COMPERE
MQTT communication protocol

V1.9

HENAN COMPERE SMART TECHNOLOGY CO., LTD

Content

1. Purpose & Scope

The MQTT communication protocol is used to agree on the upload format and analysis method of power data. It mainly involves the timing of data upload and the Json message format transmitted by the MQTT protocol.

2. Overview

This protocol is for power meter from COMPERE connecting server or cloud platform using MQTT communication protocol.

The connection is configured and connected according to the manual, where the supported meters are KPM37, KPM312, KPM31, KPM33B.

The connection to the server and the format of data transmission are the same for each meter, but the data tags are different.

The KPM312 has a special data transmission. KPM312 supports uploading data of 4 loops, and the device ID is also divided into 4 different IDs, such as

If the device ID is set to 3120208700001, the default is the device ID of the first loop, and the device IDs of 2~3 loops will be increased by 1, which should be 3120208700001. The device IDs of 2~3 loops will be added 1, which should be 3120208700002, 3120208700003, 3120208700004.

In the entire communication framework, the power meter acts as a client, and the system or platform acts as a server. The server and the client use the MQTT communication protocol as data exchange. Data interaction uses WIFI or 4G as the physical medium of transmission. The payload transmitted by the MQTT communication protocol is a data message in Json format, which contains various real-time electrical parameter data and power consumption data collected by the power meter.

3. The data to be uploaded and the time of uploading

The data to be uploaded mainly includes real-time (second/minute level) electrical parameter data, daily and monthly data.

3.1 Real-time data second level

3.1.1 Real-time data second level electrical parameters include:

- **KPM37:** 3 Phase line voltage and phase voltage, 3 Phase current, 3 phase and total active power, 3 phase and total reactive power, 3 phase and total apparent power, 3 phase and total power factor, frequency, positive sequence, negative sequence, zero sequence voltage and current; three-phase voltage phase angle, three-phase current phase angle, voltage and current unbalance rate, total active power demand, total reactive power demand, total apparent power demand, residual current (optional), four-way temperature (optional). etc.
- **KPM312:** 3 Phase line voltage and phase voltage, 3 Phase current, 3 phase and total active power, 3 phase and total reactive power, 3 phase and total apparent power, 3 phase and total power factor, frequency, voltage and current unbalance rate, total active power demand, total reactive power demand, total apparent power demand, residual current (optional), four-way temperature (optional). etc.
- **KPM31A/B/C:** voltage, current, active power, reactive power, apparent power, power factor, frequency.
- **KPM33A/B:** 3 Phase line voltage and phase voltage, 3 Phase current, 3 phase and total active power, 3 phase and total reactive power, 3 phase and total apparent power, 3 phase and total power factor, frequency.

3.1.2 Data uploaded frequency can be set to:

30,60,300,600,900,1200,1800,3600 seconds.

3.2 Real-time data minute level data

Real-time data minute level includes: Import, export and total active energy, Import, export and total reactive energy, import and export active energy for each rate, monthly maximum active power demand and apparent power demand and occurrence time, harmonic data (total harmonic content of three-phase voltage and current, 3rd, 5th, 7th harmonic content and current harmonic content), etc.

Data uploaded frequency can be set to: 1, 5, 10, 15, 20, 30, 60, 1440minutes.

3.3 Daily data

Daily data includes: Daily import/export active/reactive energy consumption.

Upload time: 0:00 uploading the previous day's data.

Supported models: KPM37, KPM312

For details on data tags, please refer to the description in the message format.

3.4 Remote signal data

When there is a change in DI data, the energy meter will issue a remote signal data message to the server.

Supported models: KPM31B, KPM37, KPM312

3.5 Remote control data

The energy meter subscribes to the remote control message sent by the server, and configures DO according to the remote control message.

Supported models: KPM31B, KPM31C, KPM33B are all 1-way remote control, KPM37 is 2-way remote control, and KPM312 has 1-way remote control for each circuit.

3.6 Data recall

According to the instructions issued by the platform, reply the data (monthly consumption) that needs to be recruited.

Supported model: KPM37

For details of data tags, please refer to the description in the message format.

3.7 Time synchronization

All of Compere's MQTT meters support single meter time synchronization.

3.8 Meter parameter setting

Set the operating parameters of the meters, such as Modbus address, PT, CT, multi rate parameters, etc.;

Supported models: KPM31A/B/C (only supports setting one parameter at a time), KPM37 and KPM312 (supports setting multiple parameters at the same time)

Note: The register address refers to its manual or the Modbus-RTU communication protocol of the corresponding model.

3.9 Read meters' parameters

Reads the operating parameters of the meters, such as Modbus address, PT, CT, multi rate parameters, etc..

Supported models: KPM31A/B/C (only supports reading one parameter at a time), KPM37 (supports reading multiple parameters at the same time)

Note: The register address refers to its manual or the Modbus-RTU communication protocol of the corresponding model.

