY_VALUE_CURR	113
TARIFF_CALENDAR_CONFIG	114
TARIFF_SPECIAL_DAYS_CONFIG	115
IO_SETTING	116
IO_PULSE_LENGTH_1	117
SYSTEM_STATUS_ERROR	118
SYSTEM_STATUS_WARNING	119
SYSTEM_STATUS_INFO	120

LOG_SYSTEM	121
NRG_ENERGY_VALUE_CO2	122
LOG_EVENT	123
LOG_QUALITY	124
IO_PULSE_FREQUENCY_AE_4	125
IO_PULSE_FREQUENCY_RE_1	126
IO_PULSE_FREQUENCY_RE_2	127
IO_PULSE_FREQUENCY_RE_3	128
IO_PULSE_FREQUENCY_RE_4	129
IO_PULSE_PORT_NBR_1	130
IO_PULSE_PORT_NBR_2	131
IO_PULSE_PORT_NBR_3	132
IO_PULSE_PORT_NBR_4	133
IO_PULSE_ENERGY_TYPE_1	134
IO_PULSE_ENERGY_TYPE_2	135
IO_PULSE_ENERGY_TYPE_3	136
SERIAL_NR	137
TYPE_DESIGNATION	138
FIRMWARE_VERSION	139
TARIFF_INPUT_CONFIG	140
TRAFO_CT_RATIO_NUMERATOR	141
TRAFO_CT_RATIO_DENOMINATOR	142
TRAFO_VT_RATIO_NUMERATOR	143
TRAFO_VT_RATIO_DENOMINATOR	144
CLOCK	146
TARIFF_CALENDAR_CONFIG	147
TARIFF_CALENDAR_CONFIG	148
TARIFF_ACTIVE_DAY_TYPE	149
TARIFF_ACTIVE_SEASON	150
POWER_OUTAGE_TIME	151
POWER_FAIL_COUNTER	152
IO_PULSE_ENERGY_TYPE_4	153
IO_CURRENT_OUTPUT_STATE	154
CLOCK_DST	155
HDLC_SLAVE_BACKPLANE_1	156
PHASE_ANGLE_VOLTAGE_L1	158
PHASE_ANGLE_CURRENT_L1	159
CURRENT_QUADRANT_L1	160
PHASE_ANGLE_VOLTAGE_L2	161
PHASE_ANGLE_CURRENT_L2	162
CURRENT_QUADRANT_L2	163
PHASE_ANGLE_VOLTAGE_L3	164
PHASE_ANGLE_CURRENT_L3	165
CURRENT_QUADRANT_L3	166

phase_angle_voltage_l1-l2	167
phase_angle_voltage_l2-l3	168
CURRENT_QUADRANT_TOTAL	169
FREQUENCY	170
RSTREG_ACT_ENERGY_IMP	171
RSTREG_ACT_ENERGY_EXP	172
RSTREG_REACT_ENERGY_IMP	173
RSTREG_REACT_ENERGY_EXP	174
RSTREG_ACT_CNTR_IMP	175
RSTREG_ACT_CNTR_EXP	176
RSTREG_REACT_CNTR_IMP	177
RSTREG_REACT_CNTR_EXP	178
HARMONICS_VOLTAGE_L1	179
HARMONICS_VOLTAGE_L2	180
HARMONICS_VOLTAGE_L3	181
HARMONICS_VOLTAGE_L1_L2	182
HARMONICS_VOLTAGE_L2_L3	183
HARMONICS_VOLTAGE_L1_L3	184
HARMONICS_CURRENT_L1	185
HARMONICS_CURRENT_L2	186
HARMONICS_CURRENT_L3	187
HARMONICS_CURRENT_NEUTRAL	188
TARIFF_ACTIVE	189
TARIFF_SOURCE	190
HARDWARE_VERSION_0	191
HARDWARE_VERSION_1	192
HARDWARE_VERSION_2	193
HARDWARE_VERSION_3	194
HARDWARE_VERSION_4	195
HARDWARE_VERSION_5	196
HARDWARE_VERSION_6	197
HARDWARE_VERSION_7	198
MBUS_CLIENT_0	199
PREVIOUS_VALUES	201
DEMAND	202
LOAD_PROFILE_CH_1	203
RSTREG_ACT_ENERGY_IMP	204
RSTREG_ACT_ENERGY_EXP	205
RSTREG_REACT_ENERGY_IMP	206
RSTREG_REACT_ENERGY_EXP	207
IO_INPUT_CUMULATION_1	208
IO_INPUT_CUMULATION_2	209
IO_INPUT_CUMULATION_3	210

```
IO_INPUT_CUMULATION_4          211
IO_STORED_INPUT_STATE_1        212
IO_STORED_INPUT_STATE_2        213
IO_STORED_INPUT_STATE_3        214
IO_STORED_INPUT_STATE_4        215

CLOCK                          221
TRAFO_CT_RATIO_NUMERATOR       222
TRAFO_CT_RATIO_DENOMINATOR     223
TRAFO_VT_RATIO_NUMERATOR       224
TRAFO_VT_RATIO_DENOMINATOR     225
ALARM_SETTINGS_1               226
PREVIOUS_VALUES                227
DEMAND                         228
LOAD_PROFILE_CH_1              229
NRG_CONVERSION_FACTOR_CURR     230
NRG_CONVERSION_FACTOR_CO2      231
TARIFF_CALENDAR_CONFIG         232
TARIFF_CALENDAR_CONFIG         233
TARIFF_CALENDAR_CONFIG         234
TARIFF_SPECIAL_DAYS_CONFIG     235
TARIFF_ACTIVE                  236
TARIFF_SOURCE                  237
TARIFF_INPUT_CONFIG            238
IO_CURRENT_OUTPUT_STATE        239
IO_SETTING                     240
CLOCK_DST                   241
```

Note

To Read the parameter refer the index from 1 to 200. To Reset the parameter refer the index from 201 to 220. To configure the parameter refer the index from 221 to 256.

Protected

Yes (Authentication required)

Request

```
GET /meters/<serial>/associationobjects HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/associationobjects

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "associationobjects": [
        1,
        1,
        1,
        1,
        1,
        1,
        .
        .
        .
        .
        .
        .
        .
        .
        1,
        1,
        1
    ],
    "role": "View Only",
    "serial": "ABB_1"
}
```

Description

- associationobjects[array of 256 bits] - It contains array of association bitstring.
- role[string(9)] - Meter access level for active user.
- serial[string(17)] - Meter serial number.
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.31 GET /meters

A GET call to /meters returns list of meters registered with gateway.

Protected Yes (Authentication required)

Request

```
GET /meters HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "alias": "Gateway",
    "location": "ABB Gateway Zone",
    "serial": "ABBXXXXXXXXXXXXX",
    "locations": [
        {
            "location": "Level1",
            "serial": " ",
            "meters": [
                {
                    "alias": "ABB_8",
                    "serial": "ABB_8",
                    "protocol": "3",
                    "address": "17",
                    "status": 1
                },
                {
                    "alias": "ABB_9",
                    "serial": "ABB_9",
                    "protocol": "1",
                    "address": "18",
                    "status": 1
                }
            ]
        },
        {
            "location": "Level2",
            "serial": " ",
            "meters": [
                {
                    "alias": "ABB_6",
                    "serial": "ABB_6",
                    "protocol": "1",
                    "address": "12",
                    "status": 1
```

```
                    },
                    {
                        "alias": "ABB_7",
                        "serial": "ABB_7",
                        "protocol": "2",
                        "address": "13",
                        "status": 1
                    }
                ]
            }
        ]
    }
```

Description

- alias [String 21] - Alias of gateway.
- location [String 51] - Location of gateway.
- serial [String 17] - Serial number of gateway.
- locations [Array] - Array of location objects.
  location [String 51] - Location name
  serial [String 17] - Optional value of main meter applicable in case of sub-metering.
  meters [Array] - Array of meter objects.
  alias [String 21] - Alias name of the meter.
  serial [String 17] - Serial number of meter.
  protocol [Int] - Backplane protocol over which meter is registered.
  M-Bus Wired (1)
  M-Bus IR(2)
  EQ bus(3)
  Wireless(4)
  address [Int] - Address of meter for this protocol. In case of wireless, this would be the Mac address
  status [Int] - Status of connectivity. Offline(-1), Registered(0), Online(1)
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.32 POST /meters/<serial>

A POST call to /meters/<serial> updates device alias name and location configuration.

Protected

Yes (Authentication required)

Request

```
POST /meters/ABB_8 HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Type:application/json
Content-Length: 56

{"alias":"ABB_8", "location":"location1", "serial":"ABB_8"}
```

Description

- alias[String 21] - Gateway alias name
- location[String 51] - Gateway location name
- serial[String 17] - Gateway serial number

Examples

- https://192.168.1.12/meters/ABB_2

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 400 Bad Request

```
HTTP/1.1 400 Bad Request
Server: embOS/IP

Accept-Ranges: bytes
Content-Length: 57
Content-Type: application/json

{
    "status":"error",
    "des":"JSON parse error."
}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.33 PUT /meters

A PUT call to /meters registers a meter with gateway.

Protected

Yes (Authentication required)

Request

```
PUT /meters HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Type:application/json
Content-Length: 56

{"alias":"ABB_8", "location":"location2", "serial":"ABB_8"}
```

- ***Description***
- alias[String 21] - Gateway alias name
- location[String 51] - Gateway location name
- serial[String 17] - Gateway serial number

Examples

- https://192.168.1.12/meters

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 400 Bad Request

```
HTTP/1.1 400 Bad Request
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 57
Content-Type: application/json

{
    "status":"error",
    "des":"JSON parse error."
}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.34 DELETE /meters/<serial>

A DELETE call to /meters/ABB_8 deregisters a meter with gateway.

