Version: 1.0

PM1

The PM1 component handles electrical power metering capabilities. It uses `PM1` as the RPC namespace and provides the methods:

- `PM1.SetConfig` to update the component's configuration
- `PM1.GetConfig` to obtain the component's configuration
- `PM1.GetStatus` to obtain the component's status
- `PM1.ResetCounters` to reset component's energy counters

PM1 components are identified with `PM1:<id>` in objects containing multiple component payloads.

Methods

PM1.SetConfig

Properties:

Property	Type	Description
`id`	*number*	Id of the PM1 component instance
`config`	*object*	Configuration that the method takes

Find more about the config properties in config section

PM1.GetConfig

Properties:

Property	Type	Description
`id`	*number*	Id of the PM1 component instance

Find the PM1.GetConfig response properties in config section

PM1.GetStatus

Properties:

Property	Type	Description
`id`	*number*	Id of the PM1 component instance

Find more about the status response properties in status section

PM1.ResetCounters

This method resets associated counters.

Request

Parameters:

Property	Type	Description
`id`	*number*	Id of the PM1 component instance. **Required**
`type`	*array of strings*	Array of strings, selects which counter to reset **Optional**

> **NOTE**
>
> If no 'type' is provided, the method will reset all available counters.

Response

Attributes in the result:

Property	Type	Description
`aenergy`	*object*	Information about the active energy counter prior to reset **Property** / **Type** / **Description** `total` / *number* / Last counter value of the total energy consumed in Watt-hours
`ret_aenergy`	*object*	Information about the returned active energy counter prior to reset **Property** / **Type** / **Description** `total` / *number* / Last counter value of the total returned energy consumed in Watt-hours

Configuration

The configuration of the PM1 component contains information about the id and name of the component.

Property	Type	Description
`id`	*number*	Id of the PM1 component instance
`name`	*string or null*	Name of the PM1 instance

Property	Type	Description
`reverse`	*bool*	Reverse measurement direction of active power and energy for the PM1 component. *setting the reverse option requires restart*

Status

The status of the PM1 component contains information about the measured variables. To obtain the status of the PM1 component its `id` must be specified.

Property	Type	Description
`id`	*number*	Id of the PM1 component instance
`voltage`	*number*	Last measured voltage in Volts
`current`	*number*	Last measured current in Amperes
`apower`	*number*	Last measured instantaneous active power (in Watts) delivered to the attached load
`aprtpower`	*number*	Last measured instantaneous apparent power (in Volt-Amperes) delivered to the attached load (shown if applicable)
`pf`	*number*	Last measured power factor (shown if applicable)
`freq`	*number*	Last measured network frequency (shown if applicable)
`aenergy`	*object*	Information about the active energy counter

<table>
<tr><th>Property</th><th>Type</th><th>Description</th></tr>
<tr><td></td><td></td><td>
<table>
<tr><th>Property</th><th>Type</th><th>Description</th></tr>
<tr><td><code>total</code></td><td>number</td><td>Total energy consumed in Watt-hours</td></tr>
<tr><td><code>by_minute</code></td><td>array of type number</td><td>Energy consumption in Milliwatt-hours for the last three complete minutes. The 0-th element indicates the counts accumulated during the minute preceding <code>minute_ts</code>. Present only if the device clock is synced.</td></tr>
<tr><td><code>minute_ts</code></td><td>number</td><td>Unix timestamp marking the start of the current minute (in UTC).</td></tr>
</table>
</td></tr>
<tr><td><code>ret_aenergy</code></td><td>object</td><td>Information about the returned active energy counter *
<table>
<tr><th>Property</th><th>Type</th><th>Description</th></tr>
<tr><td><code>total</code></td><td>number</td><td>Total returned energy consumed in Watt-hours</td></tr>
<tr><td><code>by_minute</code></td><td>array of type number</td><td>Returned energy consumption by minute (in Milliwatt-hours) for the last three minutes (the lower the index of the element in the array, the closer to the current moment the minute)</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Property</th><th>Type</th><th>Description</th></tr>
<tr><td></td><td></td><td>
<table>
<tr><th>Property</th><th>Type</th><th>Description</th></tr>
<tr><td><code>minute_ts</code></td><td>number</td><td>Unix timestamp marking the start of the current minute (in UTC).</td></tr>
</table>
</td></tr>
<tr><td><code>errors</code></td><td>array of type string</td><td>Error conditions occurred. May contain <code>power_meter_failure</code>, <code>out_of_range:voltage</code>, <code>out_of_range:current</code>, <code>out_of_range:aprtpower</code>, <code>out_of_range:apower</code>(shown if at least one error is present)</td></tr>
</table>

NOTE

- `ret_aenergy` - the active energy added to this container is also added to `aenergy` container. All the consumed energy is collected in `aenergy` regardless of the direction(consumed or returned) of the active energy.

Webhook Events

PM1 component supports conditional webhooks.

Events related to the PM1 component that can trigger webhooks:

- `voltage_change` - when the voltage has changed with at least 1V and 10% from the last reported value.
  - `voltage_change` event supports attributes, that can be used to compose conditional webhooks:

Property	Type	Description
`voltage`	*number*	New voltage in Volts

- `current_change` - when the current has changed with at least 0.02A and 5% from the last reported value.
  - `current_change` event supports attributes, that can be used to compose conditional webhooks:

Property	Type	Description
`current`	*number*	New current in Amps

- `apower_change` - when the active power has changed with at least 1W and 5% from the last reported value.
  - `apower_change` event supports attributes, that can be used to compose conditional webhooks:

Property	Type	Description
`apower`	*number*	New active power in Watts

Examples

PM1.SetConfig example

PM1.SetConfig HTTP GET Request | PM1.SetConfig Curl Request | PM1.SetConfig Mos Request

```
http://192.168.33.1/rpc/PM1.SetConfig?id=0&config=
{"name":"Meter","reverse":true}
```

Response

```
{
  "restart_required": true
}
```

PM1.GetConfig example

PM1.GetConfig HTTP GET Request | PM1.GetConfig Curl Request | PM1.GetConfig Mos Request

```
http://192.168.33.1/rpc/PM1.GetConfig?id=0
```

Response

```
{
  "id": 0,
  "name": "PM1device",
  "reverse": false
}
```

PM1.GetStatus example

PM1.GetStatus HTTP GET Request | PM1.GetStatus Curl Request | PM1.GetStatus Mos Request

```
http://192.168.33.1/rpc/PM1.GetStatus?id=0
```

Response

```
{
  "id": 0,
  "voltage": 225.9,
  "current": 0,
  "apower": 0,
  "freq": 50,
  "aenergy": {
    "total": 11.679,
    "by_minute": [
      0,
      0,
      0
    ],
    "minute_ts": 1654511972
  },
  "ret_aenergy": {
    "total": 4.126,
```

```
    "by_minute": [
      0,
      0,
      0
    ],
    "minute_ts": 1654511318
  },
  "errors": [
    "power_meter_failure",
    "out_of_range:voltage",
    "out_of_range:current",
    "out_of_range:apower",
    "out_of_range:aprtpower"
  ]
}
```


PM1.ResetCounters example

PM1.ResetCounters HTTP GET Request | PM1.ResetCounters Curl Request | PM1.ResetCounters Mos Request

```
http://192.168.33.1/rpc/PM1.ResetCounters?id=0&type=["aenergy","ret_aenergy"]
```

Response


```
{
  "aenergy": {
    "total": 11.679
  },
  "ret_aenergy": {
    "total": 5.817
  }
}
```