3.10 Reset meter MQTT parameters

You can reset the meter's MQTT server information and reconfigure it.

Supported instruments: All MQTT meters.

3.11 Read/set MQTT upload frequency

Read and set the MQTT upload frequency of the meters, read and set the second level and minute level separately.

Supported models: all MQTT meters.

Second level (in seconds): 30, 60, 300, 600, 900, 1200, 1800, 3600

Minute level (in minutes): 1, 5, 10, 15, 20, 30, 60, 1440

4. Message format of uploaded data

The communication between the energy meter and the cloud platform adopts the MQTT protocol, and the payload of the protocol transmission adopts Json coding. And the operation id when setting parameters (oprid)

must be 32 bits, otherwise the meter will not respond. The encoding format is as follows:

4.1 Second level data

Topic Name ： MQTT_RT_DATA

MQTT_RT_DATA data packet is sent in multiple splitting packages, distinguished by the data segment " isend ". When the value of the " isend " field in the received data packet is "1", it means that all data packets have been transmitted.

Json data format shown as below:

(Take KPM37 as a sample, in wifi communication, it's sent in splitting packages. In 4G communication, it's sent in one package.)

The 1st part data after splitting

```
{
    "id": "meter_id",       //Meter ID
    "ua": 220.0     ,       //Ua, float,  unit V
    "ub": 220.0     ,       //Ub, float,  unit V
    "uc": 220.0     ,       //Uc, float,  unit V
    "ia": 5.0       ,       //Ia, float,  unit A
    "ib": 5.0       ,       //Ib, float,  unit A
    "time":"20200108104555"   //Time tag,  Year-month-day 2020-01-08,
Clock 10:45:55
    "isend":"0"
}
```

The 2nd part data after splitting：

```
{
    "id": "meter_id",          // Meter ID
    "ic": 5.0     ,            // Ic, float，unit A
    "uab": 380.0     ,         //Uab, float，unit V
    "ubc": 380.0     ,         // Ubc, float，unit V
    "uca": 380.0     ,         // Uca, float，unit V
    "pa": 5.0     ,            //Pa, float，unit kW
"time":"20200108104555"   //Time tag，Year-month-day 2020-01-08,
Clock 10:45:55
    "isend":"0"
}
```

The 3rd part data after splitting:

```
{
    "id": "meter_id",       //Meter ID
    "pb": 5.0     ,         // Pb, float，unit kW
    "pc": 5.0     ,         // Pc, float，unit kW
    "zyggl":123.5,          // Total Active Power(P), float，unit kW
    "qa": 5.0     ,         // Qa, float，unit kvar
    "qb": 5.0     ,         // Qb, float，unit kvar
```

"time":"20200108104555" // Time tag，Year-month-day 2020-01-08, Clock 10:45:55

"isend":"0"

}

The 4th part data after splitting：

{

"id": "meter_id", //Meter ID

"qc": 5.0 , // Qc, float，unit kvar

"zwggl":123.456, // Total reactive power (Q), float, unit kvar

"sa": 5.0 , //Sa, float, unit kVA

"sb": 5.0 , // Sb, float, unit kVA

"sc": 5.0 , // Sc, float, unit kVA

"time":"20200108104555" // Time tag，Year-month-day 2020-01-08, Clock 10:45:55

"isend":"0"

}

The 5th part data after splitting：

{

"id": "meter_id", // Meter ID

"zszgl":123.456, //Total apparent power (S)，kVA

"pfa":0.5 , //Phase A power factor

"pfb":0.5, // Phase B power factor

"pfc":0.5, // Phase C power factor

"zglys":0.5, // Total power factor

"f": 50 , //System frequency, float, unit Hz

"time":"20200108104555" // Time tag, Year-month-day 2020-01-08, Clock 10:45:55

"isend":"0"

}

The 6th part data after splitting:

{

"id": "meter_id", //Meter ID

"U0":0.0, // Zero sequence voltage, V

"U+":0.0, // Positive sequence voltage, V

"U-":0.0, // Negative sequence voltage, V

"I0":0.0, // Zero sequence current, A

"I+":0.0, // Positive sequence current, A

"I-":0.0, // Negative sequence current, A

"time":"20200108104555" // Time tag, Year-month-day 2020-01-08, Clock 10:45:55

"isend":"0"

}

The 7th part data after splitting:

{

"id": "meter_id", //Meter ID

"UXJA":0.0, //UA phase angle，°

"UXJB":0.0, //UB phase angle，°

"UXJC":0.0, //UC phase angle，°

"IXJA":0.0, //IA phase angle，°

"IXJB":0.0, //IB phase angle，°

"IXJC":0.0, //IC phase angle，°

"time":"20200108104555" // Time tag，Year-month-day 2020-01-08, Clock 10:45:55

"isend":"0"