Protected

Yes (Authentication required)

Request

```
DELETE /meters HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Type:application/json
Content-Length: 56

{"alias":"ABB_8", "location":"location2", "serial":"ABB_8"}
```

Description

- alias[String 21] - Gateway alias name
- location[String 51] - Gateway location name
- serial[String 17] - Gateway serial number

Example

- https://192.168.1.12/meters/ABB_2

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 400 Bad Request

```
HTTP/1.1 400 Bad Request
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 57
Content-Type: application/json

{
    "status":"error",
    "des":"JSON parse error."
}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.35 GET /meters/scanned

A GET call to /meters returns list of scanned meters.

Protected

Yes (Authentication required)

Request

```
GET /meters/scanned HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Example

- https://192.168.1.12/meters/scanned

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "scanresult": [
        {
            "token": 1337581086,
            "serial": "ABB_3",
            "protocol": "3",
            "address": "17"
        }
    ],
    "scancomplete": 0
}
```

Description

- scanresult [Array of Objects] - Array of scanned meter objects
  token [Int] - token for scan.
  serial [String 17] - Serial Number of scanned meter
  protocol [Int] - Protocol
  M-Bus Wired (1)
  M-Bus IR(2)
  EQ bus(3)
  Wireless(4)
  address [String] - Address of meter for this protocol. In case of wireless, this would be the Mac address
- scancomplete [Int] - Complete(1) or In-progress(2).
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.36 POST /meters/scanned/<token>

A POST call to /meters/scanned/<token> triggers a new scan request if another one is not in progress.

Protected Yes (Authentication required)

Request

```
POST /meters/scanned/token HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Type:application/json
Content-Length: 278

{
    "zigbee": 3,
    "eqbus": {
        "start": 17,
        "end": 49
    },
    "mbus-wired": {
        "baudrate": 2400,
        "devices": [1, 2, 3]
    },
    "mbus-ir": {
        "baudrate": 4800,
        "primary": {}
    },
    "token": 1342526556
}
```

Description

- zigbee : Maximum of 32 devices can be specified
- eqbus : HDLC addresses can be specified in two ways. In the form of start(17) and end(49) or a range ('devices') of up to 10 addresses.
- mbus-wired : MBus addresses can be specified in two ways:
  In Primary:
  start(1) , end(16) and baudrate. OR
  range (devices) of up to 10 addresses and baudrate.
- Secondary and baudrate:
  No input needs to be provided for the secondary scan. It makes a search in which it will find all slaves regardless of their serial number, manufacturer, protocol version and medium.
- mbus-ir : Mbus legacy addresses can be specified in two ways:
  Primary and baudrate.
  Secondary and baudrate.

Scanning can be specified for any set of protocols. Above request scans for all protocols.

However following payloads are also valid.

Here mbus-wired and mbus-ir are specified in secondary addressing mode:

{ "mbus-wired": { "baudrate": 19200, "secondary": {} }, "mbus-ir": { "baudrate": 4800, "secondary": {} }, "token": 1342527820 }

Here mbus-wired and mbus-ir are specified in primary addressing mode:

{ "mbus-wired": { "baudrate": 38400, "start": 1, "end": 16 }, "mbus-ir": { "baudrate": 9600, "primary": {} }, "token": 1342530975 }

A sum of 32 devices out of all protocols can be scanned per request. Only one scan can be scheduled at any given time.

Examples

- https://192.168.1.12/meters/scanned/1342526556

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 400 Bad Request

```
HTTP/1.1 400 Bad Request
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 57
Content-Type: application/json

{
    "status":"error",
    "des":"JSON parse error."
}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.37 DELETE /meters/scanned

A DELETE call to /meters/scanned deletes requested devices from scan list.

Protected Yes (Authentication required)

Request

```
DELETE /meters/scanned HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Type:application/json
Content-Length: 59

{
    "deletescanned": [
     {"serial": "ABB_4"},
     {"serial": "ABB_2"}
   ]
}
```

Description

- deletescanned [Array of objects] - Array of scanned meter objects
- serial[String 17] : Serial Number of scanned meter.

Note

Request sent without any payload (empty) deletes all scanned meters. Maximum meters specified in payload should not exceed 15.

Examples

- https://192.168.1.12/meters/scanned

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "deletescanned": [
     {"serial": "ABB_4","status": 1},
     {"serial": "ABB_2","status": -1}
   ]
}
```

Description

deletescanned [Array of objects] - Array of scanned meter objects

- serial[String 17] : Serial Number of scanned meter.
- status[Int32] : Indicates the status of meter deleted.
  1 : meter deleted successfully from scan list
  -1 : meter is not found in the scan list
- 400 Bad Request

```
HTTP/1.1 400 Bad Request
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 57
Content-Type: application/json

{
    "status":"error",
    "des":"JSON parse error."
}
```

6.3.38 GET /permittedmeters

A GET call to /permittedmeters returns list of permitted wireless meters.

Protected

Yes (Authentication required)

Request

```
GET /permittedmeters HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/permittedmeters

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "meters": [
        "ABBXXXXXXXXXXXXX1",
        "ABBXXXXXXXXXXXXX2",
        "ABBXXXXXXXXXXXXX3"
    ]
}
```

Description

- meters [Array of serial numbers (string 17)]

Note

At any time max of 25 serial numbers can be specified

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.39 PUT /permittedmeters

A PUT call to /permittedmeters adds a new meter to list.

Protected

Yes (Authentication required)

Request

```
PUT /permittedmeters HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Type:application/json
Content-Length: 23

{"meters":["ABBXXXXXXXXXXXXX1", "ABBXXXXXXXXXXXXX2"]}
```

Description

- Meters is an array of serial numbers of zigbee devices.

Examples

- https://192.168.1.12/permittedmeters

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 400 Bad Request

```
HTTP/1.1 400 Bad Request
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 57
Content-Type: application/json

{
    "status":"error",
    "des":"missing datetime tag in JSON ."
}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.40 DELETE /permittedmeters

A DELETE call to /permittedmeters deletes meter from list.

Protected

Yes (Authentication required)

Request

```
DELETE /permittedmeters HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Type:application/json
Content-Length: 23

{
    "meters":[
      "ABBXXXXXXXXXXXXX1",
      "ABBXXXXXXXXXXXXX2"]
}
```

Description

- meters [array of serial numbers (string 17)]

Examples

- https://192.168.1.12/permittedmeters

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 400 Bad Request

```
HTTP/1.1 400 Bad Request
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 57
Content-Type: application/json

{
    "status":"error",
    "des":"missing datetime tag in JSON ."
}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.41 GET /meters/<serial>/datetime

A GET call to /meters/<serial>/datetime returns datetime settings of meter.

Protected

Yes (Authentication required)

Request

```
GET /meters/<serial>/datetime HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/datetime

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "datetime": "2012-08-05T01: 11: 31",
    "dayofweek": 7,
    "dst": 1,
    "dstenable": true,
    "dstend": {
        "month": 10,
        "day": 1,
        "week": 1,
        "hour": 1
    },
    "dststart": {
        "month": 6,
        "day": 1,
        "week": 1,
        "hour": 2
    },
    "status": "set"
}
```

Description

- datetime [String] - Meter Datetime.
- dayofweek [Int] - DayOfWeek[Monday-1 to Sunday-7].
- dst [Int] - Set to 1 if current datetime lies between dststart and dstend, else set to 0
- dstenable [Boolean] - True - DST enabled, False - DST disabled.
- dstend [String] - DST end date time.
- dststart [String] - DST start date time.
- status [String(8)] - ClockStatus.["not set","set"].
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.42 POST /meters/<serial>/datetime

A POST call to /meters/<serial>/datetime updates date time of meter and DST Configuration.

Protected Yes (Authentication required)

Request

```
POST /meters/ABB_3/datetime HTTP/1.1
   Host: 192.168.1.12
   Content-Length: 171
   Authorization: Basic XXXXXXXXXXXXXXX=

   {
       "datetime": "2012-08-05T01:11:31",
       "dstenable": true,
       "dststart":
{"month":6,"day":1,"week":1,"hour":2},
     "dstend": {"month":10,"day":1,"week":1,"hour":1}
   }
```

Description

It is possible to do in the following ways also:

- POST only datetime.
- POST datetime and disable the DST configuration by setting dstenable to false
- POST datetime, dstenable, dststart, dstend

Note

Thus if 'dstenable' is set to 'true', then the user should also send 'dststart' and 'dstend'.

If 'dstenable' is set to 'false', then 'dststart', 'dstend' will be ignored.

Examples

- https://192.168.1.12/meters/ABB_8/datetime

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.43 GET /meters/<serial>/info

A GET call to /meters/<serial>/info returns general information of meter.

Protected

Yes (Authentication required)

Request

```
GET /meters/<serial>/info HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/info

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "manufacturing": {
        "serial": "1231",
        "label": "A44 514-100",
        "firmwareversion": "A0.0.0.0"
    },
    "ratio": {
        "ctdenum": "1",
        "ctnum": "1",
        "vtdenum": "1",
        "vtnum": "1"
    },
    "conversionfactor": {
        "co2": [
            1.123,
            "kg/kWh"
        ],
        "currency": [
            100.789,
            "curr/kWh"
        ]
    },
    "power": {
        "powerfailcounter": "4",
        "poweroutagetime": "0"
    }
}
```

Description

- manufacturing [Object] - Object of manufacturing information.
  serial [String 17] - Serial number of meter.
  label [String 11] - Meter label like A44 412-100. For details please refer to "**Annexure1**" section.
  firmwareversion [String 17] - Firmware version like A0.0.0.0
- ratio [Object] - Object of transformer ratio settings.
  ctnum [Int] - CT numerator.
  ctdenum [Int] - CT de-nominator.
  vtnum [Int] - VT numerator.
  vtdenum [Int] - VT de-nominator.
- power [Object] - Object of power outage information.
  poweroutagetime [Int] - Duration of power outage time in seconds.
  powerfailcounter [Int] - General counter of powerfail.
- conversion factor [Object] - Object of conversion factors.
  co2 [Array] - co2 conversion factor and unit.
  currency [Array] - currency conversion factor and unit.
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.44 GET /meters/<serial>/energy/conversionfactor

A GET call to /meters/<serial>/conversionfactor returns CO2 and Currency conversion factor.

Protected Yes (Authentication required)

Request

```
GET /meters/<serial>/energy/conversionfactor HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/conversionfactor

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "conversionfactor": {
        "co2": [
            1.123,
            "kg/kWh"
        ],
        "currency": [
            100.789,
            "curr/kWh"
        ]
    }
}
```

Description

- co2 [Array] - co2 conversion factor and unit.
- currency [Array] - currency conversion factor and unit.
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.45 POST /meters/<serial>/energy/conversionfactor

A GET call to /meters/<serial>/conversionfactor updates co2 and currency conversion factor.

Protected Yes (Authentication required)

Request

```
POST /meters/ABB_3/energy/conversionfactor HTTP/1.1
   Host: 192.168.1.12
   Content-Length: 53
   Authorization: Basic XXXXXXXXXXXXXXX=

   {
       "conversionfactor": {
           "co2": 1.123,
           "currency": 100.789
       }
   }
```

Description

- co2 [Double/Int] - co2 conversion factor value should be less than 9.999.
- currency [Double/Int] - currency conversion factor value should be less than 999999.000.

Note

- User can update either co2 or currency or both.
- Here, values are specified as per the following units, co2=>"kg/kWh", currency =>"curr/kWh".

Examples

- https://192.168.1.12/meters/ABB_8/conversionfactor

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.46 GET /meters/<serial>/hardwareversion

A GET call to /meters/<serial>/hardwareversion returns version numbers of various components of meter (main+I/O+RS-485).

Protected

Yes (Authentication required)

Request

```
GET /meters/<serial>/hardwareversion HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/hardwareversion

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "version": [
        "ABCDEFGHIJKLMNOP_0",
        "ABCDEFGHIJKLMNOP_1",
        "ABCDEFGHIJKLMNOP_2",
        "ABCDEFGHIJKLMNOP_3",
        "ABCDEFGHIJKLMNOP_4",
        "ABCDEFGHIJKLMNOP_5",
        "ABCDEFGHIJKLMNOP_6",
        "ABCDEFGHIJKLMNOP_7"
    ]
}
```

Description

Version is an array of version numbers up to 8 components. Each version number is [string18].

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.47 GET /meters/<serial>/mbusinfo

A GET call to /meters/<serial>/mbusinfo returns M-Bus meter related information.

Protected

Yes (Authentication required)

Request

```
GET /meters/<serial>/mbusinfo HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/mbusinfo

