

4G 网关 Modbus 配置说明

一、版本适配说明

说明：本篇文档所适用的网关型号为：**XY-3422**。使用的配置工具的最低版本为 **v1.1.4**。

二、概览。

说明：目前 modbus 配置只支持配合 mqtt 通道使用，且仅支持 modbus-RTU 协议。

Modbus 配置与数据流模板的关系：上行数据流先经过 modbus 解析，然后再经过数据流模板处理；下行数据流先经过数据流模板解析，再经过 modbus 解析。

Modbus 配置分为自定义 modbus 配置和预置空调协议配置，配置页面如下图所示：

注：此部分需单独配置，暂只支持配合MQTT通道使用。

导入JSON文件 导出JSON文件 写入配置 读取配置

是否启用： 启用 不启用

协议类型： RTU

配置方式： 自定义modbus配置 预置空调协议

添加串口

串口配置1

绑定串口id： 1 2 3

读取方式： 周期读取 定时读取

轮询周期： (单位：s)

指令下发间隔： (单位：ms)

解析方式： 整包解析 逐行解析

时间戳： 添加 不添加

属性名前缀字符串：

新增规则 重置

规则1

设备ID： 提示：多个设备id之间用英文逗号隔开

读操作 =====>>> >

写操作 =====>>> >

图 1 自定义 modbus 配置

基本参数	串口参数	网络通道参数	预置信息	自动采集任务	GPIO	数据流	预警	任务	ModBus	DLT645
------	------	--------	------	--------	------	-----	----	----	--------	--------

注：此部分需单独配置，暂只支持配合MQTT通道使用。

是否启用： 启用 不启用

协议类型： RTU

配置方式： 自定义modbus配置 预置空调协议

绑定串口id： 1 2 3

轮询周期： (单位: s)

指令下发间隔： (单位: ms)

解析方式： 整包解析 逐行解析

属性名前缀字符串： ⓘ

设备ID： 提示: 多个设备id之间用英文逗号隔开

图 2 预置空调协议配置

通过配置 modbus 解析参数，可以实现：网关自动下发 modbus 读指令，并将从机的 modbus 响应数据转化为 JSON 格式数据包，包内数据的属性名由用户自定义。也可以将用户通过 mqtt 服务器下发的 JSON 格式的消息转化为 modbus 写指令，然后写入指定串口。

三、配置项说明。

1、自定义 modbus 配置

The screenshot shows a web-based configuration interface for Modbus. At the top, there is a button labeled "添加串口" (Add Serial Port). Below it, a configuration box titled "串口配置1" (Serial Port Config 1) contains the following settings:

- 绑定串口id: Radio buttons for 1 (selected), 2, and 3.
- 读取方式: Radio buttons for 周期读取 (Periodic Read, selected) and 定时读取 (Timed Read).
- 轮询周期: Input field with value 60, unit (s).
- 指令下发间隔: Input field with value 1000, unit (ms).
- 解析方式: Radio buttons for 整包解析 (Whole Packet Parse, selected) and 逐行解析 (Line-by-line Parse).
- 时间戳: Radio buttons for 添加 (Add) and 不添加 (Do not add, selected).
- 属性名前缀字符串: Input field with a help icon.

Below the serial port configuration, there is a button "新增规则" (Add Rule) and a "重置" (Reset) button. A rule configuration box titled "规则1" (Rule 1) contains:

- 设备ID: Input field with a hint: "提示: 多个设备id之间用英文逗号隔开" (Hint: Separate multiple device IDs with commas).
- 读操作: Input field with "====>>>" and a right arrow.
- 写操作: Input field with "====>>>" and a right arrow.