}

The 8th part data after splitting：

{

"id": "meter_id", //Meter ID

"unb": 0.6, // Three-phase voltage unbalance rate

"inb": 0.6, // Three-phase current unbalance rate

"pdm": 10.5, //Total active power demand kw

"qdm": 5.6, //Total reactive power demand kvar

"sdm": 15.6, //Total apparent power demand kVA

"ig":1.3, // Residual current, A

"time":"20200108104555" // Time tag, Year-month-day 2020-01-08, Clock 10:45:55

"isend":"0"

}

The 9th part data after splitting:

{

"id": "meter_id", //Meter ID

"ta":26.5, //Phase A temperature, unit ℃

"tb":26.5, // Phase B temperature, unit ℃

"tc":26.5, // Phase C temperature, unit ℃

"tn":26.5, // Phase N temperature, unit ℃

"time":"20200108104555" // Time tag, Year-month-day 2020-01-08, Clock 10:45:55

"isend":"1"

}

Note: KPM31A/B/C and KPM33A/B upload data in one package, while KPM312 uploads four channels separately.

4.2 Packet format of minute-level data

Topic Name: MQTT_ENY_NOW

MQTT_ENY_NOW data can be split into multiple packets for transmission, distinguished by the field "isend". When the value of the "isend" field in the received data packet is "1", it means that all data packets have been transmitted.

The Json data format is as follows:

(Take KPM37 as a sample, in wifi communication, it's sent in splitting packages. In 4G communication, it's sent in 2 packages.)

The data is divided into 11 packets, part 1:

```
{
   "id": "meter_id",          //Meter ID
   "zygsz":100.00,            //Import total active energy, float, unit kWh
   "fygsz":100.00,            // Export total active energy, float, unit kWh
   "zwgsz" :100.00,           // Import total reactive energy, float, unit kvarh
   "fwgsz" :100.00,           // Export total reactive energy, float, unit kvarh
   "time"   :"20200108104555"   // Time tag,  Year-month-day 2020-01-08,
   Clock 10:45:55
   "isend" :"0"
}
```

2nd part of total 11 packets is:

```
{
   "id": "meter_id",     // Meter ID
   "zyjsz" :100.00,      // Import active tariff 1 energy value, float, unit kWh
```

"fyjsz" :100.00, // Export active tariff 1 energy value, float, unit kWh

"zyfsz" :100.00, // Import active tariff 2 energy value, float, unit kWh

"fyfsz" :100.00, // Export active tariff 2 energy value, float, unit kWh

"time" :"20200108104555" // Time tag, Year-month-day 2020-01-08, Clock 10:45:55

"isend" :"0"

}

3rd part of total 11 packets is:

{

"id": "meter_id", // Meter ID

"zypsz" :100.00, // Import active tariff 3 energy value, float, unit kWh

"fypsz" :100.00, // Export active tariff 3 energy value, float, unit kWh

"zyvsz" :100.00, // Import active tariff 4 energy value, float, unit kWh

"fyvsz" :100.00, // Export active tariff 4 energy value, float, unit kWh

"time" :"20200108104555" // Time tag, Year-month-day 2020-01-08, Clock 10:45:55

"isend" :"0"

}

The data is divided into 11 packets, the 4th part:

{

"id":"meter_id", // Meter ID

"zydvsz":100.00, // Import active tariff 5 energy value, float, unit kWh

"fydvsz" :100.00, // Export active tariff 5 energy value, float, unit kWh

"zy6sz":100.00, // Import active tariff 6 energy value, float, unit kWh

"fy6sz":100.00, // Export active tariff 6 energy value, float, unit kWh

"time":"20200108104555" // Time tag, Year-month-day 2020-01-08, Clock 10:45:55

"isend":"0"

}

The data is divided into 11 packets, the 5th part:

{

"id":"meter_id", // Meter ID

"dmpmax":200.00, // Monthly max active power demand, unit kW

"dmpmaxoct":1675405205, // UTC timestamp of monthly max active power demand occurs, accurate to the second

"dmsmax":260.00, // Monthly max apparent power demand, unit kVA

"dmpmaxoct":1675405205, // UTC timestamp of monthly max apparent power demand occurs, accurate to the second

"time":"20220108104555" // Time tag, Year-month-day 2020-01-08, Clock 10:45:55

"isend" :"0" // If there is no harmonic function, it is 1; if there is harmonic, it is 0

}

The data is divided into 11 packets, the 6th part:

```
{
    "id": "meter_id",                  //Meter ID
    "uathd" :2.00,                    //THD ua
    "ubthd" :2.00,                    //THD ub
    "ucthd" :2.00,                    //THD uc
    "iathd" :2.00,                    //THD ia
     "ibthd" :2.00,                    //THD ib
     "icthd" :2.00,                    //THD ic
    "time"   :"20200108104555"   // Time tag,  Year-month-day 2020-01-
  08, Clock 10:45:55
    "isend" :"0"
}
```