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "manufacturer": "ABB",
    "devicetype": 2,
    "addressingmode": 1,
    "version": 32,
    "id": 1234
}
```

Description

- manufacturer [string 3] : "ABB",
- medium [int]
  Other: 00
  Oil: 01
  Electricity: 02
  Gas: 03
  Heat: 04
  Steam: 05
  Warm Water (30°C-90°C): 06
  Water: 07
  Heat Cost Allocator: 08
  Compressed Air: 09
  Cooling load meter (Volume measured at return temperature: outlet): 0A
  Cooling load meter (Volume measured at flow temperature: inlet): 0B
  Heat (Volume measured at flow temperature: inlet): 0C
  Heat / Cooling load meter: 0D
  Bus / System component: 0E
  Unknown Medium: 0F
  Hot water (≥90°C): 15
  Cold Water: 16
  Dual register (hot/cold) Water: 17
  Pressure: 18
  A/D Converter: 19
- addressingmode [int]
  Primary: Any value 0-250 or 0xFF
  Secondary: 0xFD
- version [int]: version number
- id [int]: Four digits
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.48 POST /meters/<serial>/transformersettings

A POST call to /meters/<serial>/transformersettings updates transformer settings of meter.

Protected

Yes (Authentication required)

Request

```
POST /meters/ABB_3/transformersettings HTTP/1.1
   Host: 192.168.1.12
   Content-Length: 54
   Authorization: Basic XXXXXXXXXXXXXXX=

   {"ctnum":"1","ctdenum":"1","vtnum":"1","vtde-
num":"1"}
```

Description

- ctnum [Int] - CT numerator
- ctdenum [Int] - CT de-nominator
- vtnum [Int] - VT numerator
- vtdenum [Int] - VT de-nominator

Examples

- https://192.168.1.12/meters/ABB_8/transformersettings

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
```

```
}
```

6.3.49 GET /meters/<serial>/status

A GET call to /meters/<serial>/status returns communication status of requested meter.

Protected

Yes (Authentication required)

Request

```
GET /meters/<serial>/status HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Example

- https://192.168.1.12/meters/ABB_8/status

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{ "serial":1238, "label": "A44 512-100"  }
```

Note

This resource can be used to determine communication status of meter. A positive response (200) indicates healthy status.

Also it can be used to detect meter type out of label information.

Description

- serial [String 17] - Serial number of meter.
- label [String 11] - Meter label like A44 412-100. For details please refer to "**Annexure1**" section.
  1 - Single phase direct connected
  2 - Single phase indirect connected
  3 - Three phase direct connected
  4 - Three phase indirect connected
  Type of meter
  0 - Iron
  1 - Steel
  2 - Bronze
  3 - Silver
  4 - Gold
  5 - Platinum
  1 - Class 1.0
  2 - RS 485 Port
  1 - IEC approved + MID approved and verified
  0 - ABB standard version
  0 - ABB standard version
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.50 GET /meters/<serial>/statusflags

A GET call to /meters/<serial>/statusflags returns status flags of requested meter.

Protected Yes (Authentication required)

Request

```
GET /meters/<serial>/statusflags HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/statusflags

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "errors": [
        "ABB Specific Str 2",
        "ABB Specific Str 3"
    ],
    "warnings": [
        "NEG_POW Element2 "
    ],
    "informationflags": [
        "Alarm6 Active",
        "Alarm7 Active"
    ]
}
```

Note

Actual response may be different than documented here.

Description

- errors [Array of String] - Object of active errors. Array may include:
  "Audit Log"
  "Program Crc"
  "Persistent Storage"
  "Rtc Circuit"
  "Acref"
  "Temp Sensor"
  "ABB Specific Str 1"
  "ABB Specific Str 2"
  "ABB Specific Str 3"
  "ABB Specific Str 4"
  "ABB Specific Str 5"
  "ABB Specific Str 6"
  "ABB Specific Str 7"
  "ABB Specific Str 8"
- warnings [Array of String] - Object of active warnings. Array may include:
  "U1 Missing"
  "U2 Missing"
  "U3 Missing"
  "Phase Conn Neutral"
  "NEG_POW Element1"
  "NEG_POW Element2"
  "NEG_POW Element3"
  "Neg Tot Pow"
  "Frequency"
  "External Input"
  "Date Not Set"
  "Time Not SET"
- informationflags [Array of String] - Object of active information flags. Array may include:
  "Alarm1 Active"
  "Alarm2 Active"
  "Alarm3 Active"
  "Alarm4 Active"
  "Alarm5 Active"
  "Alarm6 Active"
  "Alarm7 Active"
  "Alarm8 Active"
  "Alarm9 Active"

"Alarm10 Active"
"Alarm11 Active"
"Alarm12 Active"
"Alarm13 Active"
"Alarm14 Active"
"Alarm15 Active"
"Alarm16 Active"
"Alarm17 Active"
"Alarm18 Active"
"Alarm19 Active"
"Alarm20 Active"
"Alarm21 Active"
"Alarm22 Active"
"Alarm23 Active"
"Alarm24 Active"
"Alarm25 Active"
"Pulses Merged "
"Powerfail"

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.51 GET /meters/<serial>/events/<datetime>/<count>

A GET call to /meters/<serial>/events/<datetime>/<count> returns events logged in requested meter.

Protected Yes (Authentication required)

Request

```
GET /meters/<serial>/events/<datetime>/<count> HTTP/
1.1

Authorization: Basic XXXXXXXXXXXXXXX=
```

Parameters

- serial [String 17] - Serial number of the meter.
- datetime [DateTime] - Events until a date time. eg. 2012-05-16T13:54:00.
- count [Int, Optional] - Default value 100. Number of events requested. Max value can be 100.

Example

- https://192.168.1.12/meters/ABB_8/events/2012-05-15T13:43:00
- https://192.168.1.12/meters/ABB_8/events/2012-05-15T13:43:00/25

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "system": [
        {
            "datetime": "2011-12-19T10:12:00",
            "duration": 0,
            "category": 2,
            "logid": 1001
        },
        {
            "datetime": "2011-12-18T10:12:00",
            "duration": 0,
            "category": 2,
            "logid": 1003
        }
    ],
    "quality": [
        {
            "datetime": "2011-12-19T10:12:00",
            "duration": 0,
            "category": 2,
            "logid": 1001
        },
        {
            "datetime": "2011-12-18T10:12:00",
            "duration": 0,
            "category": 2,
            "logid": 1003
        }
    ],
    "event": [
        {
            "datetime": "2011-12-19T10:12:00",
            "duration": 0,
            "category": 2,
            "logid": 1001
```

```
            },
            {
                "datetime": "2011-12-18T10:12:00",
                "duration": 0,
                "category": 2,
                "logid": 1003
            }
        ]
    }
```

Description

Three types of events are logged in system and each type of event is returned in reverse chronological order.

system - contains system related array of objects

quality - contains quality related array of objects

event - contains event related array of objects

{duration,datetime,category,logid}

- category[Int] : Category of the event
  1 # Exception
  2 # Error
  3 # Warning
  8 # Information
- logid[Int] : Id of log structure as per following table

```
 Error

 40     AUDIT_LOG_ERROR            Audit log error
 41      PROGRAM_CRC_ERROR            Program CRC
error
 42      PERSISTENT_STORAGE_ERROR     Persistent
memory storage error
 43     ABB_SPECIFIC_STR_1_ERROR    ABB sepcific
STR 1 error
 44     ABB_SPECIFIC_STR_2_ERROR    ABB sepcific
STR 2 error
 45     ABB_SPECIFIC_STR_3_ERROR    ABB sepcific
STR 3 error
 46     ABB_SPECIFIC_STR_4_ERROR    ABB sepcific
STR 4 error
 47     ABB_SPECIFIC_STR_5_ERROR    ABB sepcific
STR 5 error
 48     ABB_SPECIFIC_STR_6_ERROR    ABB sepcific
STR 6 error
 49     ABB_SPECIFIC_STR_7_ERROR    ABB sepcific
STR 7 error
 50     ABB_SPECIFIC_STR_8_ERROR    ABB sepcific
STR 8 error
 51      ACREF_ERROR                 ACREF Error
 52      TEMP_SENSOR_ERROR           Temperature
sensor error
 53      RTC_CIRCUIT_ERROR           Real Time
Clock circuit error
```

```
Warning

 1000    U1_MISSING_WARNING           Phase 1 is
missing warning
 1001    U2_MISSING_WARNING           Phase 2 is
missing warning
 1002    U3_MISSING_WARNING           Phase 3 is
missing warning
 1003    PHASE_CONN_NEUTRAL_WARNING   Neutral
warning
 1004    NEG_POW_ELEMENT_1_WARNING    Negative
power of Element 1 warning
 1005    NEG_POW_ELEMENT_2_WARNING    Negative
power of Element 2 warning
 1006    NEG_POW_ELEMENT_3_WARNING    Negative
power of Element 3 warning
 1007   NEG_TOT_POW_WARNING        Negative total
power warning
 1008    FREQUENCY_WARNING            Frequency
warning
 1009   EXTERNAL_INPUT_WARNING     External Input
warning
 1010    DATE_NOT_SET_WARNING         Date is not
set warning
 1011    TIME_NOT_SET_WARNING         Time is not
set warning

 Information

 2013   ALARM1_ACTIVE_INFO         Alarm 1 Active
information
 2014   ALARM2_ACTIVE_INFO         Alarm 2 Active
information
 2015   ALARM3_ACTIVE_INFO         Alarm 3 Active
information
 2016   ALARM4_ACTIVE_INFO         Alarm 4 Active
information
 2017   ALARM5_ACTIVE_INFO         Alarm 5 Active
information
 2018   ALARM6_ACTIVE_INFO         Alarm 6 Active
information
 2019   ALARM7_ACTIVE_INFO         Alarm 7 Active
information
 2020   ALARM8_ACTIVE_INFO         Alarm 8 Active
information
 2021   ALARM9_ACTIVE_INFO         Alarm 9 Active
```

```
information
 2022    ALARM10_ACTIVE_INFO          Alarm 10
Active information
 2023    ALARM11_ACTIVE_INFO          Alarm 11
Active information
 2024    ALARM12_ACTIVE_INFO          Alarm 12
Active information
 2025    ALARM13_ACTIVE_INFO          Alarm 13
Active information
 2026    ALARM14_ACTIVE_INFO          Alarm 14
Active information
 2027    ALARM15_ACTIVE_INFO          Alarm 15
Active information
 2028    ALARM16_ACTIVE_INFO          Alarm 16
Active information
 2029    ALARM17_ACTIVE_INFO          Alarm 17
Active information
 2030    ALARM18_ACTIVE_INFO          Alarm 18
Active information
 2031    ALARM19_ACTIVE_INFO          Alarm 19
Active information
 2032    ALARM20_ACTIVE_INFO          Alarm 20
Active information
 2033    ALARM21_ACTIVE_INFO          Alarm 21
Active information
 2034    ALARM22_ACTIVE_INFO          Alarm 22
Active information
 2035    ALARM23_ACTIVE_INFO          Alarm 23
Active information
 2036    ALARM24_ACTIVE_INFO          Alarm 24
Active information
 2037    ALARM25_ACTIVE_INFO          Alarm 25
Active information