图 3 自定义配置项

- **添加串口:** 点击“添加串口”按钮，会添加一个“串口配置”框，在不同的串口配置框中，可以对指定的串口进行独立的 modbus 配置。
 - **绑定串口 id:** 本串口配置对应的串口 ID。
 - **读取方式:** 可选择周期性读取或者定时读取 modbus 数据。
 - **定时读取:** 选择定时下发读数据指令的时刻，可自定义输入任意时刻，仅支持“hh:MM”格式。
 - **轮询周期:** 两次下发同一条指令之间的间隔。
 - **指令下发间隔:** 每次轮询中的两条指令之间的时间间隔。
 - **解析方式:** 整包解析——每轮指令下发后一次上报本轮响应中解析的全部数据。
逐行解析——每条指令下发后，立即上报本条响应解析的数据。
 - **时间戳:** 若添加时间戳，则解析后的 JSON 会在外层再嵌一层 JSON，并添加时间戳属性:“ts”。
- 示例:
- **属性名前缀字符串:** 配置之后，表格中配置的所有属性名将被解析为 'xxx_属性名' 的格式。如: 填入字符串 'slave1'，则属性名 'remote_mode' 将被解析为 'slave1_remote_mode'。目前支持三个模板字符串: '\$imei', '\$iccid' 和 '\$devid'，其中 devid 将会自动替换为下面

配置项中的“设备 ID”。例如：“设备 ID”项配置的值为：‘2’，“属性名前缀字符串”项配置的值为：‘slave\${devid}’，那么该值将会解析为：‘slave2’。

- **新增规则：**点击“新增规则”按钮，会添加一个“规则配置框”，可以用来配置不同 modbus 协议的设备。

- **设备 ID：**modbus 指令中的设备 id（10 进制数字），有多个设备 id 时，会按顺序发完每个设备的查询指令。若多个设备的读写内容完全相同，可以在一个规则中统一配置，如果不同，则需要“新增规则”。

- **读写操作**

注：配置读写操作时，“新建一行”即为新增一条指令，读操作中的所有行作为一轮指令。

(一) 读操作



图 4 读操作配置表

点击“新增一行”后，添加一行指令配置行，填入功能码等。

- (1) 功能码：modbus 协议功能码，读操作的功能码支持 1,2,3,4（10 进制）
- (2) 起始地址：以功能码 3 为例，读取的寄存器起始地址。（10 进制）
- (3) 数量：以功能码 3 为例，读取的连续寄存器的个数。（10 进制）
- (4) 状态：若为停用，则网关不会下发本条指令。
- (5) 备注：用户自定义备注。（尽量言简意赅）
- (6) 操作：新建解析——在本行（本条指令）下的子表中新增一行解析。
 停用/启用——停用或启用本条指令。
 删除——删除本条指令。

子表是对从机响应数据的解析，说明（以功能码 3 为例）：

上图中的查询指令为（16 进制）：01 03 00 01 00 03 54 0B

- 01 —— 设备地址（设备 id），1 个字节
- 03 —— 功能码，1 个字节
- 00 01 —— 寄存器起始地址，2 个字节
- 00 03 —— 查询寄存器数量，2 个字节
- 54 0B —— CRC 校验码，2 个字节

那么设备响应的数据为（16 进制，数据区是随机填写的）：01 03 06 00 13 00 22 00 74 04

9B

- 01 —— 设备地址，1 个字节
- 03 —— 功能码，1 个字节

06 —— 数据区字节数，1 个字节
 00 13 00 22 00 74 —— 数据区，n 个字节
 04 9B —— CRC 校验码

(1) 起始字节：数据区的起始字节的下标（从 1 开始），那么上面的数据区的每个字节对应的下标为 1,2,3,4,5,6。

(2) 数据类型：Int16——2 个字节的无符号短整型数据，即从起始字节开始的两个字节解析成一个整型数据。

IntS16——2 个字节的有符号短整型数据。

Int32——4 个字节的整型数据。

IntS32——4 个字节的有符号整型数据。

Float——4 个字节的浮点数。

Bit——位数据。将每个寄存器的 2 字节数据转为 16 位 2 进制数据。从右往左，每一位对应的下标为 1~16。

(3) 起始位：数据类型为 bit 时需填写，从右往左数的起始位下标。

(4) 结束位：数据类型为 bit 时需填写，从右往左数的结束位下标。

(5) 比例：设备采集值和真实值的比值。如 采集值：真实值 = 10:1，那么比例填 10。

(6) 属性名：这个值所对应的属性名，用户自定义，即为上报到服务器的 JSON 包中的属性名。

(7) 状态：是否解析这个值，若停用，则不会添加到 JSON 包中。

(二) 写操作



图 5 写操作配置表格

写操作支持写单个寄存器或线圈，可用功能码：5,6,16（10 进制）。会根据数据类型和功能码对寄存器数量进行限制。6 功能码只支持数据类型为无符号 16 位整型和有符号 16 位整型。16 功能码还可以支持浮点数等。