The data is divided into 11 packets, the 7th part:

```
{
    "id": "meter_id",                  // Meter ID
    "uaxbl3" :2.00,                   // Ua 3rd harmonic content rate
    "ubxbl3" :2.00,                   // Ub 3rd harmonic content rate
    "ucxbl3" :2.00,                   // Uc 3rd harmonic content rate
```

"iaxbl3" :2.00, // Ia 3rd harmonic content rate

"ibxbl3" :2.00, // Ib 3rd harmonic content rate

"icxbl3" :2.00, // Ic 3rd harmonic content rate

"time" :"20200108104555" // Time tag, Year-month-day 2020-01-08, Clock 10:45:55

"isend" :"0"

}

The data is divided into 11 packets, the 8th part:

{

"id": "meter_id", // Meter ID

"uaxbl5" :2.00, // Ua 5th harmonic content rate

"ubxbl5" :2.00, // Ub 5th harmonic content rate

"uaxbl5" :2.00, // Uc 5th harmonic content rate

"iaxbl5" :2.00, //Ia 5th harmonic content rate

"ibxbl5" :2.00, //Ib 5th harmonic content rate

"icxbl5" :2.00, //Ic 5th harmonic content rate

"time" :"20200108104555" // Time tag, Year-month-day 2020-01-08, Clock 10:45:55

"isend" :"0"

}

The data is divided into 11 packets, the 9th part:

{

"id": "meter_id", // Meter ID

"uaxbl7" :2.00, // Ua 7th harmonic content rate

"ubxbl7" :2.00, // Ub 7th harmonic content rate

"ucxbl7" :2.00, // Uc 7th harmonic content rate

"iaxbl7" :2.00, // Ia 7th harmonic content rate

"ibxbl7" :2.00, // Ib 7th harmonic content rate

"icxbl7" :2.00, // Ic 7th harmonic content rate

"time" :"20200108104555" // Time tag, Year-month-day 2020-01-08, Clock 10:45:55

"isend" :"1"

}

The data is divided into 11 packets, the 10th part:

{

"id":"meter_id", // Meter ID

"iaxb3":2.00, // Ia 3rd current harmonic content

"ibxb3":2.00, // Ib 3rd current harmonic content

"icxb3":2.00, // Ic 3rd current harmonic content

"iaxb5":2.00, // Ia 5th current harmonic content

"ibxb5":2.00, // Ib 5th current harmonic content

"icxb5":2.00, // Ic 5th current harmonic content

"time":"20200108104555" // Time tag, Year-month-day 2020-01-08, Clock 10:45:55

"isend":"0"

}

The data is divided into 11 packets, the 11th part:

{

"id":"meter_id", // Meter ID

"iaxb7":2.00, // Ia 7th current harmonic content

"ibxb7":2.00, // Ib 7th current harmonic content

"icxb7":2.00, // Ic 7th current harmonic content

"time":"20200108104555" // Time tag, Year-month-day 2020-01-08, Clock 10:45:55

"isend":"1"

}

Note: KPM31A/B/C and KPM33A/B upload data in one package, while KPM312 uploads four channels separately.

4.3 Daily data format

Topic Name：MQTT_DAY_DATA

MQTT_DAY_DATA data can be split into multiple packets for transmission, distinguished by the field "isend". When the value of the "isend" field in the received data packet is "1", it means that all data packets have been transmitted.

(Only KPM37 support this function. In wifi communication, it's sent in splitting packages. In 4G communication, it's sent in 1 packages.)

Json data format as below:

The data is split into 3 packets, 1st part is:

```
{
"id": "meter_id",    //Meter ID
"zygdd":100.00,       //Import total active energy at 0:00, float, unit kWh
"fygdd":100.00,       // Export total active energy at 0:00, float, unit kWh
"zwgdd" :100.00,      // Import total reactive energy at 0:00, float, unit kvah
"fwgdd" :100.00,      // Export total reactive energy at 0:00, float, unit kvah
 "time":"20200108000000"    // Time tag，Year-month-day 2020-01-08, Clock 10:45:55
      "isend":"1"
}
```

The data is split into 3 packets, 2nd part is:

```
{
      "id":"meter_id",    // Meter ID
      "zyjsz":100.00,      // Tariff 1 import total active energy at 0:00, float, unit kWh
```

"fyjsz":100.00, // Tariff 1 export total active energy at 0:00, float, unit kWh

"zyfsz":100.00, // Tariff 2 import total active energy at 0:00, float, unit kWh

"fyfsz":100.00, // Tariff 2 export total active energy at 0:00, float, unit kWh

"time":"20200108104555" // Time tag, Year-month-day 2020-01-08, Clock 10:45:55

"isend":"0"

}

The data is split into 3 packets, 3rd part is:

{

"id":"meter_id", // Meter ID

"zypsz":100.00, // Tariff 3 import total active energy at 0:00, float, unit kWh

"fypsz" :100.00, // Tariff 3 export total active energy at 0:00, float, unit kWh

"zyvsz":100.00, // Tariff 4 import total active energy at 0:00, float, unit kWh

"fyvsz":100.00, // Tariff 5 import total active energy at 0:00, float, unit kWh

"time":"20200108104555" // Time tag, Year-month-day 2020-01-08, Clock 10:45:55

"isend":"0"

}

Note: Upload last day's frozen energy reading at 0:00 every day.