 2054    PULSES_MERGED_INFO          Pulses Merged
information
 2055   POWERFAIL_INFO            Powerfail Information
```

- datetime[String]: timestamp when event occurred
- duration[Int]: duration of an event persists in seconds
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json
```

```
{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.52 GET /meters/<serial>/alarms/configuration

A GET call to /meters/<serial>/alarms/configuration returns meter alarm configuration.

Protected Yes (Authentication required)

Request

```
GET /users HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/alarms/configuration

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "alarmconfig": [
        {
            "id": 1,
            "quantity": 75,
            "onthreshold": 310,
            "offthreshold": 0,
            "ondelay": 240,
            "offdelay": 0,
            "log": 4000,
            "output": 5000
        },
        {
            "id": 2,
            "quantity": 70,
            "onthreshold": -1,
            "offthreshold": -1,
            "ondelay": 2,
            "offdelay": 0,
            "log": 5,
            "output": 10
        },

        {
            "id": 20
        },

        ......
        {
            "id": 25,
            "quantity": 72,
            "onthreshold": 0,
            "offthreshold": 0,
            "ondelay": 0,
            "offdelay": 0,
```

```
                "log": 1000,
                "output": 1000
            }
        ]
}
```

Description

- id[Int]: Alarm number(1 to 25)
- quantity[Int]: quantity index as per parametermapping
- onthreshold[Int]: threshold level
- offthreshold[Int]: threshold level
- ondelay[Int]: on delay in secons
- offdelay[Int]: off delay in seconds
- log[Bool]: boolean says, log is true or false
- output[Int]: outport number (1 to 4)

Notes

If a channel is inactive, then it contains “id” parameter only. For example: `{ "id": 1 }`

400 Bad Request

```
HTTP/1.1 400 Bad Request
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"Invalid Request."
}
```

- 502 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.53 POST /meters/<serial>/alarms/configuration

A POST call to /meters/<serial>/alarms/configuration updates meter alarm configuration.

Note

Prior IO configuration is required to be able to configure any of the outputs.

Protected Yes (Authentication required)

Request

```
POST /meters/&lt;serial>/alarms/configuration HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Type:application/json
Content-Length: 197

{
 "alarmconfig":
     {
       "id": 25,
       "quantity": 70,
       "onthreshold": 0,
       "offthreshold": 0,
       "ondelay": 0,
       "offdelay": 0,
       "log": true,
      "output": 1
     }
 }
```

Description

- id[Int]: Alarm number(1 to 25
- quantity[Int]: quantity index as per parametermapping
- onthreshold[Int]: threshold level
- offthreshold[Int]: threshold level
- ondelay[Int]: on delay in secons
- offdelay[Int]: off delay in seconds
- log[Bool]: boolean says, log is true or false
- output[Int]: outport number (1 to 4)

Note

To make a channel "inactive", the request should skip all parameters except id as shown below:

```
{
 "alarmconfig":
      {
        "id": 25,
      }
 }
```

Examples

- https://192.168.1.12/meters/ABB_8/alarms/configuration

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 400 Bad Request

```
HTTP/1.1 400 Bad Request
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 57
Content-Type: application/json

{
     "status":"error",
     "des":"missing name and key tag in JSON ."
}
```

- 502 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
     "status":"error",
     "des":"received invalid response from COSEM."
```

```
}
```

6.3.54 GET /meters/<serial>/energy/<type>

A GET call to /meters/<serial>/energy/<type> returns realtime energy values of requested meter.

Each component of energy like active, reactive and apparent energies are measured separately.

User needs to obtain each value with separate requests by specifying the type accordingly.

Also meters measure import as well as export energy.

Energy consumed from is measured as import energy.

Energy exported to utility is measured as export energy.

Differential value is measured as net energy. Net energy could be negative when export is more than import.

For simple cases total energy would be a choice.

Protected

Yes (Authentication required)

Request

```
GET /meters/<serial>/energy/<type>/<mode> HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Parameters

- type - 'active', 'reactive', 'apparent'

Note

Not all meter types support reactive and apparent measurements. Please refer to product manual for list of supported parameters.

Examples

- https://192.168.1.12/meters/ABB_8/energy/active
- https://192.168.1.12/meters/ABB_8/energy/reactive
- https://192.168.1.12/meters/ABB_8/energy/apparent

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "importenergy": {
        "l1": [
            "1758307.29",
            "kWh"
        ],
        "l2": [
            "12718357.27",
            "kWh"
        ],
        "l3": [
            "77587522.36",
            "kWh"
        ],
        "t1": [
            "0.16",
            "kWh"
        ],
        "t2": [
            "0.00",
            "kWh"
        ],
        "t3": [
            "0.00",
            "kWh"
        ],
        "t4": [
            "0.01",
            "kWh"
        ],
        "total": [
            "1271.31",
            "kWh"
        ]
    },
```

```
"exportenergy": {
    "l1": [
        "77587522.36",
        "kWh"
    ],
    "l2": [
        "12718357.27",
        "kWh"
    ],
    "l3": [
        "77583.27",
        "kWh"
    ],
    "t1": [
        "2.36",
        "kWh"
    ],
    "t2": [
        "0.00",
        "kWh"
    ],
    "t3": [
        "0.00",
        "kWh"
    ],
    "t4": [
        "2.36",
        "kWh"
    ],
    "total": [
        "12718.36",
        "kWh"
    ]
},
"netenergy": {
    "l1": [
        "75824875.12",
        "kWh"
    ],
    "l2": [
        "-12443117.40",
        "kWh"
    ],
    "l3": [
```

```
                "-77580916.94",
                "kWh"
            ],
            "total": [
                "-10959959.88",
                "kWh"
            ]
        }
    }
```

Description

- exportenergy [Object] - Object of export energy.
  lx [Array] - line export energy and unit [x can be 1 to 3].
  tx [Array] - line export tariff energy and unit [x can be 1 to 4].
  total [Array] - total export energy and unit.
- importenergy [Object] - Object of import energy.
  lx [Array] - line import energy and unit [x can be 1 to 3].
  tx [Array] - line import tariff energy and unit [x can be 1 to 4].
  total [Array] - total import energy and unit.
- netenergy [Object] - Object of net energy.
  lx [Array] - line net energy and unit [x can be 1 to 3].
  total [Array] - total net energy and unit.
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.55 GET /meters/<serial>/energy/<type>/<mode>

A GET call to /meters/<serial>/energy/<type>/<mode> returns converted energy values of requested meter.

Meters log energy as measured on lines (kWh) and their equivalent values in converted format. Conversions can be in terms of CO2 (carbon emission) or Local Currency.

Energy conversion types:

- co2 : contains active import energy
- currency: contains active import energy

Protected

Yes (Authentication required)

Request

```
GET /meters/<serial>/energy/<type>/<mode> HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Parameters

- type - should be 'active' only
- mode - 'co2' or 'currency'

Examples

- https://192.168.1.12/meters/ABB_8/energy/active/co2
- https://192.168.1.12/meters/ABB_8/energy/active/currency

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "importenergy": {
        "total": [
            "0.00",
            "KG"
        ]
    }
}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

Description

- importenergy[object] - total import energy
  total [Array] -total import energy and unit.
  JSON response is the same for active/co2 and active/currency energy except unit.

6.3.56 GET /meters/<serial>/energy/resettable

A GET call to /meters/<serial>/energy/resettable returns resettable energy values of requested meter.

Meters are capable of maintaining resettable registers. These can be reset as needed by user. Reset counter is incremented each time these registers are reset. Resettable energy could include active import, active export, reactive import and reactive export energies.

Protected

Yes (Authentication required)

Request

```
GET /meters/<serial>/energy/resettable HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/energy/resettable

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
"resettableenergy": {
    "activeexport": [
        "12718359.562",
        "Wh"
    ],
    "activeimport": [
        "1758397.363",
        "Wh"
    ]
  }
}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

Description

resettable energy: combination of active import and export energies

- activeimport[array] - value of active import and unit
- activeexport[array] - value of active export and unit

6.3.57 POST /meters/<serial>/energy/resettable

A POST call to /meters/<serial>/energy/resettable clears resettable energy register values of requested meter.

Reset counter is incremented each time resettable energy registers are reset.

User can reset any one or all of following registers

1. activeimport
2. activeexport
3. reactiveimport
4. reactiveexport

Protected Yes (Authentication required)

Request

```
POST /meters/<serial>/energy/resettable HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

```
{
"resettableenergy": {
    "activeimport": true,
    "activeexport": true,
    "reactiveimport": true,
    "reactiveexport": true
  }
}
```

Examples

- https://192.168.1.12/meters/ABB_8/energy/resettable

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "status":"success"
}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.58 GET /meters/<serial>/energy/resetcounter

A GET call to /meters/<serial>/energy/resetcounter returns reset count of resettable energy values of requested meter.

Reset counter is incremented each time these registers are reset. It holds number of times the resettable registers are reset.

Protected

Yes (Authentication required)

Request

```
GET /meters/<serial>/energy/resetcounter HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/energy/resetcounter

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "resetcounter": {
        "activeexport": "0.00",
        "activeimport": "0.00"
      }
}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

Description

reset counter: combination of active import and export energy reset counts

- activeimport[array] - value of active import
- activeexport[array] - value of active export

6.3.59 GET /meters/<serial>/powers

A GET call to /meters/<serial>/power returns realtime power values of requested meter.

Protected

Yes (Authentication required)

Request

```
GET /meters/<serial>/power HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/power

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
  "active" : {
    "l1" : [ "2000001", "W" ],
    "l2" : [ "2000002", "W" ],
    "l3" : [ "2000003", "W" ],
    "total" : [ "2000000", "W" ]
  },

  "reactive" : {
    "l1" : [ "100000", "W" ],
    "l2" : [ "1000002", "W" ],
    "l3" : [ "1000003", "W" ],
    "total" : [ "100000", "W" ]
  },
  "apparent" : {
    "l1" : [ "1100001", "W" ],
    "l2" : [ "1100002", "W" ],
    "l3" : [ "1100003", "W" ],
    "total" : [ "1100004", "W" ]
  }
}
```

Description

- active [Object] - Object of active power.
  lx [Array] - Active line power value and unit [x can be 1 to 3].
  total [Array] - Total active power and unit.
- reactive [Object] - Object of reactive energy.
  lx [Array] - Reactive line power and unit[x can be 1 to 3].
  total [Array] - Total reactive power and unit.
- apparent [Object] - Object of apparent energy.
  lx [Array] - Apparent line power and unit [x can be 1 to 3].
  total [Array] - Total apparent power and unit.
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.60 GET /meters/<serial>/instrument

A GET call to /meters/<serial>/instrument returns realtime voltage and current values.