写操作的参数同上。服务器需下发配置中的属性的 JSON 包，如‘{“xyz”:12.5}’，网关根据属性名下发对应的写命令，将属性值写入寄存器。如果有多个设备 id，可以通过配置“属性名前缀字符串”对不同设备的属性名进行区分，服务器下发的 JSON 包中的属性名需要加上对应的前缀，如：‘{“s1_xyz”:12.5}’，页面中配置的不需要添加前缀。

解析完成后上报到服务器的 JSON 包的格式为（不添加时间戳）：

```
{
  "属性名 1": "值 1",
  "属性名 2": "值 2",
  "属性名 3": "值 3",
  ....
}
```

若添加时间戳，则数据包变为：

```
{
  "ts":xxxxxxxxxxxx,
  "data":{
    "属性名 1":"值 1",
    "属性名 2":"值 2",
    "属性名 3":"值 3",
    ....
  }
}
```

如果对上报的数据有其他要求，可以在“发送数据流模板”中对这个 JSON 数据包进行进一步的处理。

2、预置空调协议

配置方式： 自定义modbus配置 预置空调协议 约克-风冷-YSPA

绑定串口id： 1 2 3

轮询周期： (单位：s)

指令下发间隔： (单位：ms)

解析方式： 整包解析 逐行解析

属性名前缀字符串： ⓘ

设备ID： 提示：多个设备id之间用英文逗号隔开

图 6 预置空调协议配置项

- **配置方式：**选择“预置空调协议”后，可以在右侧下拉框中选择空调型号，对于未收录的空调协议，用户需要自行在“自定义 modbus 配置”中进行配置。
- **绑定串口 ID：**本串口配置对应的串口 ID。
- **轮询周期：**两次下发同一条指令之间的间隔。
- **指令下发间隔：**每次轮询中的两条指令之间的时间间隔。
- **解析方式：**整包解析——每轮指令下发后一次上报本轮响应中解析的全部数据。
逐行解析——每条指令下发后，立即上报本条响应解析的数据。
- **属性名前缀字符串：**同“自定义 modbus 配置”。
- **设备 ID：**modbus 指令中的设备 id（10 进制数字），有多个设备 id 时，会按顺序发完每个设备的查询指令。多个设备 ID 之间使用英文逗号连接，如：'1,2,3'。

在配置完成后，写入配置，网关会在短暂延迟之后将空调协议的点位表返回给配置工具，之后用户可以再对查询点位进行选取、属性名重命名等。

四、示例。

以“气象多要素百叶盒”为例，设备地址出厂默认为 0x01。设备通讯协议是 modbus-rtu 协议。

500到507号寄存器中的内容如下表所示（支持03/04功能码）：

寄存器地址	PLC 或组态地址	内容	操作
500	40501	湿度值（实际值 10 倍）	只读
501	40502	温度值（实际值 10 倍）	只读
502	40503	噪声值（实际值 10 倍）	只读
503	40504	CO2（实际值）	只读
504	40505	0	只读
505	40506	大气压值（单位 Kpa,实际值 10 倍）	只读
506	40507	20W 的 Lux 值高 16 位值(实际值)	只读
507	40508	20W 的 Lux 值低 16 位值(实际值)	只读

图 7 百叶箱寄存器表

现需采集温湿度数据上传到服务器，温度字段名定义为“temperature”，湿度字段名定义为“humidity”。下面开始填入配置项。

< 数据 TCP服务器 预置信息 自动采集任务 GPIO GPS 数据流 预警 任务 ModBus DLT645 HJ212 >

注：此部分需单独配置，暂只支持配合MQTT通道使用。

是否启用: 启用 不启用

协议类型: RTU

配置方式: 自定义modbus配置 预置空调协议

串口配置1

绑定串口id: 1 2 3

读取方式: 周期读取 定时读取

轮询周期: (单位: s)

指令下发间隔: (单位: ms)

解析方式: 整包解析 逐行解析

时间戳: 添加 不添加

属性名前缀字符串:

规则1

设备ID: 提示: 多个设备id之间用英文逗号隔开

读操作 =====>>

功能码	起始地址	数量	状态	备注	操作			
收起	3	500	2	启用	新建解析 停用 删除			
起始字节	数据类型	起始位	结束位	比例	属性名	状态	备注	操作
1	无符号			10	humidity	启用		停用 删除
3	无符号			10	temperat	启用		停用 删除