4.4 Remote signal data

Topic Name: MQTT_TELEIND

Json data format as below:

```
{
    "id": "meter_id",                    //Meter ID
     "value":"00000003@00000003",       //DI@DO
     "time":"20200108104555"
}
```

Note: DI and DO states are uploaded bit by bit, the above data indicates that DI1 and DI2 are closed, and DO1 and DO2 are also closed. And only upload data when DI and DO functions are selected

4.5 Remote control data

Topic Name: MQTT_TELECTRL_meterid last 8 bits

Json data format as below:

```
{
    "dox":    "1",        //1 means on ,0 means off
```

"oprid":"Opertaion ID" //32 bit strings

}

Note: The x in dox can be 1~32. KPM31B, KPM31C, and KPM33B only support do1. KPM37 supports do1 and do2. KPM312 is a divided loop, each loop also only supports do1. And KPM37 and KPM312 will only work when the DO function is selected.

Remote control respond:

Topic Name: MQTT_TELECTRL_REP

Json data format as below:

```
{
    "id":"meter_id",      //Meter ID
    "dox":"1",            //1 means on ,0 means off
    "oprid":"Operation id"        //32 bit strings"
    "code":"01/02"          //01 succeeded; 02 failed
    "msg": "xxxx";          //Fail reason
}
```

Note: The x in dox can be 1~32. The KPM37 only supports do1 and do2.

4.6 Data recall

Topic Name: MQTT_RECALL_ meterid last 8 bits

(Only KPM37 and KPM312 support this function. In wifi communication, it's sent in splitting packages. In 4G communication, it's sent in 1 package.)

Json data format as below:

Platform issue:

```
{
    "date":"yyyyMM",            //year month, eg: 202207
    "oprid" :"Operation id",    //32 bit strings",
    "oprtype":"2"               //2 means recall monthly frozen data
}
```

Recall meter respond:

Topic Name: MQTT_RECALL_REP

Json data format as below, split into 4 packets:

1st packet:

```
{
    "id": "meter_id",                   //Meter ID
     "oprid":"Operation id"    //32 bit strings",
     "code":"01/02"              //01 succeeded; 02 failed
    "msg": "xxx"
    "frz_type": "2",          //Frozen type,    2 means monthly
    "time" :"202001",     //Corresponding time to the frozen value, the
month is yyyyMM, and the day is yyyyMMdd
    "isend" :"0"
}
```

2nd packet:

```
{
```

"id": "meter_id", //Meter ID

"frz_type": "2", //Frozen type, 2 means monthly

"zygdd" :100, // Import total active energy, float, unit kWh

"fygdd" :100, // Export total active energy, float, unit kWh

"zwgdd" :100, // Import total reactive energy, float, unit kvarh

"fwgdd" :100, // Export total reactive energy, float, unit kvarh

"time" :"202001" // Corresponding time to the frozen value, the month is yyyyMM, and the day is yyyyMMdd

"isend" :"0"

}

3rd packet:

{

"id": "meter_id", //Meter ID

"frz_type": "2", // Frozen type, 2 means monthly

"zyjdd" :100, // Import active tariff 1 energy, float, unit kWh

"fyjdd" :100, // Export reactive tariff 1 energy, float, unit kWh

"zyfdd" :100, // Import active tariff 2 energy, float, unit kWh

"fyfdd" :100, // Export reactive tariff 2 energy, float, unit kWh

"time" :"202001" // Corresponding time to the frozen value, the month is yyyyMM, and the day is yyyyMMdd

"isend" :"0"

}

4th packet:

{

"id": "meter_id", //Meter ID

"frz_type": "2", // Frozen type, 2 means monthly

"zypdd" :100, // Import active tariff 3 energy, float, unit kWh

"fypdd" :100, // Export active tariff 3 energy, float, unit kWh

"zyvdd" :100, // Import active tariff 4 energy, float, unit kWh

"fyvdd" :100, // Export active tariff 4 energy, float, unit kWh

"time" :"202001" // Corresponding time to the frozen value, the month is yyyyMM, and the day is yyyyMMdd

"isend" :"1"

}

4.7 Time synchronization

Single meter time synchronization (All MQTT meter supported)

Topic : MQTT_SETTIME_（meterid last 8 bits）

{

"oprid":"Operation id" // 32 bit strings

"time":"yyyymmddhhmmss"