Protected

Yes (Authentication required)

Request

```
GET /meters/<serial>/instrument HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Example

- https://192.168.1.12/meters/ABB_8/instrument

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "l1":{
        "current":["0.00","A"],
        "currentquadrant":["0",""],
        "phaseanglecurrent":["0.0","deg"],
        "phaseanglepower":["0.00","deg"],
        "phaseanglevoltage":["0.0","deg"],
        "powerfactor":["0.000",""],
        "voltage":["227.1","V"]
    },
    "l2":{
        "current":["0.00","A"],
        "currentquadrant":["0",""],
        "phaseanglecurrent":["0.0","deg"],
        "phaseanglepower":["0.00","deg"],
        "phaseanglevoltage":["5.5","deg"],
        "powerfactor":["0.000",""],
        "voltage":["56.8","V"]
    },
    "l3":{
        "current":["0.00","A"],
        "currentquadrant":["0",""],
        "phaseanglecurrent":["0.0","deg"],
        "phaseanglepower":["0.00","deg"],
        "phaseanglevoltage":["-164.7","deg"],
        "powerfactor":["0.000",""],
        "voltage":["42.9","V"]
    },
    "l1-l2":{
        "voltage":["182.8","V"],
    },
    "l2-l3":{
        "voltage":["81.80","V"],
    },
    "l1-l3":{
```

```
            "voltage":["251.920","V"],
        },
        "neutral":{
            "current":["0.00","A"]
        },
        "total":{
            "currentquadrant":["0",""],
            "phaseanglepower":["0.00","deg"],
            "powerfactor":["0.000",""]
        },
        "frequency":{
            "frequency":["49.78","Hz"]
        }
    }
```

Description

- lx [Object] - Object of line values[x can be 1 to 3].
  current [Array] - line current and unit.
  currentquadrant [Array] - line currentquadrant and unit is null.
  phaseanglecurrent [Array] - line phaseanglecurrent and unit.
  phaseanglepower [Array] - line phaseanglepower and unit.
  phaseanglevoltage [array] - line phaseanglevoltage and unit.
  powerfactor [Array] - line powerfactor and unit is null.
  voltage [Array] - line voltage and unit.
- l1-l2 [Object] - Object of line values.
  voltage [Array] - voltage between l1 and l2 and unit.
- l2-l3 [Object] - Object of line values.
  voltage [Array] - voltage between l2 and l3 and unit
- l1-l3 [Object] - Object of line values.
  voltage [Array] - voltage between l1 and l3 and unit.
- neutral [Object] - Object of neutral current values.
  current [Array] - neutral current and unit.
- total [Object] - Object of total values.
  currentquadrant [Array] - total currentquadrant and unit is null.
  phaseanglepower [Array] - total phaseanglepower and unit.
  powerfactor [Array] - total powerfactor and unit is null.
- frequency [Object] - Object of frequency values.
  frequency [Array] - frequency value and unit.
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.61 GET /meters/<serial>/harmonics/voltage

A GET call to /meters/<serial>/harmonics/voltage returns voltage harmonic values of requested meter.

Protected Yes (Authentication required)

Request

```
GET /meters/<serial>/harmonics/voltage HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/harmonics/voltage

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "l1": {
        "harmonics": [
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0"
        ],
        "thd": "0"
    },
    "l2": {
        "harmonics": [
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
```

```
                "0",
                "0",
                "0",
                "0",
                "0"
            ],
            "thd": "0"
        },
        "l3": {
            "harmonics": [
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0"
            ],
            "thd": "0"
        },
        "l1-l2": {
            "harmonics": [
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
```

```
                "0",
                "0",
                "0"
            ],
            "thd": "0"
        },
        "l2-l3": {
            "harmonics": [
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0"
            ],
            "thd": "0"
        },
        "l1-l3": {
            "harmonics": [
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
                "0",
```

```
                "0"
            ],
            "thd": "0"
        }
    }
```

Description

There can be a max of 16 harmonics in current version of meters.

Note

Values are represented in %.

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.62 GET /meters/<serial>/harmonics/current

A GET call to /meters/<serial>/harmonics/current returns current harmonic values of requested meter.

Protected Yes (Authentication required)

Request

```
GET /meters/<serial>/harmonics/current HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/harmonics/current

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

  {
      "l1": {
          "harmonics": [
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0"
          ],
          "thd": "0"
      },
      "l2": {
          "harmonics": [
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
              "0",
```

```
            "0",
            "0",
            "0",
            "0",
            "0"
        ],
        "thd": "0"
    },
    "l3": {
        "harmonics": [
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0"
        ],
        "thd": "0"
    },
    "neutral": {
        "harmonics": [
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
            "0",
```

```
                    "0",
                    "0",
                    "0"
                ],
                "thd": "0"
            }
        }
```

Description

There can be a max of 16 harmonics in current version of meters.

Note

Values are represented in %.

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.63 GET /meters/<serial>/io

A GET call to /meters/<serial>/io returns io status.

Protected Yes (Authentication required)

Request

```
GET /meters/<serial>/io HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Example

- https://192.168.1.12/meters/ABB_8/io

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "iostatus": [
        {
            "currentstate": 0
        },
        {
            "currentstate": 0,
            "storedstate": 0,
            "counter": 292939
        },
        {
            "currentstate": 1,
            "storedstate": 1,
            "counter": 48448
        },
        {
            "currentstate": 0
        }
    ]
}
```

Description

User can configure maximum of 4 I/O ports in the meter, either input or output.

- currentstate[Int] - currentstate of input [ON or OFF] - Applicable in case of Input/Output
- storedstate[Int] - storedstate of input [ON or OFF] - Applicable in case of Input only
- counter [Int] - counter value of input - Applicable in case of Input only
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.64 GET /meters/<serial>/io/configuration

A GET call to /meters/<serial>/io/configuration returns configuration of Input and Outputs.

Protected Yes (Authentication required)

Request

```
GET /meters/<serial>/io/configuration HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/io/configuration

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0s

{
  "ioconfig": [
     "pulse_out",
     "always_on",
     "alarm_out",
     "input"
  ]
}
```

Description

Possible channels for an IO port are:

- input
- communiation_out
- alarm_out
- pulse_out
- tariff_out
- always_on
- always_off
- dont_change
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.65 GET /meters/<serial>/io/pulse

A GET call to /meters/<serial>/io/pulse returns configuration of pulse Input and Outputs.

Protected

Yes (Authentication required)

Request

```
GET /meters/<serial>/io/pulse HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/io/pulse

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "pulse": [
        {
            "length": 100,
            "frequency": 10000,
            "energytype": "active_import",
            "port": 0
        },
        {
            "length": 100,
            "frequency": 10000,
            "energytype": "active_export",
            "port": 1
        },
        {
            "length": 100,
            "frequency": 10000,
            "energytype": "active_import",
            "port": 255
        },
        {
            "length": 100,
            "frequency": 10000,
            "energytype": "reactive_export",
            "port": 255
        }
    ]
}
```

Description

pulse[Array of Objects]

- length[int] - Pulse length in milliseconds
- frequency[int] - Pulse frequency in impulses per second
- energytype[string] - Pulse energy type
  Energy types are:
  active_import = 0
  active_export = 1
  reactive_import = 2
  reactive_export = 3
  turned_off = 4
  invalid = 0xff
- port[int] - Port number[1...4 or 255(Turned Off)]
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
  Server: embOS/IP
  Accept-Ranges: bytes
  Content-Length: 64
  Content-Type: application/json

  {
   "status":"error",
   "des":"received invalid response from COSEM."
  }
```

6.3.66 POST /meters/<serial>/io

A POST call to /meters/<serial>/io updates Input Output state.

Protected Yes (Authentication required)

Request

```
POST /meters/<serial>/io HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Type:application/json
Content-Length: 437

{
    "iostatus": [
        {
            "port": 1,
            "currentstate": 0
        },
        {
            "port": 2,
            "reset": [
                "storedstate",
                "counter"
            ]
        },
        {
            "port": 3,
            "reset": [
                "storedstate",
                "counter"
            ]
        },
        {
            "port": 4,
            "currentstate": 0
        }
    ]
}
```

Description

User can configure maximum of 4 I/O ports in the meter, either input or output.

- port [int] - Can be 1, 2, 3 or 4
- reset [array] - Array can include 'storedstate', 'counter' or both. Applicable in case of input only.
- currentstate [int] - 1 to set to ON, 0 to set to OFF. Applicable in case of outputs only.

Some possible payload examples:

- port1 as output, and its state is set to OFF state.
  { "iostatus": [{ "port": 1, "currentstate": 0 }, { "port": 2, "reset": ["storedstate", "counter"] }] }
  * Port1 as output, and its state is set to OFF state,
  Port2 as input, and its reset array contains ["storedstae","counter"]

```
{
    "iostatus": [
        {
        "port": 1,
        "currentstate": 0
        },
        {
            "port": 2,
            "reset": [
            "storedstate",
            "counter"
            ]
        }
    ]
}
```

- Port1 as output, and its state is set to OFF state,
  Port4 as output, and it's state is set to ON state
  {
  "iostatus": [
  { "port": 1, "currentstate": 0 }, { "port": 4, "currentstate": 1 }] }
- Port3 as input, and it's reset array contains ["storedstae","counter"],
  Port4 as output, and it's state is set to ON state
  {
  "iostatus": [
  { "port": 3, "reset": ["storedstate", "counter"] }, { "port": 4, "currentstate": 1 }] }

Examples

- https://192.168.1.12/meters/ABB_8/io

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.67 POST /meters/<serial>/io/configuration

A POST call to /meters/<serial>/io/configuration updates Input Output configuration.

Protected

Yes (Authentication required)

Request

```
POST /meters/<serial>/io/configuration HTTP/1.1
  Authorization: Basic XXXXXXXXXXXXXXX=
  Content-Type:application/json
  Content-Length: 22

  {
      "ioconfig":  [ 3 , 5 , 4 , 2 ]
  }
```

Description

ioconfig[Array of Input and outputs] - Array can contain up to 4 elements.

The following code to be provided for specific port:

- Input = 0
- Communication Out = 1
- Alarm Out = 2
- Pulse Out = 3
- Tariff Out = 4
- Always On = 5
- Always Off = 6
- Dont Change = 0xff

Examples

- https://192.168.1.12/meters/ABB_8/io/configuration

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.68 POST /meters/<serial>/io/pulse

A POST call to /meters/<serial>/io/pulse updates pulse configuration.

Protected Yes (Authentication required)

Request

```
POST /meters/<serial>/io/pulse HTTP/1.1
 Authorization: Basic XXXXXXXXXXXXXXX=
 Content-Type:application/json
 Content-Length: 398

 {
   "pulse": [
       {
         "id": 1,
         "port": 2,
         "length": 100,
         "frequency": 10000,
         "energytype": 2
       },
       {
         "id": 2,
         "port": 1,
         "length": 100,
         "frequency": 10000,
         "energytype": 2
        },
        {
         "id": 3,
         "port": 4,
         "length": 100,
         "frequency": 10000,
         "energytype": 0
        },
        {
         "id": 4,
         "port": 255,
         "length": 100,
```

```
            "frequency": 10000,
            "energytype": 3
          }
    ]
}
```

Description

- pulse[Array of Objects]
- length[int] - Pulse length in milliseconds
- frequency[int] - Pulse frequency in impulses per second
- energytype[string] - Pulse energy type
  Energy types are:
  active_import = 0
  active_export = 1
  reactive_import = 2
  reactive_export = 3
  turned_off = 4
  invalid = 0xff
- port[int] - Port number[1...4 or 255(Turned Off)]

Note

There are 4 pulse outputs.

Any pulse output can be assigned to any port.

Thus each of pulse description to include id field specifying pulse output 1,2,3 or 4.

Then specify port to assign.

Pulse output can be turned off by specifying port number as 255.

Example

- https://192.168.1.12/meters/ABB_8/io/pulse

Response

- 200 Success

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.69 GET /meters/<serial>/previousvalues/<from*date*>/<*count*|*to*date>

A GET call to /meters/<serial>/previousvalues/<from*date*>/<*count* | *to*date> returns previous values for the given combinations:

- from date and count
- between from date & to date

Protected

Yes (Authentication required)

Request