写操作 =====>>

1、是否启用

启用。

2、协议类型

目前仅支持 MODBUS-RTU 协议。

3、配置方式

非预置空调协议，所以进行自定义配置。

4、绑定串口 ID

设备型号为 XY-3422，有 3 个 RS485 串口，现在使用第二个 485 串口进行接入，对应为串口 2，因此串口 id 选 2。

5、读取方式

这里采用周期读取的方式。

6、轮询周期

用户自定义。这里希望每 60 秒读取一次温湿度数据，因此填 60。

7、指令下发间隔

所有指令下发时间总和不能超过轮询周期，这里只有一条指令，填 5ms ~ 2000ms 均可，不建议间隔时间过久。

8、解析方式

用户自定义。

9、时间戳

用户自定义。

10、属性名前缀字符串

用户自定义。

11、规则

(1) 设备 id

即设备地址，填入 10 进制。这里设备地址为 0x01，10 进制即为 1，因此填 1。

(2) 读操作

由表可知，温湿度都是只读的属性，因此需要在读操作中配置指令。

- **功能码：**使用 0x03 功能码，10 进制即为 3，因此功能码选择 3。
- **起始地址：**湿度的寄存器地址是 500，温度的寄存器地址为 501，因此查询温湿度的起始地址为 500。
- **数量：**查询温湿度两个连续寄存器，则数量填 2。

上面配置的是查询指令，下面的子表中配置的是对设备响应数据的解析。

- **起始字节：**查询指令读取了两个寄存器的数据，那么从机响应帧的数据区为 4 个字节，第 1、2 字节的数据为湿度数据，第 3、4 字节的数据为温度数据。因此湿度的起始字节为 1，温度的起始字节为 3。
- **数据类型：**每个寄存器的数据占两个字节，即 16 位，用两个字节表示一个属性值，因此这里选 Int16。
- **比例：**由表可知，采集值是真实值的 10 倍，因此比例填 10。
- **属性名：**用户自定义（勿填中文）。

到这里配置就完成了。将配置写入网关，可以看到，每隔 60 秒，网关会发送一条 modbus 查询指令。

[通讯端口](#) [串口设置](#) [显示](#) [发送](#) [多字符串](#) [小工具](#) [帮助](#) [联系作者](#) [大虾论坛](#)

```
[15:54:42.708]收←◆01 03 01 F4 00 02 84 05
[15:55:42.708]收←◆01 03 01 F4 00 02 84 05
[15:56:42.709]收←◆01 03 01 F4 00 02 84 05
```

网关会将从机返回的数据解析为：

```

{
  "ts":1666267200,
  "data":{
    "temperature":26.3,
    "humidity":65.7
  }
}

```

并将此 JSON 数据发布到用户配置的 MQTT 主题。

用户还可以将这页的 modbus 配置导出为一个 JSON 文件，以便后续配置相同设备时，直接导入这个 JSON 文件即可。

上述配置导出的 JSON 文件内容为：

```

{
  "MODBUS_ENABLE": 1,
  "CUSTOM": 1,
  "UART_CMD_TABLE": [
    {
      "UID": 2,
      "QUERYSYCLE": 60,
      "INTERVAL": 1000,
      "PARSE_ALL": 1,
      "PREFIX": "",
      "ADD_TS": 1,
      "CMD_TABLE": [
        {
          "SLAVE_NUM": [
            "1"
          ],
          "MRC": [],
          "MRD": [],
          "MRK": [
            {
              "addr": "500",
              "count": "2",
              "return": [
                {
                  "name": "",
                  "enable": 1,
                  "scale": "10",
                  "attribute": "humidity",
                  "key": "r_humidity",
                  "type": "Int16",
                  "start": "1"
                }
              ]
            }
          ]
        }
      ]
    }
  ]
}

```

```
        "name": "",
        "enable": 1,
        "scale": "10",
        "attribute": "temperature",
        "key": "r_temperature",
        "type": "Int16",
        "start": "3"
    }
},
"enable": 1,
"remark": ""
}
],
"MRI": [],
"SC": [],
"SW": [],
"WM": []
}
]
}
}
```