}

Meter respond:

Topic: MQTT_METER_TIME_REP

{

```
    "id":"meter_id",            //Meter ID
    "oprid":"Operation id"   // 32 bit strings
     "code":"01/02"            //01 succeeded；02 failed
      "msg":"xxx"              // Failed reason
}
```

4.8 Parameter setting

Platform publish setting parameter topic, meter subscribes. (Only KPM37 and KPM312 supported)

Topic：MQTT_SYS__CFG_（meterid last 8 bits）

```
{
   "oprid":"Operation id"     //32 bit strings
   "addr":"0000",             //modbus register address，hexadecimal
   "value":"6666;1;2;3…n",    //Wroten value, n≤100
   "type":"1"                 //1. Integer  2.float
}
```

Meter respond：

Topic：MQTT_SYS_SET_REP // Meter publish, platform subscribes

```
{
    "id":"meter_id",           //Meter ID
     "oprid":"Operation id"    //32 bit strings
    "code":"01/02"             //01 succeeded 02 failed
```

}

4.9 Platform reading meter parameters

Platform publish reading meter parameters topic, meter subscribe

Topic: MQTT_SYS_READ_meterid (last 8 bits)

```
{
   "oprid":"Operation id"     //32 bit strings
   "addr": "0000",            //modbus register address, hexadecimal
   "lenth":"n",               //Length of read parameter, n≤100
   "type":"1"                 //Data type 1. Integer   2.float
}
```

Topic: MQTT_SYS_REPLY // Meter publish, platform subscribes

```
{
    "id":   "meter_id",        //Meter ID
   "oprid":"Operation id"     //32 bit strings
    "addr": "0000",           //modbus register address, hexadecimal
   "lenth":"n",               //Length of read parameter, n≤100
    "value":"6666",           //Read value
     "type":"1"               // Data type 1. Integer   2.float
}
```

4.10 Meter MQTT parameter reconfiguration

The platform publishes the topic, the meter subscribes to the topic and returns the result. (All MQTT meters supported)

1）Platform publish topic

Topic：MQTT_RECONFIG_meterid　(last 8 bits)

```
{
    "oprid":"Operation id"     //32 bit strings

}
```

2）**Meter receives and returns the results**

Topic：MQTT_RECONFIG_REPLY

```
{
   "id":"meter_id", //Meter ID
   "oprid":"Operation id"     //32 bit strings
   "code":"01/02"             //01 succeeded 02 failed
   "msg":"xxx"                //Failed reason
}
```

4.11 Read/set MQTT upload frequency

1) Platform publish set upload frequency topic, meter subscribe

Topic：MQTT_COMMOD_SET_（meterid last 8 bits）

```
{
   "oprid":"Operation id"     //32 bit strings
```

"Cmd":"0000", //Command CMD 0000 is second-level data setting. 0001 is minute-level data setting

"value":"30", //Wroten value

"types":"1" //1.Integar 2.Float

}

Note: Second level (in seconds): 30, 60, 300, 600, 900, 1200, 1800, 3600

Minute level (in minutes): 1, 5, 10, 15, 20, 30, 60, 1440

Meter response：

Topic: MQTT_COMMOD_SET_REP // **Meter publish, platform subscribes**

{

"id":"meter_id", //Meter ID

"oprid":"Operation id" //32 bit strings

"code":"01" //01 Success 02 Fail

}

2) Platform publish Read upload frequency topic, meter subscribe

Topic: MQTT_COMMOD_READ_（meterid last 8 bits）

{

"oprid":"Operation id" //32 bit strings

"Cmd":"0000", // Command CMD 0000 is second-level data setting. 0001 is minute-level data setting

"types":"1" //1.Integar 2.Float

```
}
```

Meter response：

Topic：MQTT_COMMOD_READ_REP // **Meter publish, platform subscribe**

```
{
   "id":"meter_id"              //Meter ID
   "oprid":"Operation id"     //32 bit strings
   "Cmd":"0000"                // Command CMD 0000 is second-level data
                                  setting. 0001 is minute-level data setting
   "value":"30"          //Read value
   "code":"01"          //01 Success 02 Fail
}
```

5. MQTT Communication parameter configuration message (maintenance platform)

5.1 Meter Request Configuration

Topic:MQTT_METER_REQ_CFG

```
{
   "code":"Meter ID"
}
```

5.2 Meter configuration success response

Topic：MQTT_METER_REPLY

```
{
```

"code":"Meter ID",

"status":"ok"

}

5.3. Meter configuration fail response

Topic：MQTT_METER_REPLY

{

"code":" Meter ID",

"status":"fail"

}

5.4 Configuration data issued by the platform

5.4.1.WIFI meter

Topic：MQTT_SET_meterid　(last 8 bits)

{

"info":"ip_addr;port;mqtt_account;mqtt_password;WIFI_acount;WIFI_passw

ord;WPA2_method;WIFI_usrname2;WIFI_usrpwd2"

}

Note： WPA2_method; WIFI_usrname2; WIFI_usrpwd are newly added.