```
 GET /meters/<serial>/previousvalues/<fromdate>/
<count|todate> HTTP/1.1
    Authorization: Basic XXXXXXXXXXXXXXX=
```

Description

- *fromdate*
  datetime in **YYYY:MM:DDTHH:MM:SS** form, data is requested from this time
- *todate*
  datetime in **YYYY:MM:DDTHH:MM:SS** form, data is requested till this time
- *count*
  count up to 100

Examples

from date and count

https://192.168.1.12/meters/ABB_8/previousvalues/2012-07-01T01:00:00/100

- from date and to date
  https://192.168.1.12/meters/ABB_8/previousvalues/2012-07-01T01:00:00/2012-07-07T01:00:00
- default case - no date and count given. All values in meter will be returned.
  https://192.168.1.12/meters/ABB_8/previousvalues

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "mapping": [
        [ "active_import_total",  "Wh"  ],
        [ "active_export_total",  "Wh"  ],
    ],
    "interval": "daily",
    "items": [
        [ [ "1", "0.000" ], [ "1", "0.000" ],
                                { "time": "2012-07-
23T00:00:00" } ],
        [ [ "1", "0.000" ], [ "1", "0.000" ],
                                { "time": "2012-07-
22T00:00:00" } ],
        [ [ "1", "0.000" ], [ "1", "0.000" ],
                                { "time": "2012-07-
24T00:00:00" } ]
    ]
}
```

Description

Items contains array of snapshot values.

No of snapshots depends on requested time period.

Each snapshot is a hash of **status**, **timestamp** and **value**.

Valid values for status- it is combination of bit positions.

- BIT0: 0 - Not available, 1 - Snapshot already exists
- BIT1: 0 - OK, 1 - Data Error
- BIT2: 0 - OK, 1 - Power off

The following explanation elaborates the status handling:

1. If bit1 is set it is data error and no need to parse further.
   Else follow next step
2. If bit0 is set follow next step
   Else conclude as data not available
3. If bit2 is set it is a power off scenario
   Else it is good and OK status.

Time tag provides each interval end time during which values are captured.

- 502 Internal Server Error

```
HTTP/1.1 502 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.70 GET /meters/<serial>/previousvalues/configuration

A GET call to /meters/<serial>/previousvalues/configuration returns previousvalues configuration.

Protected

Yes (Authentication required)

Request

```
 GET /meters/<serial>/previousvalues/configuration
HTTP/1.1
    Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/previousvalues/configuration

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
  "interval":"weekly",
  "weekday" : 1,
  "pvconfig":[1,2,3,4,5,6,7,8,9,10,11,12,53,54]
}
```

Description

- *interval*
  Interval can be daily, weekly or monthly
- *weekday*
  Week day is applicable if period is week.
  And values are 1 - Monday to 7 - Sunday
- *pvconfig*
  This is an array of channel ID mapped to /storablequantities.
- 502 Internal Server Error

```
HTTP/1.1 502 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.71 POST /meters/<serial>/previousvalues/configuration

A POST call to /meters/<serial>/previousvalues/configuration configures channels of previous values. Maximum of 50 channels can be configured.

Note

Configuration erases all entries stored so far and clears current configuration.

Protected

Yes (Authentication required)

Request

```
 POST /meters/<serial>/previousvalues/configuration
HTTP/1.1
  Authorization: Basic xxxxxxxx=
  Content-Type:application/json
  Content-Length: 93

    {
      "interval":"weekly",
      "weekday" : 1,
      "pvconfig":[1,2,3,4,5,6,7,8,9,10,11,12,53,54]
    }
```

Description

- *interval*
  Interval can be "daily", "weekly" or "monthly"
- *weekday*
  Weekday is optional and applicable in case if interval is "weekly"
- *pvconfig*
  Array of quantity ids as per /storablequantities

Examples

- https://192.168.1.12/meters/ABB_8/previousvalues/configuration

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "status":"success"
}
```

- 502 Internal Server Error

```
HTTP/1.1 502 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.72 DELETE /meters/<serial>/previousvalues

A DELETE call to /meters/<serial>/previousvalues clears all Previous Values.

Protected

Yes (Authentication required)

Request

```
DELETE /meters/<serial>/previousvalues HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/previousvalues

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "status":"success"
}
```

- 403 Forbidden

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "status": "error",
    "des": "received READ_WRITE_DENIED response
from COSEM."
}
```

- 502 Internal Server Error

```
HTTP/1.1 502 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.73 GET /meters/<serial>/loadprofiles/<channel>/<from*date*>/ <*count*|*to*date>

A GET call to /meters/<serial>/loadprofiles/<channel>/<fromdate>/<count/to-date> returns lp values per channel for the given combinations:

- date & count
- between from date & to date

Request query parameters

- fromdate : datetime in yyyy:mm:ddThh:mm:ss form, data is requested from this time
- count:count can be up to 40000 depending on configuration and meter type.
- todate : datetime in yyyy:mm:ddThh:mm:ss form, data is requested till this time

Protected

Yes (Authentication required)

Request

```
GET /meters/<serial>/loadprofiles/<channel>/<from>/
<count/to> HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

Following are the possible scenarios:

- from date and count
  https://192.168.1.12/meters/ABB_8/loadprofiles/2012-07-01T01:00:00/100
- from date and to date
  https://192.168.1.12/meters/ABB_8/loadprofiles/2012-07-01T01:00:00/2012-07-07T01:00:00
- default case - no date and count given
  https://192.168.1.12/meters/ABB_8/loadprofiles

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
  "mapping":
    [
    "active_import_total",  "Wh"
    ],
    "items": [
     [1, "12:31:2327 17:3:9",0]
   ]
}
```

Description

- Items contains array of snapshot values. No of snapshots depends on requested time period.
- Each snapshot is a hash of **status**, **timestamp** and **value**.
- Valid values for status - it is a combination of bit positions as described below.
  BIT0: 0x01 - Snapshot already exist
  BIT1: 0x02 - Snapshot restart
  BIT2: 0x04 - Snapshot interval is too long
  BIT3: 0x08 - Snapshot interval is too short
  BIT4: 0x10 - Date time changed during snapshot-interval
  BIT5: 0x20 - Bad value, means accumulator value is not correct
  BIT6: 0x40 - Reserved
  BIT7: 0x80 - Reserved
- 502 Internal Server Error

```
HTTP/1.1 502 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.74 GET /meters/<serial>/loadprofiles/configuration

A GET call to /meters/<serial>/loadprofiles/configuration returns lp configuration.

Protected

Yes (Authentication required)

Request

```
GET /meters/<serial>/loadprofiles/configuration HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/loadprofiles/configuration

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "lpconfig": [
        [
            60,
            1,
            15000
        ],
        [
            120,
            69,
            7000
        ],
        [
            21600,
            72,
            2000
        ],
        [
            180,
            75,
            5000
        ],
        [
            3600,
            78,
            1000
        ],
        [
            600,
            79,
            2000
        ],
        [
            450,
            80,
```

```
                3000
            ],
            [
                86400,
                70,
                5000
            ]
        ]
    }
```

Description

lpconfig [Array of lp objects[interval,quantity,Item count]]

- interval[Int]: interval in seconds.
  Possible values are 1m, 2m , 3m, 5m , 10m , 15m , 30m, 1h, 2h , 3h , 6h , 12h, 1d.
  Ex: 5 mins will be represented as 300 secs
- quantity[Int]:
  Quantity is the index of measurement parameter from / storablequantities list.
- Item count[Int]:
  Sum of all item counts in 8 channels should not exceed 40000.
- 502 Internal Server Error

```
HTTP/1.1 502 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.75 POST /meters/<serial>/loadprofiles/configuration

A POST call to /meters/<serial>/loadprofiles/configuration configures lp configuration for all 8 channels.

Note

Configuration erases all entries stored so far and clears current configuration.

Protected

Yes (Authentication required)

Request

```
POST /meters/<serial>/loadprofiles/configuration HTTP/1.1
```

```
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Type:application/json
Content-Length: 205

{
    "lpconfig": [
        [
            60,
            1,
            15000
        ],
        [
            120,
            69,
            7000
        ],
        [
            21600,
            72,
            2000
        ],
        [
            180,
            75,
            5000
        ],
        [
            3600,
            78,
            1000
        ],
        [
            600,
            79,
```

```
            2000
        ],
        [
            450,
            80,
            3000
        ],
        [
            86400,
            70,
            5000
        ]
    ]
}
```

Description

lpconfig [Array of lp objects[interval,quantity,Item count]]

- interval[Int]: interval in seconds.
  Possible values are 1m, 2m , 3m, 5m , 10m , 15m , 30m, 1h, 2h , 3h , 6h , 12h, 1d.
  Ex: 5 mins will be represented as 300 secs
- quantity[Int]:
  Quantity is the index of measurement parameter from /storablequantities list.
- Item count[Int]:
  Sum of all item counts in 8 channels should not exceed 40000.

Examples

- https://192.168.1.12/meters/ABB_8/loadprofiles/configuration

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "status":"success"
}
```

- 502 Internal Server Error

```
HTTP/1.1 502 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.76 DELETE /meters/<serial>/loadprofiles

A DELETE call to /meters/<serial>/loadprofiles clears LoadProfile values.

Protected

Yes (Authentication required)

Request

```
DELETE /meters/<serial>/loadprofiles/<channel> HTTP/
1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Description

Channel is optional. When omitted it erases all channels.

Examples

- https://192.168.1.12/meters/ABB_8/loadprofiles/1
- https://192.168.1.12/meters/ABB_8/loadprofiles

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "status":"success"
}
```

- 403 Forbidden

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status": "error","des": "received READ_WRITE_DENIED
response from COSEM."}
```

- 502 Internal Server Error

```
HTTP/1.1 502 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.77 GET /meters/<serial>/demand/<from*date*><*count*|*to*date>

A GET call to /meters/<serial>/demand/<channel>/<from*date*>/<*count* | *to*date> returns demand values for the given combinations:

- from date & count
- between from date & to date

Protected

Yes (Authentication required)

Request

```
 GET /meters/<serial>/demand/<channel>/<fromdate>/
<count|todate> HTTP/1.1

    Authorization: Basic XXXXXXXXXXXXXXX=
```

Request parameters

- fromdate : datetime in YYYY:MM:DDTHH:MM:SS form, data is requested from this time
- count: count can be up to 500
- todate: datetime in **YYYY:MM:DDTHH:MM:SS** form, data is requested till this time

Examples

- from date and count
  https://192.168.1.12/meters/ABB_8/demand/2012-07-01T01:00:00/100
- from date and to date
  https://192.168.1.12/meters/ABB_8/demand/2012-07-01T01:00:00/2012-07-07T01:00:00
- default case - no date and count given
  https://192.168.1.12/meters/ABB_8/demand

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "mapping": [
        [
            "active_import_total",
            "W"
        ],
        [
            "active_import_total",
            "W"
        ],
        [
            "active_import_total",
            "W"
        ]
    ],
    "items": [
        [
            [
                "0",
                "4294967295",
                "2012-06-17T02:40:00",
                "1"
            ],
            [
                "0",
                "4294967295",
                "2012-06-17T02:40:00",
                "2"
            ],
            [
                "0",
                "4294967295",
                "2012-06-17T02:40:00",
                "3"
            ],
```

```
                    {
                        "time": "2012-06-18T00:00:00"
                    }
                ],
                [
                    [
                        "0",
                        "4294967295",
                        "2012-06-16T02:40:00",
                        "1"
                    ],
                    [
                        "0",
                        "4294967295",
                        "2012-06-16T02:40:00",
                        "2"
                    ],
                    [
                        "0",
                        "4294967295",
                        "2012-06-16T02:40:00",
                        "3"
                    ],
                    {
                        "time": "2012-06-17T00:00:00"
                    }
                ]
            ]
        }
```


Description

- mapping
  Configuration map
- Items contains array of snapshot values.
  Array of arrays. Inner array is as per format described below

[status, value, time stamp, level]

- status
  Valid values for status- it is combination of bit positions:
  BIT0: 0 - Not available, 1 - Snapshot already exists
  BIT1: 0 - OK, 1 - Data Error
  BIT2: 0 - OK, 1 - Power off

Following explanation elaborates the status handling:

1. If bit1 is set it is data error and no need to parse further
   Else follow next step
2. If bit0 is set follow next step
   Else conclude as data not available
3. If bit2 is set it is a power off scenario
   Else it is good and OK status

- value
  Values in units as per mapping section
- timestamp time stamp when the event is recorded.
  There could be duplication of entries if meter time is set backwards
- level For indicative purposes.
  But whether it is max/min demand to be derived from demand code in configuration.

Time tag provides each interval end time during which the demand values are captured.

- 502 Internal Server Error

```
HTTP/1.1 502 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.78 GET /meters/<serial>/demand/configuration

A GET call to /meters/<serial>/demand/configuration returns demand configuration.

Protected Yes (Authentication required)

Request

```
GET /meters/<serial>/demand/configuration HTTP/1.1
   Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/demand/configuration

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
  "dmc": [
  [ 1, 1],
  [ 1, 2]
  .
  .
  up to 50 channels
 ],
 "interval": [ 900, 60 ]
 "period": "weekly",
 "weekday" : 2
}
```

Description

- dmc
  This is array of configuration of each channel. Array sequence is formatted as [QuantityId, Dem and Code].
  QuantityID is ID from storable quantities
  And Demand code is encoded with billing period, demand type and level as below:

Billing	Demand	Level	Code
1	Min	1	1
1	Min	2	2
1	Min	3	3
1	Max	1	5
1	Max	2	6
1	Max	3	7
2	Min	1	9
2	Min	2	10
2	Min	3	11
2	Max	1	13
2	Max	2	14
2	Max	3	15

- period
  period can be daily, weekly or monthly
- week day
  Week day is applicable if period is week.
  And values are 1 - Monday to 7 - Sunday
- interval
  Interval to be specified as array of [interval, subinterval] in seconds.
  Valid values are:
  60 (1 min)
  120 (2 min)
  180 (3 min)
  300 (5 min)
  600 (10 min)
  900 (15 min)
  1800 (30 min)
  3600 (60 min
- 502 Internal Server Error

```
HTTP/1.1 502 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
```

```
        "status":"error",
        "des":"received invalid response from COSEM."
}
```

6.3.79 POST /meters/ <serial>/demand/ configuration

A POST call to /meters/<serial>/demand/configuration configures demand configuration for all 50 channels.

Note

Configuration erases all entries stored so far and clears current configuration.

Protected Yes (Authentication required)

Request

```
POST /meters/<serial>/demand/configuration HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
Content-Type:application/json
Content-Length: 119

{
  "dmc": [
  [ 1, 1],
  [ 1, 2]
  .
  .
  up to 50 channels
 ],
 "interval": [ 900, 60 ]
 "period": "weekly",
 "weekday" : 2
}
```

Description

- dmc
  This is array of configuration of each channel. Array sequence is formatted as [QuantityId, Dem and Code].
  QuantityID is ID from storable quantities
  Demand code is encoded with billing period, demand type and level as below

Billing	Demand	Level	Code
1	Min	1	1
1	Min	2	2
1	Min	3	3
1	Max	1	5
1	Max	2	6
1	Max	3	7
2	Min	1	9
2	Min	2	10
2	Min	3	11
2	Max	1	13
2	Max	2	14
2	Max	3	15

- period
  period can be daily, weekly or monthly
- week day
  Week day is applicable if period is week.
  And values are 1 - Monday to 7 - Sunday
- interval
  Interval to specified as array of [interval, subinterval] in seconds.
  Sub interval should always be less than or equal to interval value.
  Valid values are 60 (1 min), 120 (2 min), 180 (3 min), 300 (5 min), 600 (10 min), 900 (15 min), 1800 (30 min) and 3600 (60 min).

Examples

- https://192.168.1.12/meters/ABB_8/demand/configuration

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "status":"success"
}
```

- 502 Internal Server Error

```
HTTP/1.1 502 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.80 DELETE /meters/<serial>/demand

A DELETE call to /meters/<serial>/demand clears all Demand values.

Protected Yes (Authentication required)

Request

```
DELETE /meters/<serial>/demand HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/demand

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "status":"success"
}
```

- 403 Forbidden

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
        "status": "error",
      "des": "received READ_WRITE_DENIED response
from COSEM."
    }
```

- 502 Internal Server Error

```
HTTP/1.1 502 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.81 GET /meters/ <serial> /tariff

A GET call to /meters/<serial>/tariff returns daytype, tariff, tariffsource and tariff input configuration.

Protected

Yes (Authentication required)

Request

```
GET /meters/<serial>/tariff HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/tariff

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "daytype":2,
    "season":1,
    "tariff":1,
    "source":1
}
```

Description

- daytype [Int] - Current day's DayId [Refer dayid from [GET]/meters/<serial>/dayprofile].
- season [Int] - Current day's Season [Refer season from [GET]/meters/<serial>/seasonprofile].
- tariff [Int] - value between 1 to 4.
- tariffsource [Int] - [0-Clock, 1-Communication, 2-Input].

Note

Daytype and Season are available only if source is set to Clock.

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "tariff":1,
    "source":2,
    "inputconfig":[1,2,3,4]
}
```

Description

- tariff [Int] - value between 1 to 4.
- source [Int] - [0-Clock, 1-Communication, 2-Input].
- inputconfig [Int] - length of array is equal to number of tariffs.

  Example

  When number of tariffs are 4 array would be [1,2,3,4]. Description of each element of array is described below

```
 Input 4      Input 3
    1     OFF          OFF
    2     OFF          ON
    3     ON           OFF
    4    ON          ON
```

Note

input is available, only if source as Input.

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.82 POST /meters <serial> /tariff

A POST call to /meters/<serial>/tariff updates tariff, source and tariff input configuration.

Protected

Yes (Authentication required)

Request

```
POST /meters/ABB_3/tariff HTTP/1.1
   Host: 192.168.1.12
   Content-Length: 12
   Authorization: Basic XXXXXXXXXXXXXXX=

   {
       "tariff": 2
   }
```

Examples

- https://192.168.1.12/meters/ABB_8/tariff

Description

tariff [Int]: 1 to 4. Tariffsource should be in Communication mode before setting the tariff.

Request

```
POST /meters/ABB_3/tariff HTTP/1.1
   Host: 192.168.1.12
   Content-Length: 30
   Authorization: Basic XXXXXXXXXXXXXXX=

   {
       "source": 2,
       "inputconfig": [1,2,3,4]
   }
```

Examples

- https://192.168.1.12/meters/ABB_8/tariff

Description

source [Int] - [0-Clock, 1-Communication, 2-Input].

inputconfig [Int] - length of array is equal to number of tariffs.

Example

When number of tariffs are 4 array would be [1,2,3,4]. Description of each element of array is described below:

```
Input 4      Input 3
        1    OFF          OFF
        2    OFF          ON
        3    ON           OFF
        4    ON           ON
```

Note

If source is configured as Input, then only user can configure the tariff input.

User can set source alone or with tariff or with input.

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.83 GET /meters/<serial>/tariff/dayprofiles

A GET call to /meters/<serial>/tariff/dayprofiles returns dayprofiles configuration

Protected

Yes (Authentication required)

Request

```
GET /meters/<serial>/tariff/dayprofiles HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/tariff/dayprofiles

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "dayprofiles": [
        {
            "actions": [
                [
                    1,
                    "00:00"
                ],
                [
                    2,
                    "12:00"
                ]
            ],
            "profile": 0
        },
        {
            "actions": [
                [
                    2,
                    "12:00"
                ],
                [
                    3,
                    "15:00"
                ]
            ],
            "profile": 1
        },
        {
            "actions": [
                [
                    9,
                    "02:00"
                ]
            ],
```

```
        "profile": 2
    },
    {
        "actions": [
            [
                11,
                "03:00"
            ]
        ],
        "profile": 3
    },
    {
        "actions": [
            [
                1,
                "04:00"
            ]
        ],
        "profile": 4
    },
    {
        "actions": [
            [
                2,
                "05:00"
            ]
        ],
        "profile": 5
    },
    {
        "actions": [
            [
                3,
                "06:00"
            ]
        ],
        "profile": 6
    },
    {
        "actions": [
            [
                1,
                "07:00"
            ]
```

```
                ],
                "profile": 7
            },
            {
                "actions": [
                    [
                        2,
                        "08:00"
                    ]
                ],
                "profile": 8
            },
            {
                "actions": [
                    [
                        1,
                        "09:00"
                    ]
                ],
                "profile": 9
            }
        ]
    }
```

Description

- profile [Int] - Dayprofileid[1-16, maximum of 16 day profiles can be configured].
- actions [actionId, starttime]:
  startime [String] - Action start time
  [If the profile has only one action then the action will perform from starttime to till end of the day.
  If profile has more than one action, then the second action starttime is, first action's end time.]
  actionId [Int] - Action to be performed when a switchpoint is activated.
- Action Id refers the following actions:
  1 - "action*activate*tariff_1"
  2 - "action*activate*tariff_2"
  3 - "action*activate*tariff_3"
  4 - "action*activate*tariff_4"
  5 - "action*set*output_1"
  6 - "action*reset*output_1"
  7 - "action*set*output_2"
  8 - "action*reset*output_2"
  9 - "action*set*output_3"
  10 - "action*reset*output_3"
  11 - "action*set*output_4"
  12 - "action*reset*output_4"
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.84 POST /meters/<serial>/tariff/dayprofiles

A POST call to /meters/<serial>/tariff/dayprofiles updates dayprofileid, profile-starttime, action.

Protected Yes (Authentication required)

Request

```
POST /meters/ABB_3/tariff/dayprofiles HTTP/1.1
   Host: 192.168.1.12
   Content-Length: 51
   Authorization: Basic XXXXXXXXXXXXXXX=