WPA2_method with 3 options：TLS ,PEAP,TTLS. Default blank.

WIFI_usrname2：Stage 2 user name，1~32 characters. Only displayed when WPA2_method is set to PEAP/TTLS, and the corresponding fields need to be filled in.

WIFI_usrpwd2：Stage 2 user password，1~32 characters. Only displayed when WPA2_method is set to PEAP/TTLS, and the corresponding fields need to be filled in.

5.4.2.4G meter

Topic：MQTT_SET_meterid　(last 8 bits)

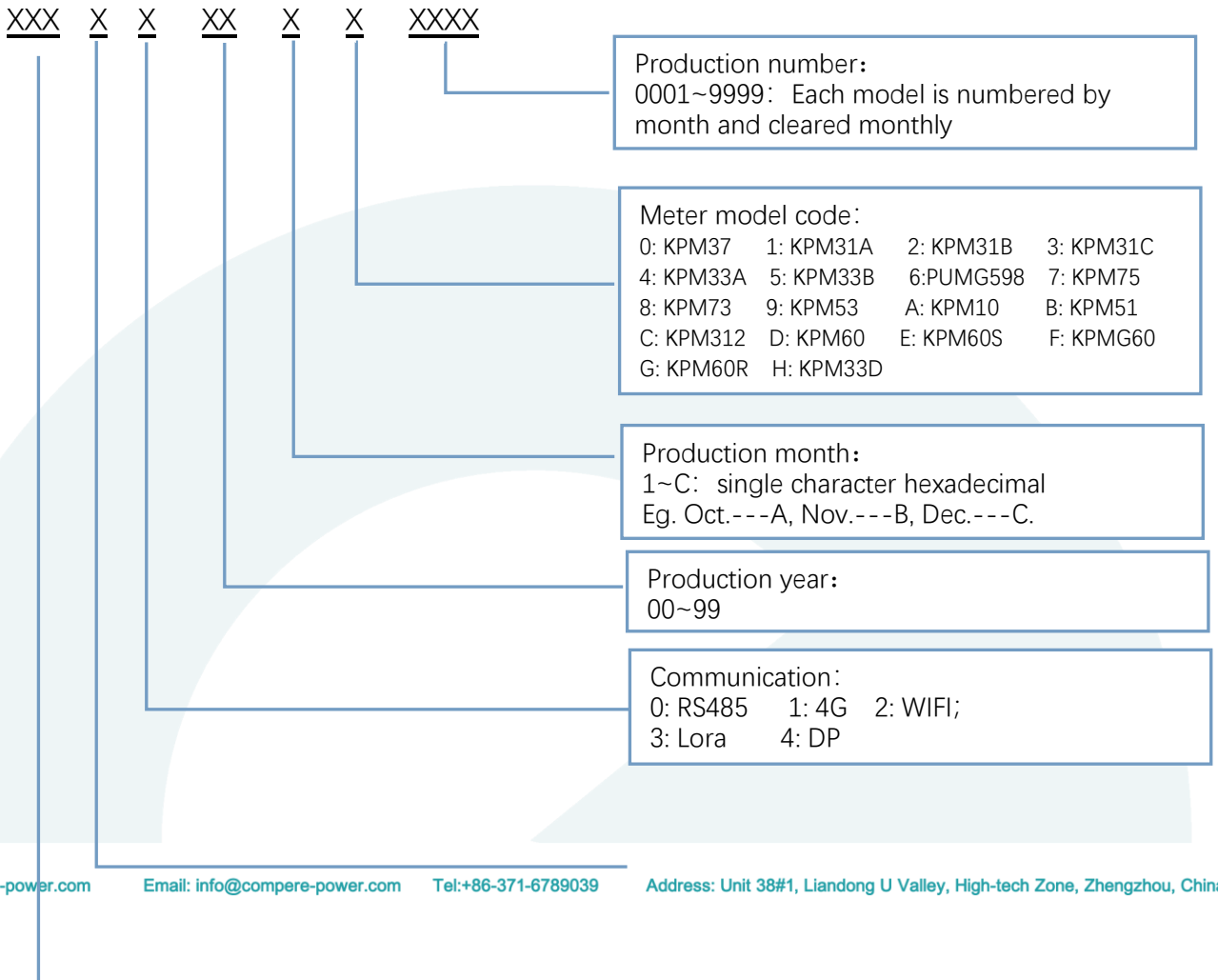
{

"info": "ip_addr;port;mqtt_account;mqtt_password"

}

6. Meter code rules

6.1 MQTT protocol meter code total 13 bits



Maintenance platform:
0: China
1: Abroad

Meter model code:
KPM37:307； KPM31A:31A； KPM31B:31B;
KPM31C:31C; KPM33A:33A; KPM33B:33B;
PUMG598:598； KPM75:750； KPM73:730；
KPM53:530; KPM10:130; KPM51:510;
KPM312:312； KPM60:650； KPM60S： 60S

6.2 Compere Water Meter

XXXX X X X X XXXX

Production serial number:
0001~9999: Each model is numbered by month and cleared every month.