   {
     "actions":[[1,"12:00"],[2,"15:00"]],"profile":1
   }
```

Parameters

profile [Int] - Dayprofileid[1-16, maximum of16 day profiles can be configured].

- actions [actionId, starttime]:
  actionId [Int] - Which action has to perform when a switchpoint is activated.
  Action Id refers to the following actions:
  1 - "action*activate*tariff_1"
  2 - "action*activate*tariff_2"
  3 - "action*activate*tariff_3"
  4 - "action*activate*tariff_4"
  5 - "action*set*output_1"
  6 - "action*reset*output_1"
  7 - "action*set*output_2"
  8 - "action*reset*output_2"
  9 - "action*set*output_3"
  10 - "action*reset*output_3"
  11 - "action*set*output_4"
  12 - "action*reset*output_4"
- startime [String] - Action stattime
  Note
  If the profile has only one action then the action will perform from starttime to till end of the day.
  If profile has more than one action, then the second action starttime is, first action's end time.

Examples

- https://192.168.1.12/meters/ABB_8/tariff/dayprofiles

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.85 GET /meters/<serial>/tariff/weekprofiles

A GET call to /meters/<serial>/tariff/weekprofiles returns weekprofiles configuration.

Protected

Yes (Authentication required)

Request

```
GET /meters/<serial>/tariff/weekprofiles HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/tariff/weekprofiles

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "weekprofiles": [
        {
            "dayid": [
                1,
                4,
                2,
                5,
                3,
                6,
                1
            ],
            "profile": "Week1"
        },
        {
            "dayid": [
                1,
                2,
                3,
                7,
                5,
                6,
                7
            ],
            "profile": "Week2"
        },
        {
            "dayid": [
                7,
                6,
                5,
                4,
                3,
                2,
                1
```

```
                ],
                "profile": "Week3"
            },
            {
                "dayid": [
                    3,
                    4,
                    7,
                    8,
                    11,
                    12,
                    14
                ],
                "profile": "Week4"
            }
        ]
}
```

Description

- profile [String(30)] - WeekProfileName[maximum of 4 week profiles can be configured].
- dayid [Array of Int value] - Id of dayprofile to execute. [Refer DayProfileId from meters*get*dayprofile].
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.86 POST /meters <serial> /tariff /weekprofiles

A POST call to /meters/<serial>/tariff/weekprofiles updates weekprofiles configuration.

Protected Yes (Authentication required)

Request

```
POST /meters/ABB_3/tariff/weekprofiles HTTP/1.1
   Host: 192.168.1.12
   Content-Length: 207
   Authorization: Basic XXXXXXXXXXXXXXX=

   {
        "weekprofiles":[
        {"profile":"Week1","dayid":[1,4,2,5,3,6,1]},
        {"profile":"Week2","dayid":[1,2,3,7,5,6,7]},
        {"profile":"Week3","dayid":[7,6,5,4,3,2,1]},
       {"profile":"Week4","dayid":[3,4,7,8,11,12,14]}]
   }
```

Parameters

- profile [string(30)] - WeekProfileName[maximum of 4 week profiles can be configured].
- dayid [Array of Int value] - Id of dayprofile to execute. [Refer DayProfileId from meters*get*dayprofile].

Examples

- https://192.168.1.12/meters/ABB_8/tariff/weekprofiles

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.87 GET /meters/<serial>/tariff/seasonprofiles

A GET call to /meters/<serial>/tariff/seasons returns season profiles configuration.

Protected

Yes (Authentication required)

Request

```
GET /meters/<serial>/tariff/seasons HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/tariff/seasons

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "seasonprofiles": [
        {
            "day": 254,
            "month": 10,
            "profile": "Winter",
            "weekprofile": "Week1"
        },
        {
            "day": 1,
            "month": 5,
            "profile": "Summer",
            "weekprofile": "Week2"
        },
        {
            "day": 1,
            "month": 9,
            "profile": "Autumn",
            "weekprofile": "Week3"
        }
    ]
}
```

Description

- day [Int] - Season startdayOfMonth.
- month [Int] - Season startmonth.
- profile [String(30)] - Season name.
- weekprofile [String(30)] - weekprofile name to execute. [Refer weekname from meters*get*weekprofile].
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.88 POST /meters/<serial>/tariff/seasonprofiles

A POST call to /meters/<serial>/tariff/seasonprofiles updates seasonprofiles configuration

Protected Yes (Authentication required)

Request

```
POST /meters/ABB_3/tariff/seasonprofiles HTTP/1.1
   Host: 192.168.1.12
   Content-Length: 296
   Authorization: Basic XXXXXXXXXXXXXXX=

   {
       "seasonprofiles":[
      {"day":254,"month":10,"profile":"Winter","week-
profile":"Week1"},
      {"day":1,"month":5,"profile":"Summer","weekpro-
file":"Week2"},
      {"day":1,"month":9,"profile":"Autumn","weekpro-
file":"Week3"}]
   }
```

Description

- day [Int] - Season startdayOfMonth.
- month [Int] - Season startmonth.
- profile [String(30)] - Season name.
- startime [String] - Season starttime [Month:Day].
- weekprofile [String] - weekprofile name to execute. [Refer weekname from meters*get*weekprofile].

Examples

- https://192.168.1.12/meters/ABB_8/tariff/seasonprofiles

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.89 GET /meters/<serial>/tariff/specialdayprofiles

A GET call to /meters/<serial>/tariff/specialdayprofiles returns specialdayprofiles configuration.

Protected

Yes (Authentication required)

Request

```
GET /meters/<serial>/tariff/specialdayprofiles HTTP/1.1
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/meters/ABB_8/tariff/specialdayprofiles

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{
    "specialdayprofiles": [
        {
            "date": {
                "day": 254,
                "month": 11,
                "week": 255,
                "year": 65535
            },
            "dayprofile": 1,
            "profile": 1
        },
        {
            "date": {
                "day": 1,
                "month": 255,
                "week": 255,
                "year": 2011
            },
            "dayprofile": 2,
            "profile": 2
        },
        {
            "date": {
                "day": 1,
                "month": 9,
                "week": 255,
                "year": 65535
            },
            "dayprofile": 14,
            "profile": 3
        }
    ]
}
```

Description

- profile [Int] - SpecialDayProfileId[maximum of 50 special day profiles can be configured].
- Date [Object] - SpecialDay Date.
- dayprofileid - day profile id to execute. [Refer DayProfileId from meters*get*dayprofile].
- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.90 POST /meters/ <serial>/tariff/specialdayprofiles

A POST call to /meters/<serial>/tariff/specialdayprofiles updates specialdayprofiles configuration for matched date. If there is no match then it adds a new entry. If profileid already exists but it doesn't match with specified date, then it adds a new entry and profileid will be updated sequentially.

Maximum of 50 special day profiles can be configured.

Protected Yes (Authentication required)

Request

```
POST /meters/ABB_3/tariff/specialdayprofiles HTTP/1.1
   Host: 192.168.1.12
   Content-Length: 88
   Authorization: Basic XXXXXXXXXXXXXXX=

   {
       "profile":1,"dayprofile":2,

"date":{"year":1989,"month":11,"week":4,"day":254}
   }
```

Description

- profile [Int] - SpecialDayProfileId.
- Date [Object] - SpecialDay Date.
- dayprofileid - day profile id to execute. [Refer DayProfileId from meters*get*dayprofile].

Examples

- https://192.168.1.12/meters/ABB_8/tariff/specialdayprofiles

Response

- 200 OK

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{"status":"success"}
```

- 500 Internal Server Error

```
HTTP/1.1 500 Internal Server Error
Server: embOS/IP
Accept-Ranges: bytes
Content-Length: 64
Content-Type: application/json

{
    "status":"error",
    "des":"received invalid response from COSEM."
}
```

6.3.91 GET /lasterror/<id>

A GET call to /lasterror/<id> gets details about recent error of specified id.

There could be a scenario where error occurred after initiating transmission of successful response.

In that case client receives a 200 header, but gets a parse error, as body is not fully formatted.

To be able to diagnose the error condition clients can call this resource to get details of error.

Protected

Yes (Authentication required)

Request

```
GET /lasterror/&lt;id>
Authorization: Basic XXXXXXXXXXXXXXX=
```

Examples

- https://192.168.1.12/lasterror/23

Response

- 200 Successful

```
HTTP/1.1 200 OK
Content-Type: application/json
Server: embOS/IP
Transfer-Encoding: chunked
Link:/lasterror/0

{ "des" : "detailed description" }
```

6.3.92 Annexure1

This Annexure describes the details of each parameter of meter label.

Meter Label Format

Meter label is in the form of

```
XYZ ABC-DEF
```

where

- [X] stands for type of Enclosure:
  A : Enclosure -7 DIN Advanced
  B : Enclosure -4 DIN,2 DIN Advanced,Basic High,Basic Low
  C : Enclosure -3 DIN,1 DIN competitive
  D : Enclosure -2 DIN DC
  E : Enclosure -No Enclosure Embedded
  M : Enclosure -96 x 96 Multimeter
  N : Enclosure -96 x 96 Analyser
- [Y] stands for type of Electronics:
  1 : Electronics - competitive
  2 : Electronics - Basic Low
  3 : Electronics - Basic High
  4 : Electronics - Advanced
  5 : Electronics - Multimeter 96 x 96
  6 : Electronics - Analyzer 96 x 96
- [Z] stands for type of phase:
  1 : Single phase direct connected
  2 : Single phase indirect connected
  3 : Three phase direct connected
  4 : Three phase indirect connected
- [A] stands for type of functionality level of meter:
  0 : Functionality level - Iron
  1 : Functionality level - Steel
  2 : Functionality level - Bronze
  3 : Functionality level - Silver
  4 : Functionality level - Gold
  5 : Functionality level - Platinum
- [B] stands for accuracy class:
  1 : Accuracy Class 1.0
  2 : Accuracy Class 2.0
  5 : Accuracy Class 0.5
- [C] stands for type of interface:
  0 : Interface - No Interface
  1 : Interface - IR port only
  2 : Interface - RS-485 port
  3 : Interface - M-Bus

4 : Interface - (Zigbee + RS-485 port)
5 : Interface - (Zigbee + M-Bus)
6 : Interface - Zigbee only
7 : Interface - Modbus TCP
8 : Interface - LonWorks
9 : Interface - Profibus

- [D] stands for approved authority:
  1 : IEC approved + MID approved and verified
  2 : IEC approved + GOST approved abd verified
  3 : IEC approved
- [E] stands for version class:
  0 : ABB standard version
  1 : Industrial version
  2 : Rail application version
- [F] stands for class variation of ABB/BHM versions:
  0 : ABB standard version
  1 : BHM version 1
  2 : BHM version 2
  3 : BHM version 3
  4 : BHM version 4
  5 : BHM version 5
  6 : BHM version 6
  7 : BHM version 7
  8 : BHM version 8
  9 : BHM version 9
  A : BHM version A

Examples

A44 512-100 : 3 phase(indirect) Advanced platinum, RS-485 interface

A41 454-100 : 1 phase(direct) Advanced gold,(RS-485 + zigbee) interface