Water Meter Code:
S

Production month：
1~C： single character hexadecimal
Eg. Oct.---A, Nov.---B, Dec.---C.

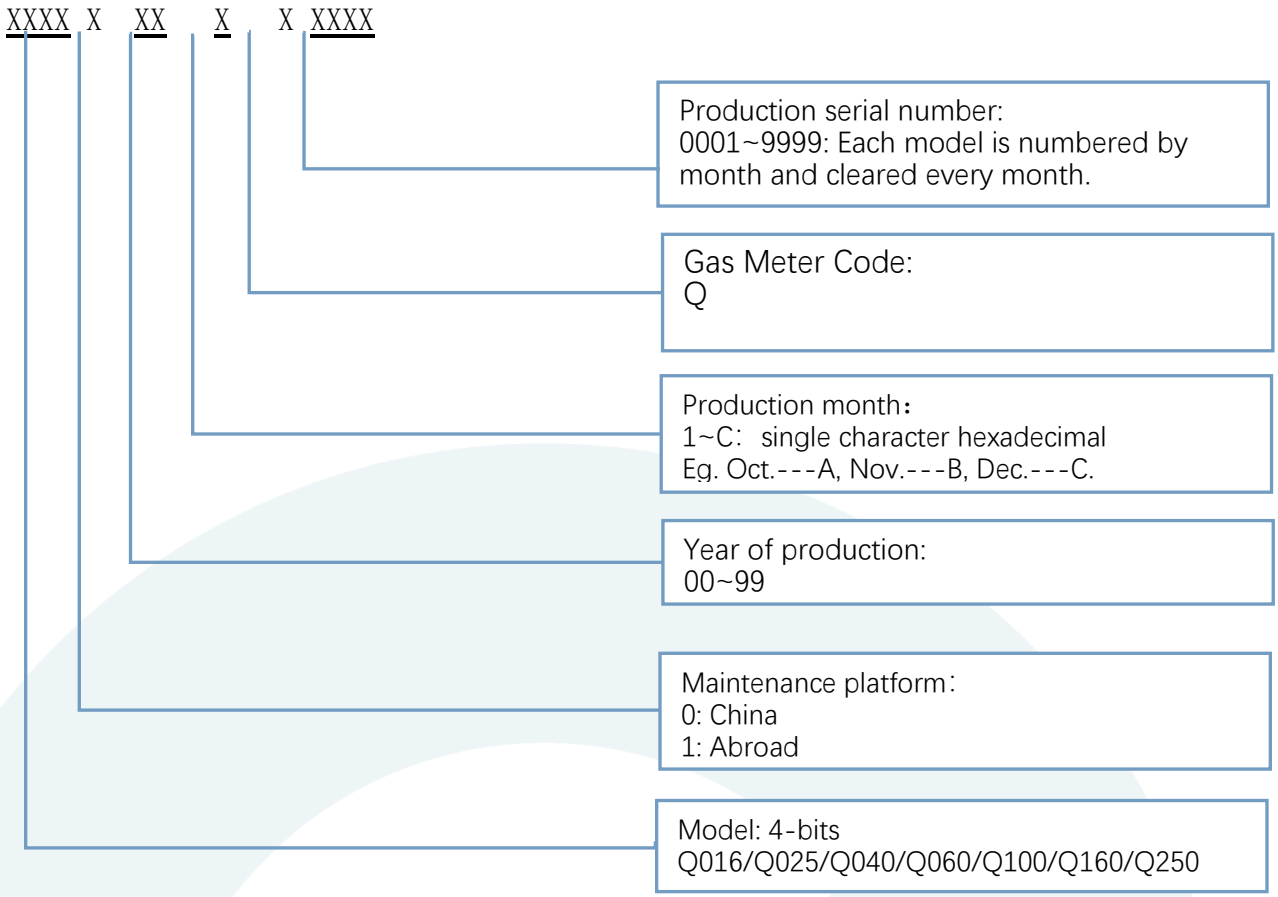
Year of production:
00~99

Maintenance platform:
0: China
1: Abroad

Model: 4-bits
W015/W020/W025/W032/W040/W050/W065/W080/W100/W150/W200/W250/W300

Note: Diameter DN15: corresponds to W015 in the code, and the others are similar.

6.3 Compere Gas Meter



Note: G1.6 in the specification corresponds to Q016 in the above code, and the others are similar.

6.4 Non MQTT protocol meter code total 13 bits